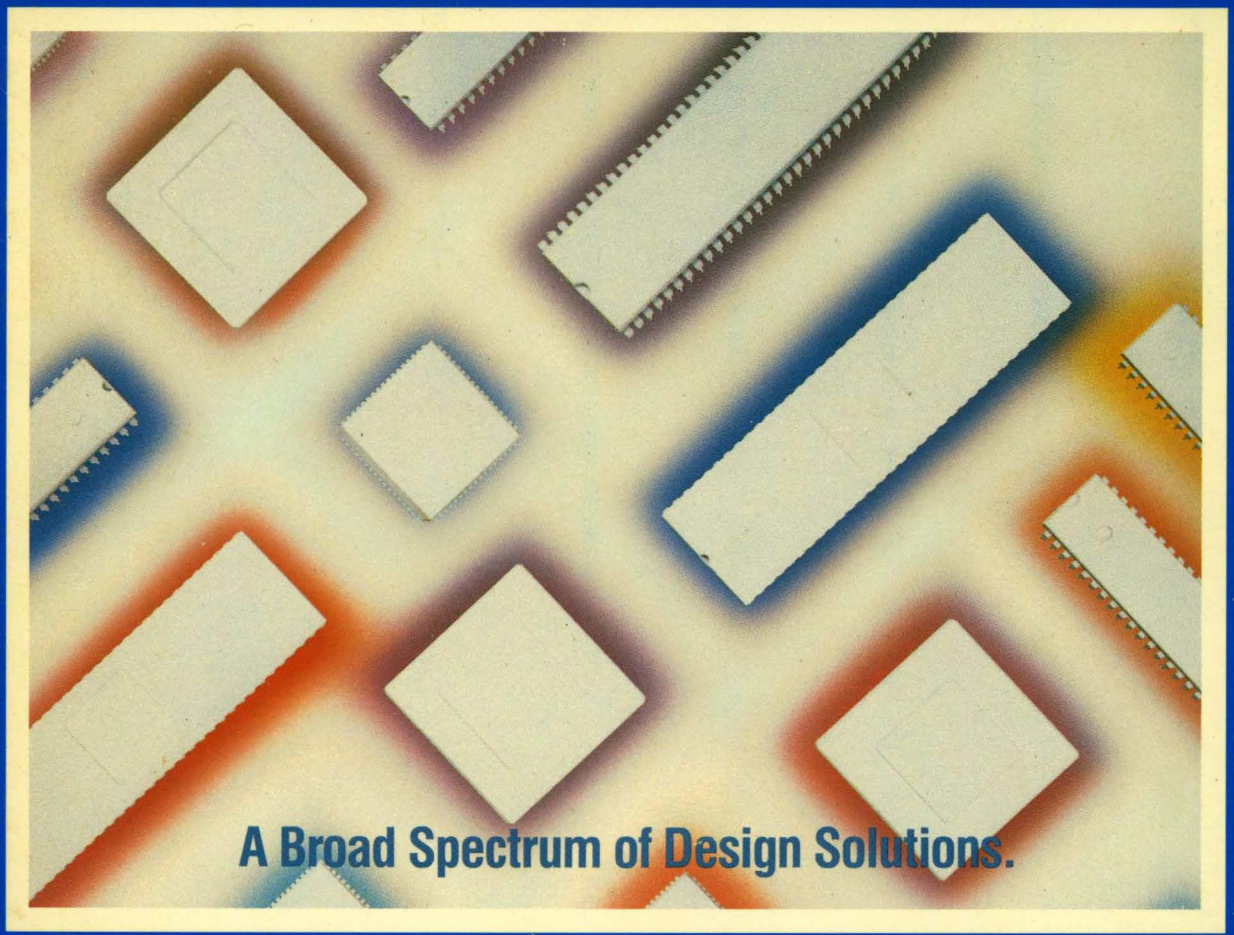
DL139

MOTOROLA



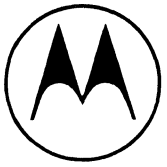
MICROPROCESSOR, MICROCONTROLLER AND PERIPHERAL DATA

VOLUME I

MOTOROLA MICROPROCESSOR, MICROCONTROLLER AND PERIPHERAL DATA

VOL. I

Q3/88
DL139



MOTOROLA

MICROPROCESSOR DATA

VOLUME I

Prepared by
Microprocessor Products Group

This book is intended to provide the design engineer with the technical data needed to completely and successfully design a microcomputer-based system. The Technical Summary and Advance Information data sheets for Motorola's microcontroller, microprocessor, and peripheral components are included.

The information in this book has been carefully checked; no responsibility, however, is assumed for any inaccuracies. Furthermore, this information does not convey to the purchase of microelectronic devices any license under the patent rights of the manufacturer.

Additional information on Motorola's new products and system development products are also included. For further marketing and application information, please contact:

Motorola Inc.
Microprocessor Products Group
Microcontroller Division
Austin, Texas 78735
(512) 891-2000

Series A

Printed in U.S.A.

DATA CLASSIFICATION

Product Preview

Data sheets herein contain information on a product under development. Motorola reserves the right to change or discontinue these products without notice.

Technical Summary

Data sheets herein contain information on new products. Specifications and information are subject to change without notice.

Advanced Information

Data sheets herein contain information on new products. Specifications and information are subject to change without notice.

TABLE OF CONTENTS

The Microprocessor/Microcontroller and Peripheral Data book consist of a two volume set. Refer to the table of contents and the master index for division of the chapters and locations of devices.

MASTER INDEX

MASTER INDEX (Continued)

MASTER INDEX (Concluded)

Motorola's Microcontroller and Microprocessor Families

Volume I

1

MOTOROLA'S MICROCONTROLLER AND MICROPROCESSOR FAMILIES

Motorola manufactures the industry's most complete selection of solid-state microcontroller units (MCU) and microprocessor (MPU), providing the performance and design flexibility needed by the design engineer.

Motorola's family concept has been extremely popular in the MCU industry. This family concept was pioneered with the introduction of the M6800 Family 1974. Four families have evolved from the M6800 Family to fulfill expanding customer requirements. These families are the M68HC11, M6801, M6805, and the M6804. Figure 1-1 illustrates the family evolution.

Numerous peripheral devices have been developed and are available to support the MCUs and MPUs.

M68HC11/M6801/M6805/M6804 FAMILIES

The M68HC11 Family offers high performance in a single-chip MCU with Electronic Eraseable Programmable Read Only Memory (EEPROM), a 16-bit timer, a Serial Communication Interface (SCI), a Serial Peripheral Interface (SPI), and an 8-bit Analog-to-Digital (A/D) converter. The M6801 Family includes high performance in a single chip with Eraseable Programmable Read Only Memory (EPROM) and SCI. The rapidly expanding M6805 Family is available in a variety of memory and package sizes with various special Input/Output (I/O) functions. The M6805 is available in High-Density N-Channel Metal Oxide Silicon (HMOS), Complementary Metal Oxide Silicon (CMOS), and High-Density Complementary Metal Oxide Silicon (HCMOS). The M6804 Family now provides the 8-bit processing capabilities that compete in the 4-bit price arena. A One Time Programmable Read Only Memory (OTPROM) is also available in the M68HC11, M6804, and M6805 Families.

Technology

Motorola's first MCUs and MPUs were produced in HMOS which offered a low cost single-chip solution in high production volumes. CMOS was then introduced which offered very low power consumption and a wide power supply tolerance at performance levels similar to HMOS. The introduction of HCMOS offered the best of both worlds, with high-density and low power consumption. Tables 1-1 and 1-2 list Motorola's MCUs, MPUs, and peripheral product line by technology.

ROM Size

The mask ROM capacities of the present single-chip MCUs range from a low of 512 bytes in the M6804 Family to a high of 8K in the M68HC11 Family. Refer to Table 1-3 through 1-7 to determine what ROM is offered in the MCU product line. In selecting ROM size, the ROM usage efficiency of the instruction set should be considered, along with the application to be programmed.

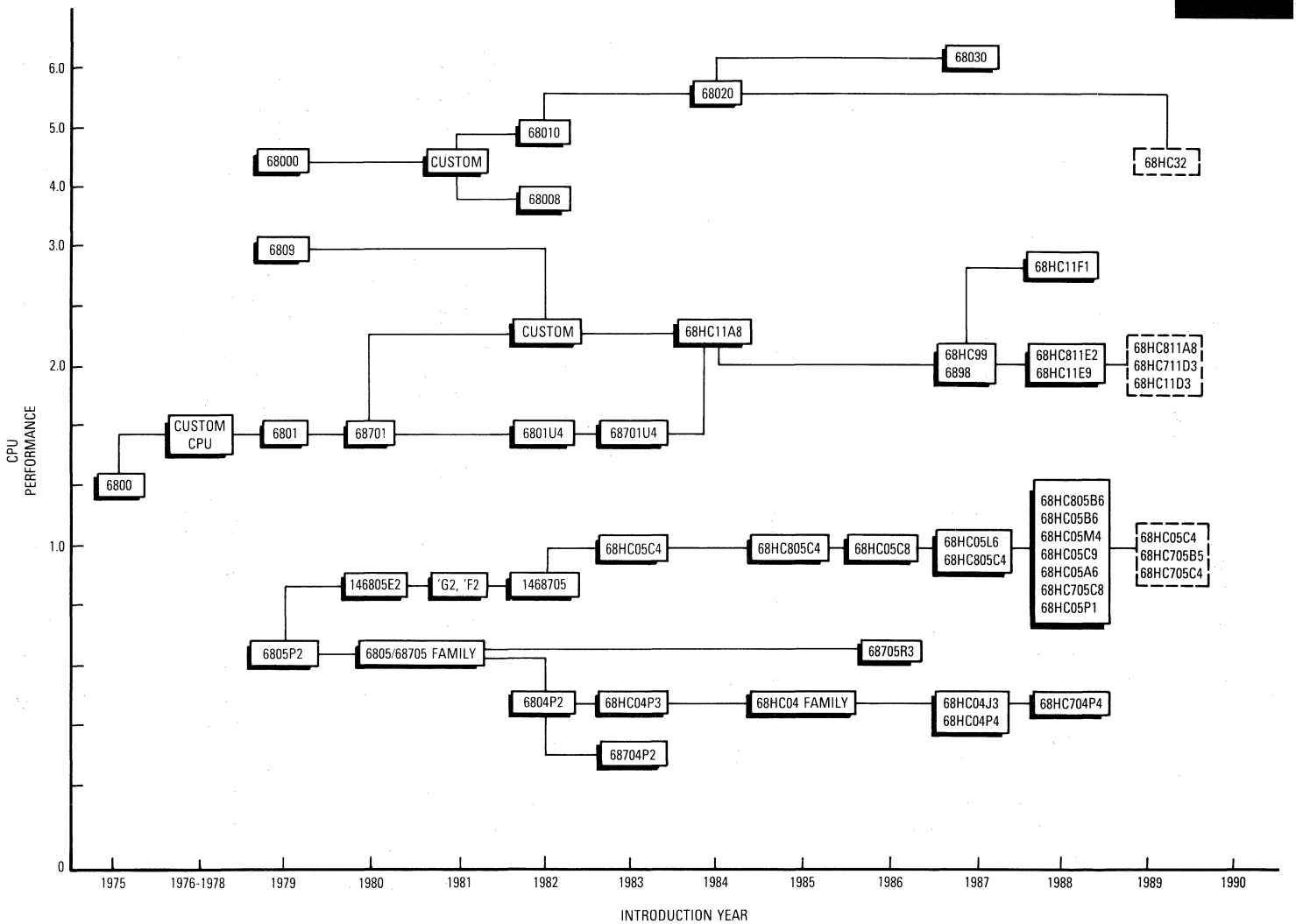


Figure 1-1. Motorla MCU/MPU Evolution

Table 1-1. MCU/MPU Technology Listing

HMOS/NMOS		HCMOS		CMOS
MC6800	MC68705P3	MC68HC04J2	MC68HSC05C4	MC146805E2
MC6801	MC68705P5	MC68HC04J3	MC68HSC05C8	MC146805F2
MC6801U4	MC6805R2	MC68HC04P4	MC68HC705B5	MC146805G2
MC68701	MC6805R3	MC68HC704P4	MC68HC705C8	
MC68701U4	MC68705R3	MC68HC05A6	MC68HC805C4	
MC6802	MC68705R5	MC68HC05B4	MC68HC11A0	
MC6803	MC6805S2	MC68HC05B6	MC68HC11A1	
MC6803U4	MC6805S3	MC68HC805B6	MC68HC11A8	
MC6804J1	MC68705S3	MC68HC05C2	MC68HC11D3	
MC6804J2	MC6805U2	MC68HC05C3	MC68HC11E1	
MC6804P2	MC6805U3	MC68HC05C4	MC68HC11E9	
MC68704P2	MC68705U3	MC68HC05C8	MC68HC11F1	
MC6805P2	MC68705U5	MC68HC05C9	MC68HC711A8	
MC6805P6	MC6809/9E	MC68HC05L6	MC68HC711D3	
		MC68HC05M4	MC68HC711E9	
		MC68HC05P1	MC68HC811E2	
		MC68HCL05C4		
		MC68HCL05C8		

Table 1-2. Peripheral Technology Listing

HMOS/NMOS		HCMOS		CMOS
MC6810	MC6821	MC68HC24	MC68HC34	MC146818
MC6840	MC6844	MC68HC99		MC146818A
MC6845	MC6850			MC146823
MC6852	MC6854			
MC6898	MC68488			
MC2672	MC2674			

Non-Mask ROM Versions

EEPROM, EPROM, OTPROM, and/or non-ROM versions are offered in practically all single-chip MCUs. These versions serve for limited to high volume applications, prototype debugging, and field trials. EEPROM and OTPROM versions are available in the M6805 and M68HC11 Families. EPROM versions are available in the M6805 and M6801 Families. Refer to Table 1-3 through 1-7 to determine what is offered in the MCU product line.

RAM Size

On-chip Random Access Memory (RAM) sizes range from 30 bytes in the M6804 Family to 512 bytes in M68HC11 Family. The M6805 has versions of 64, 104, 112, and 176 bytes. Architectures such as the M68HC11, M6801, and M6805 Families, which permit multi-level subroutines plus ROM and RAM data tables, allow trade-off ROM and RAM utilization. ROM usage can be minimized with subroutines and look-up tables, while RAM usage can be optimizes with ROM tables and fewer subroutines.

Digital Input/Output

Single-chip MCUs are available in 52-pin quad packages as well as the smaller (and lower cost) 20-pin packages. Five to fourteen pins serve power and control functions permitting up to 12 I/O

pins in a 20-pin package and up to 38 I/O pins in the 48/52 pin verions. All of the MCUs offer essentially any mix of inputs and outputs. Higher output drive current is available in the M6805 Family.

Expansion Bus

The non-ROM versions include a bus to access off-chip program memory and additional I/O. The M6801 Family also includes a three bus structure for off-chip expansion. The three bus structure permits the number of bus pins to be optimized for the amount of address space needed off-chip. The M68HC11 Family can operate in an expanded mode and address up to 64K bytes of external memory.

Interrupts

When an application program must synchronize with two or more external events, interrupt hardware in some form is usually necessary. The M68HC11, M6801, and M6805 Families include fully automatic interrupts (registers are saved) with programmable vectors for both an external and internal timer.

Timers

Timers are the most frequently used on-chip functions. Timers may generate interrupts to a program at a periodic rate, measure external events, and generate measured output waveforms. The M68HC11, M6801, and M68HC05 devices include a 16-bit timer that may be used to perform three of the preceeding functions simultaneously. The M6805 and M6804 timers consist of a programmable 8-bit counter and selectable 7-bit prescaler.

Special Functions

Various members of the MCU Families include additional I/O functions. For example, the M68HC11, M6801, and some of the M6805 Family include a SCI. The SCI is used for long-range communications, as in data transfer from an MCU to a terminal or modem. The M68HC11 Family and some of the M6805 Family also contain a SPI. The SPI is used primarily for serial communication between chips on the same printed circuit board. Selected members of the M68HC11 and M6805 Family include multi-channel A/D converters. The MC6805R/S versions contain four analog input channels, and the M68HC11 MCUs features up to eight analog input channels.

DEVELOPMENT SUPPORT

The M68HC11, M6801, and M6804, and M6805 Families are fully supported by a series of economical evaluation modules (EVM). A more powerful development system is also available in the HDS-300. The support products are covered in more detail in **Chapter 5 Evaluation Modules.**

SINGLE-CHIP SELECTOR GUIDES

Tables 1-3 through 1-7 list the different features available for devices within a family. The tables provide information as to RAM, ROM, EPROM, timer, etc. Table 1-8 lists the OTPROM devices available.

Table 1-3. M6801 Family Selector Guide

DEVICE	TECHNOLOGY	PINS	RAM (BYTES)	ROM (BYTES)	EPROM (BYTES)	I/O	TIMER (BITS)	EXT. MEM. BUS	SCI	PACKAGING
6801	HMOS	40	128	2048	—	29	16	64K	Yes	P,S
68701	HMOS	40	128	—	2048	29	16	64K	Yes	S
6803	HMOS	40	128	—	—	13	16	64K	Yes	P,S
6801U4	HMOS	40	192	4096	—	29	16	64K	Yes	P,S
68701U4	HMOS	40	192	—	4096	29	16	64K	Yes	S
6803U4	HMOS	40	192	—	—	13	16	64K	Yes	P

Definitions:

P = Plastic
S = Cerdip
I/O = Input/Output
SCI = Serial Communication Interface
RAM = Random Access Memory
ROM = Read Only Memory
EPROM = Eraseable Programmable ROM

Table 1-4. M6804 Family Selector Guide

DEVICE	TECHNOLOGY	PINS	RAM (BYTES)	ROM (BYTES)	EPROM (BYTES)	I/O	TIMER (BIT)	PACKAGING
6804P2	HMOS	28	30	1016	—	20	8	P,FN
68704P2	HMOS	28	30	—	1020	20	8	S
6804J1	HMOS	20	30	512	—	12	8	P
6804J2	HMOS	20	30	1000	—	12	8	P
68HC04P4	HCMOS	28	172	3700	—	20	8	P
68HC04J2	HCMOS	20	30	1000	—	12	8	P
68HC04J3	HCMOS	20	122	1672	—	12	8	P
68HC704P4	HCMOS	28	172	—	3700	20	8	S

Definitions:

P = Plastic
S = Cerdip
FN = Plastic Leaded Chip Carrier
I/O = Input/Output
RAM = Random Access Memory
ROM = Read Only Memory
EPROM = Eraseable Programmable ROM

Table 1-5. M6805 Family Selector Guide

DEVICE	TECHNOLOGY	PINS	RAM (BYTES)	ROM (BYTES)	EPROM (BYTES)	I/O	TIMER BIT	A/D	SPI	PACKAGING
6805P2	HMOS	28	64	1110	—	20	8	—	—	P,S,FN
6805P6	HMOS	28	64	1804	—	20	8	—	—	P,S,FN
68705P3	HMOS	28	112	—	1804	20	8	—	—	S
68705P5	HMOS	28	112	—	1804	20	8	—	—	S
6805R2	HMOS	40/44	64	2048	—	32	8	Yes	—	P,S,FN
6805R3	HMOS	40/44	112	3776	—	32	8	Yes	—	P,S,FN
68705R3	HMOS	40	112	—	3776	32	8	Yes	—	S
68705R5	HMOS	40	112	—	3776	32	8	Yes	—	S
6805S2	HMOS	28	64	1480	—	21	8	Yes	Yes	P,S,FN
6805S3	HMOS	28	104	2720	—	21	8	Yes	Yes	P,S,FN
68705S3	HMOS	28	104	—	3752	21	8	Yes	Yes	S
6805U2	HMOS	40/44	64	2048	—	32	8	—	—	P,S,FN
6805U3	HMOS	40/44	112	3776	—	32	8	—	—	P,S,FN
68705U3	HMOS	40	112	—	3776	32	8	—	—	S
68705U5	HMOS	40	112	—	3776	32	8	—	—	S

Definitions:

P = Plastic
S = Cerdip
FN = Plastic Leaded Chip Carrier
I/O = Input/Output
RAM = Random Access Memory
ROM = Read Only Memory
EEPROM = Eraseable Programmable ROM
SPI = Serial Peripheral Interface
A/D = Analog/Digital Converter

Table 1-6. M6805 HCMOS/CMOS Family Selector Guide

DEVICE	TECHNOLOGY	PINS	RAM (BYTES)	ROM (BYTES)	EPROM (BYTES)	EEPROM (BYTES)	I/O	TIMER BIT	SPI	SCI	A/D	PACKAGING
68HC05A6	HCMOS	40/44	176	4160	—	2056	32	16	Yes	Yes	—	P,FN
68HC05B4	HCMOS	48/52	176	4160	—	—	32	16	—	Yes	Yes	P,FN
68HC05B6	HCMOS	40/52	176	5952	—	256	32	16	—	Yes	Yes	P,FN
68HC05C2	HCMOS	40	176	2096	—	—	32	16	—	—	—	P
68HC05C3	HCMOS	40	176	2096	—	—	32	16	Yes	Yes	—	P
68HC05C4	HCMOS	40/44	176	4160	—	—	32	16	Yes	Yes	—	P,FN
68HC05C8	HCMOS	40/44	176	7700	—	—	32	16	Yes	Yes	—	P,FN
68HC05L6	HCMOS	68	176	6208	—	—	32	16	Yes	—	—	FN
68HC05M4	HCMOS	52	128	4K	—	—	32	8/16	—	—	Yes	FN
68HCL05C4	HCMOS	40/44	176	4160	—	—	32	16	Yes	Yes	—	P,FN
68HCL05C8	HCMOS	40/44	176	8K	—	—	32	16	Yes	Yes	—	P,FN
68HSC05C4	HCMOS	40/44	176	4160	—	—	32	16	Yes	Yes	—	P,FN
68HSC05C8	HCMOS	40/44	176	8K	—	—	32	16	Yes	Yes	—	P,FN
68HC705C8	HCMOS	40/44	304	—	—	—	32	16	Yes	Yes	—	P,FN
68HC805B6	HCMOS	48/52	176	—	8K	6208	32	16	—	Yes	—	P,FN
68HC805C4	HCMOS	40/44	176	—	—	4160	32	16	Yes	Yes	—	P,FN
146805E2	CMOS	40	112	0	—	—	16	8	—	—	—	P,S,FN
146805F2	CMOS	28	64	1089	—	—	20	8	—	—	—	P,S,FN
146805G2	CMOS	40	112	2106	—	—	32	8	—	—	—	P,S,FN

Definitions:

P = Plastic
S = Cerdip
FN = Plastic Leaded Chip Carrier
I/O = Input/Output
A/D = Analog/Digital Converter
SCI = Serial Communications Interface
SPI = Serial Peripheral Interface
RAM = Random Access Memory
ROM = Read Only Memory
EPROM = Eraseable Programmable ROM
EEPROM = Electrical Eraseable ROM

Table 1-7. M68HC11 Family Selector Guide

DEVICE	TECHNOLOGY	PINS	RAM (BYTES)	ROM (BYTES)	EEPROM (BYTES)	I/O	TIMER BITS	EXT. MEM. BUS	A/D	SPI	SCI	PACKAGING
68HC11A0	HCMOS	48/52	256	—	—	38	16	64K	Yes	Yes	Yes	P,FN
68HC11A1	HCMOS	48/52	256	—	512	38	16	64K	Yes	Yes	Yes	P,FN
68HC11A8	HCMOS	48/52	256	8192	512	38	16	64K	Yes	Yes	Yes	P,FN
68HC11D3	HCMOS	40/44	192	4096	—	30	16	64K	No	Yes	Yes	P,FN
68HC11E1	HCMOS	52	512	0	512	38	16	64K	Yes	Yes	Yes	FN
68HC11E9	HCMOS	52	512	12K	512	38	16	64K	Yes	Yes	Yes	FN
68HC11F1	HCMOS	—	—	—	—	—	—	—	—	—	—	—
68HC811E2	HCMOS	48/52	256	—	2K	38	16	64K	Yes	Yes	Yes	P,FN

Definitions:

P = Plastic
FN = Plastic Leaded Chip Carrier
I/O = Input/Output
A/D = Analog/Digital Converter
SCI = Serial Communication Interface
SPI = Serial Peripheral Interface
RAM = Random Access Memory
ROM = Read Only Memory
EPROM = Eraseable Programmable ROM
EEPROM = Electrical Eraseable ROM

Table 1-8. One-Time Programmable ROM (OTPROM) Devices

Device	OTPROM (Bytes)	RAM (Bytes)	I/O	Timer Bit	A/D, SCI SPI	COP Watchdog	Pin Package
68HC704P4	3740	124	20	8	—	—	28-DIP,DW*
68HC705B5*	6208	176	24	16	A/D, SCI	Yes	52-FN,48-DIP
68HC705C4*[1]	4160	176	24	16	SCI, SPI	Yes	44-FN,40-DIP
68HC705C8	7616	304	24	16	SCI, SPI	Yes	44-FN,40-DIP
68705R3	3776	112	24	8	A/D	—	40-P
68HC711D3*	4096	192	24	16	SCI, SPI	Yes	44-FN,40-DIP
68HC711A8*	8192	256	38	16	A/D, SCI, SPI	Yes	52-FN
68HC711E9*	12K	512	38	16	A/D, SCI, SPI	Yes	48-DIP,52-FN

NOTES:

1. Use MC68HC705C8 for window emulation.
2. Definitions:
   FN = Plastic Quad (PLCC)
   DW = Small Outline (Wide-Body SOIC)
   DIP = Dual-In-Line Package
   RAM = Random Access Memory
   I/O = Input/Output
   A/D = Analog/Digital
   SCI = Serial Communications Interface
   SPI = Serial Peripheral Interface
   COP = Computer Operating Properly
3. *Available in 1989.

Reliability
Volume I

2

MICROPROCESSOR PRODUCTS GROUP
RELIABILITY AND QUALITY ASSURANCE
1987 ANNUAL RELIABILITY REPORT
SUMMARY

INTRODUCTION

The Motorola MOS Microprocessor Reliability and Quality Monitor (R&QA) Program is designed to generate an ongoing data base of reliability and quality performance for various categories of Microprocessor products. The primary purpose of the program is to identify negative trends in the data so that immediate corrective action can be taken. The program also allows Motorola to develop a large data base of reliability and quality results that can be reported quarterly to customers. The following report summarizes the reliability and quality data for 1987.

The reliability monitor tests are conducted on sample groups pulled on a quarterly basis from major categories of products representing a matrix of processing and packaging technologies. Product mix, sample availability and equipment capacity may cause the specific sample groups pulled for a given quarter to vary from quarter to quarter. Each sample group has a specific set of reliability tests associated with it that are appropriate for that product type based on our history for that classification. At the end of each quarter, results are reported for all sample groups that have completed testing. In addition at the end of each year a complete summary of the 4 quarters is reported.

The quality results that are reported are the electrical and visual/mechanical (Average Outgoing Quality (AOQ), given in parts per million defective) for the Microprocessor Group. This data represents the summary of results from the QC gate operation performed on every lot during 1987. Electrical AOQ represents any AC, DC, or functional failure at any temperature (each lot may be typically gated at hot, room or cold temperatures). Visual/mechanical AOQ represents failures such as bent leads, incorrect marking, marking permanency problems, and cracked packages. The AOQ reported is the product the process average (ratio of defective devices to largest sample size) and the lot acceptance rate.

QUALITY AND RELIABILITY SYSTEM

A complete Reliability and Quality Assurance (R&QA) system is in place to monitor and control the performance of Motorola's MOS Microprocessor Components. Incoming Quality Control inspects starting wafers, masks, chemicals, package piece parts, and molding compounds. Process Engineering and In-Process Quality Control perform step-by-step monitoring of the wafer process to check oxidation, diffusion, photolithography, ion implantation, polysilicon deposition, metallization, passivation, and other process operations. Final visual, class probe, and capacitance-voltage plots complete the wafer area inspection. Environmental monitors are also performed for air cleanliness, water quality, temperature, and humidity.

In the assembly area, In-Process Quality Control performs monitors on equipment performance and gate inspections at the major process steps on all lots. The Outgoing Quality Control group continues this philosophy in the final test area by performing electrical and visual-mechanical

gates. The electrical inspection, which consists of AC, DC, and functional tests, is performed to a 0.1% (maximum) Acceptable Quality Level (AQL) sampling plan. The visual/mechanical inspection is also performed to a 0.1% AQL sampling plan. Any lot which fails either of these gates is returned to production for 100% rescreen. An R&QA Engineering organization exists to approve final test programs and support the Outgoing Quality Control organization. Test programs are tailored to assure all required specifications are met or the devices are rejected.

The R&QA Engineering organization is also responsible for performing qualifications of new designs and process changes prior to introduction. In addition, R&QA Engineering establishes and maintains monitor programs to assure processes stay in control once they are qualified. Results from these programs provide rapid feedback to correct problems as they occur.

Supporting these efforts is the Metrology Laboratory which includes both a Standards and a Calibration Laboratory to provide National Bureau of Standards traceability to all production measurements.

Also offering required support are a Chemical Laboratory with such equipment as a gas chromatograph/mass spectrograph and X-ray fluorescent systems for detailed incoming chemical analyses; a Surface Analysis Laboratory whose equipment includes a Scanning Electron Microscope (SEM) and a Scanning Auger Microprobe (SAM); and a Product Analysis Laboratory for detailed analyses of failure modes and mechanisms for Microprocessor devices.

PACKAGING SYSTEM

Motorola Microprocessor devices are produced in plastic, CERDIP, PGA, and sidebraze packages. The ceramic package types are hermetically sealed to protect the integrated circuit from environmental factors and permit operation over extreme temperature ranges. Although plastic devices are not hermetic, modern epoxies exhibit extremely high moisture resistance, and long lifetimes may therefore be expected from these devices in typical environments.

In recent years, plastic encapsulated devices have gained widespread acceptance throughout the electronics industry. Improvements in materials and process controls have resulted in significant improvements in reliability performance. In addition, plastic packages have the advantage of low cost and physical strength.

Encapsulated integrated circuits incorporate the simpliest processing and package construction of the various systems available. The die is attached to a leadframe, wire bonded and encapsulated using an epoxy novolac molding compound. The die may be attached to the leadframe by epoxy or by any of a variety of eutectic forming metal preforms. Wire bonding in plastic may be thermocompression or thermosonic, but the wire is always gold. The encapsulant is the most critical component of the system since it controls contamination, moisture resistance, and stress effects. Epoxy novolacs have become the industry standard molding compound since they combine excellent characteristics in all these areas.

The plastic package is, by far, the most resistant to physical damage since the die is completely encapsulated and cavity hermeticity is not a concern. Since the package is light in weight and the plastic is less brittle than ceramic, chipping and cosmetic damage are not problems. The leadframe and plating are equivalent to CERDIP.

In comparing plastic to ceramic packages, there are two characteristics to be considered: moisture resistance and thermal characteristics. Microprocessor plastic products perform very well on

moisture resistance related tests. This is due to advances in molding compounds, the characteristic low voltages and the moderate power dissipation of Microprocessor products. In most instances plastic devices will provide excellent performance, essentially equivalent to hermetic performance. Thermal resistance has been improved dramatically through the introduction of copper leadframes, and this results in lower junction temperatures, and subsequent improvements in electrical characteristics and reliability performance.

Many users of integrated circuits continue to have requirements or preferences for hermetically sealed ceramic packages. These requirements are usually based on applications in a highly humid environment, increased temperature range or high power dissipation. Motorola produces two different types of ceramic packaged devices: CERDIP and sidebraze.

The sidebraze, or solder seal, package is composed of three layers of alumina which are screened with refractory metal such as tungsten or moly manganese and fired together to form the package body with a cavity for the die. The refractory metal is then plated and Alloy 42 leadframes are brazed to the bottom, sides or top of the package, depending on the vendor. The advantage of the sidebraze version is accurate lead alignment without the need for forming. The final piece part operation is plating, which may be gold or tin with a selective gold plate in the cavity. Although epoxy die bonding is feasible in this package — due to the higher sealing temperature, most manufacturers, including Motorola, employ a eutectic bond. Both aluminum ultrasonic wire bonding and gold thermocompression bonding are used in this package.

The cerdip package is composed of two ceramic piece parts: the base and the cap. Sandwiched between these two layers is a leadframe composed of Alloy 42 imbeded in a sealing glass. the leadframe requires a forming operation similar to a plastic dip. The die is mounted in this package using a eutectic bond while the wire bonds are aluminum (ultrasonic). A tin plate is applied to the exterior leads of the package.

Some tradeoffs exist in the performance characteristics of the two hermetic packages as they are offered by Motorola. Both typically are ceramic, hermetic, employ a eutectic die bond, use ultrasonic aluminum wire bonding, and have tin plating on the exterior leads. The thermal resistance of the packages is very similar, with the sidebraze having a slight advantage. Both packages perform well on the standard thermal and mechanical environmental tests, but each is susceptible to handling damage. Loose shipping rail packaging or high velocity impacts during testing can chip the sidebraze package and sever the interlayer metallization. This type of handling will not effect the 10 mil thick leadframe of the CERDIP package, but hermeticity failures can occur. The CERDIP package is slightly thicker and heavier, but no conductive surfaces are exposed so the shorting potential in dense packaging is reduced. Extensive testing of 24, 28, and 40 lead CERDIP and sidebraze devices has indicated no significant differences in reliability.

RELIABILITY TEST

The following paragraphs describe the various reliability test included in Motorola's Reliability and Quality Assurance Program.

High Temperature Operating Life Test

High temperature operating life (HTOL) testing is performed to accelerate failure mechanisms which are thermally activated through the application of extreme temperatures and the use of dynamic operating conditions. The temperature and voltage conditions used in the stress are typically 125°C with the bias level at the maximum data sheet specification limit of 5.5 volts. All

devices used in the HTOL test are sampled directly after final electrical test with no prior burn-in or other special screening. Testing is performed with dynamic signals applied to the device for a minimum test duration of 1008 hours.

Device equivalent hours assume the Arrhenius relationship using an activation energy of 0.7 eV to extrapolate from the device junction temperature at 125°C (ambient) to the junction temperature at 70°C (ambient). Failure rates given in Failure in Time (FIT) are derived using the Chi-Square distribution to a 90% confidence limit. A FIT is one failure per billion device hours of 0.0001%/ 1000 hours.

Tables 2-1 through 2-3 show the results for the high temperature operating life test for packaging; plastic, plastic leaded chip carrier (PLCC), and ceramic. Each of these tables also lists the different technology used in the test. Table 2-4 lists the grand totals of Table 2-1 through 2-3 by technology and packaging. Figure 2-1 shows a trend chart of the high temperature operating life by technology, and Figure 2-2 is a trend chart of the total of the high temperature operating life test.

Table 2-1. High Temperature Operating Life Test
PLASTIC

STRESS VOLTAGE: 5.5 Volts
TEMPERATURE: 125°C
LONGEST STRESS: 1008 Hours

Device Type	Test Devices	125°C Device Hrs.	70°C Equiv. Device Hrs.[1]	Failures	FITS[2] 0.7 eV
NMOS DIP					
MC3870	45	45,400	6.49×10^5	1	5970
MC6800	135	136,000	1.42×10^6	0	1606
MC6802	378	380,000	5.41×10^6	0	421
MCM6810	45	45,400	4.13×10^5	0	5521
MC6840	135	136,000	1.37×10^6	0	1664
MC6844	45	45,400	5.00×10^5	0	4561
MC6845	45	45,400	4.09×10^5	0	5575
MC6846	45	45,400	4.57×10^5	0	4990
MC6850	135	136,000	1.63×10^6	0	1399
MC6852	45	45,400	7.22×10^5	0	3158
MC6854	45	45,400	3.87×10^5	0	5892
MC68661	135	136,000	1.32×10^6	0	1727
MC68652	45	45,400	3.90×10^5	0	5847
MC68901	45	45,400	8.10×10^5	0	2815
CUSTOM A	945	951,000	2.11×10^7	2	252
CUSTOM B	525	530,000	7.54×10^6	0	302
CUSTOM C	1834	1,820,000	3.76×10^7	1	103
TOTAL	**4627**	**4,633,600**	$\mathbf{8.21 \times 10^7}$	**4**	**97**
CMOS DIP					
MC146805E2	45	43,000	1.08×10^6	0	2111
MC146805F2	225	224,000	5.71×10^6	1	679
MC146805G2	89	74,200	1.88×10^6	0	1213
MC146818	45	45,400	1.17×10^6	0	1949
MC146818A	90	90,800	2.34×10^6	0	974
MC146823	340	342,000	8.82×10^6	0	259
TOTAL	**834**	**819,400**	$\mathbf{2.10 \times 10^7}$	**1**	**185**

1) Activation energy used in equivalent device hour calculation is 0.7 ev)
2) 90% confidence.

Table 2-1. High Temperature Operating Life Test
PLASTIC (Continued)

STRESS VOLTAGE: 5.5 Volts
TEMPERATURE: 125°C
LONGEST STRESS: 1008 Hours

Device Type	Test Devices	125°C Device Hrs.	70°C Equiv. Device Hrs.[1]	Failures	FITS[2] 0.7 eV
HMOS DIP					
MC2674	45	45,000	6.30×10^5	0	3619
MC2681	90	90,800	1.25×10^6	0	1824
MC6801	135	136,000	1.42×10^6	0	1606
MC6801U4	412	364,000	5.09×10^6	0	448
MC6803U4	45	43,400	5.95×10^5	0	3832
MC6804J2	167	163,000	3.16×10^6	0	722
MC6804P2	90	88,800	1.82×10^6	0	1253
MC6805P2	180	182,000	2.67×10^6	0	854
MC6805P4	43	41,700	6.13×10^5	1	6321
MC6805R2	45	41,200	4.36×10^5	0	5320
MC6805R3	180	182,000	1.92×10^6	0	1188
MC6805S2	45	45,400	7.20×10^5	0	3167
MC6805S3	45	43,900	6.99×10^5	1	5543
MC6805T2	135	122,000	1.79×10^6	1	2165
MC6809	90	90,800	1.37×10^6	1	2828
MC6809E	135	136,000	2.06×10^6	0	1107
MC68000	504	468,000	7.28×10^6	0	313
MC68008	45	45,400	5.38×10^5	0	4238
MC68010	45	45,400	6.50×10^5	0	3508
MC68230	45	45,400	8.90×10^5	0	2562
MC68681	43	41,000	6.50×10^5	0	3508
TOTAL	**2669**	**2,567,200**	$\mathbf{3.78 \times 10^7}$	**4**	**211**
HCMOS DIP					
MC68HC05C4	410	389,000	9.94×10^6	1	390
MC68HC05C8	89	89,800	2.29×10^6	0	996
XC68HC000	134	134,000	3.20×10^6	1	1211
TOTAL	**633**	**612,800**	$\mathbf{1.55 \times 10^7}$	**2**	**343**
PLASTIC DIP	**8763**	**8,633,200**	$\mathbf{1.56 \times 10^8}$	**11**	**106**

1) Activation energy used in equivalent device hour calculation is 0.7 ev)
2) 90% confidence.

Table 2-2. High Temperature Operating Life Test PLCC

STRESS VOLTAGE: 5.5 Volts
TEMPERATURE: 125°C
LONGEST STRESS: 1008 Hours

Device Type	Test Devices	125°C Device Hrs.	70°C Equiv. Device Hrs.[1]	Failures	FITS[2] 0.7 eV
HMOS PLCC					
MC6805R2	89	81,900	8.34×10^5	0	2734
MC6805R3	450	430,000	4.38×10^6	0	512
CUSTOM D	280	279,000	5.14×10^6	0	444
CUSTOM E	1189	1,200,000	2.20×10^7	0	104
TOTAL	**2008**	**1,990,900**	$\mathbf{3.24 \times 10^7}$	**0**	**70**
HCMOS PLCC					
MC68HC11	2155	2,160,000	5.66×10^7	14	356
CUSTOM E	416	417,000	7.56×10^6	1	513
CUSTOM G	1358	1,360,000	2.46×10^7	2	216
TOTAL	**3929**	**3,937,000**	$\mathbf{8.87 \times 10^7}$	**17**	**266**
CMOS PLCC					
MC146805F2	45	45,400	1.17×10^6	0	1949
MC146805G2	90	89,000	2.26×10^6	0	1009
MC146818	45	45,400	1.17×10^6	0	1949
MC146818A	89	88,900	2.26×10^6	0	1009
TOTAL	**269**	**268,700**	$\mathbf{6.83 \times 10^6}$	**0**	**334**
PLCC	**6206**	**6,196,600**	$\mathbf{1.28 \times 10^8}$	**17**	**184**

1) Activation energy used in equivalent device hour calculation is 0.7 eV.
2) 90% confidence.

Table 2-3. High Temperature Operating Life Test CERAMIC

STRESS VOLTAGE: 5.5 Volts
TEMPERATURE: 125°C
LONGEST STRESS: 1008 Hours

Device Type	Test Devices	125°C Device Hrs.	70°C Equiv. Device Hrs.[1]	Failures	FITS[2] 0.7 eV
NMOS					
MC6821L	44	44,300	6.02×10^5	0	3788
MC6821S	45	45,400	6.18×10^5	0	3690
MC6844S	135	136,000	1.96×10^6	0	1163
MC6850	90	90,800	1.39×10^6	0	1640
MC6850S	45	45,400	6.95×10^5	0	3281
MC6852S	45	44,000	7.96×10^5	0	2865
TOTAL	**404**	**405,900**	$\mathbf{6.06 \times 10^6}$	**0**	**376**
HCMOS					
MC68020R	542	541,000	9.59×10^6	2	554
MC68605R	77	77,000	1.80×10^6	0	1267
MC68824R	135	136,000	3.00×10^6	0	760
MC68851R	144	145,000	3.19×10^6	1	1215
MC68882R	604	597,000	1.40×10^7	2	379
TOTAL	230	230,000	5.30×10^6	0	430
	1732	**1,726,000**	$\mathbf{3.69 \times 10^7}$	**5**	**251**

1) Activation energy used in equivalent device hour calculation is 0.7 eV.
2) 90% confidence.

Table 2-3. High Temperature Operating Life Test
CERAMIC (Continued)

STRESS VOLTAGE: 5.5 Volts
TEMPERATURE: 125°C
LONGEST STRESS: 1008 Hours

Device Type	Test Devices	125°C Device Hrs.	70°C Equiv. Device Hrs.[1]	Failures	FITS[2] 0.7 eV
CMOS					
MC1468705F2	45	34,400	8.57×10^6	0	2661
MC1468705F2S	90	908,000	2.28×10^6	0	1000
TOTAL	**135**	**942,000**	$\mathbf{3.14 \times 10^6}$	**0**	**726**
HMOS					
MC6803L	45	45,400	6.40×10^5	0	3563
MC6809EL	45	45,400	7.07×10^5	0	3225
MC6809ES	45	45,400	6.25×10^5	0	3648
MC6809S	90	90,800	1.25×10^6	1	3100
MC68701S	45	45,400	4.74×10^5	0	4811
MC68701U4L	45	45,400	6.22×10^5	0	3666
XC68704P2S	300	301,000	6.62×10^6	1	585
MC68705P3	45	45,400	7.00×10^5	0	3258
MC68705S3	45	45,400	4.78×10^5	0	4770
MC68000L	170	171,000	2.53×10^6	0	901
MC68000R	248	245,000	3.93×10^6	0	580
MC68010L	94	94,000	1.52×10^6	0	1500
MC68010R	45	45,400	7.30×10^5	0	3124
MC68230L	90	90,800	1.74×10^6	1	2227
TOTAL	**1352**	**1,355,800**	$\mathbf{2.28 \times 10^7}$	**3**	**293**
CERAMIC	**3623**	**4,433,700**	$\mathbf{6.89 \times 10^7}$	**8**	**189**

1) Activation energy used in equivalent device hour calculation is 0.7 eV.
2) 90% confidence.

Table 2-4. High Temperature Operating Life Test
TECHNOLOGY and PACKAGING

STRESS VOLTAGE: 5.5 Volts
TEMPERATURE: 125°C
LONGEST STRESS: 1008 Hours

Device Type	Test Devices	125°C Device Hrs.	70°C Equiv. Device Hrs.[1]	Failures	FITS[2] 0.7 eV
NMOS	5,031	5,039,500	8.82×10^7	4	91
HMOS	6,029	5,913,900	9.30×10^7	7	127
CMOS	1,238	2,030,100	3.10×10^7	1	125
HCMOS	6,294	6,275,000	1.41×10^8	24	224
DIP	(8,763)	(8,633,000)	1.56×10^8	11	106
PLCC	(6,206)	(6,196,600)	1.28×10^8	17	184
PLASTIC	([14,969])	([14,829,800])	2.84×10^8	28	127
CERAMIC	[3,623]	[4,433,700]	6.89×10^8	8	189
GRAND TOTAL	**18,592**	**19,259,300**	$\mathbf{3.53 \times 10^8}$	**36**	**117**

1) Activation energy used in equivalent device hour calculation is 0.7 eV.
2) 90% confidence.

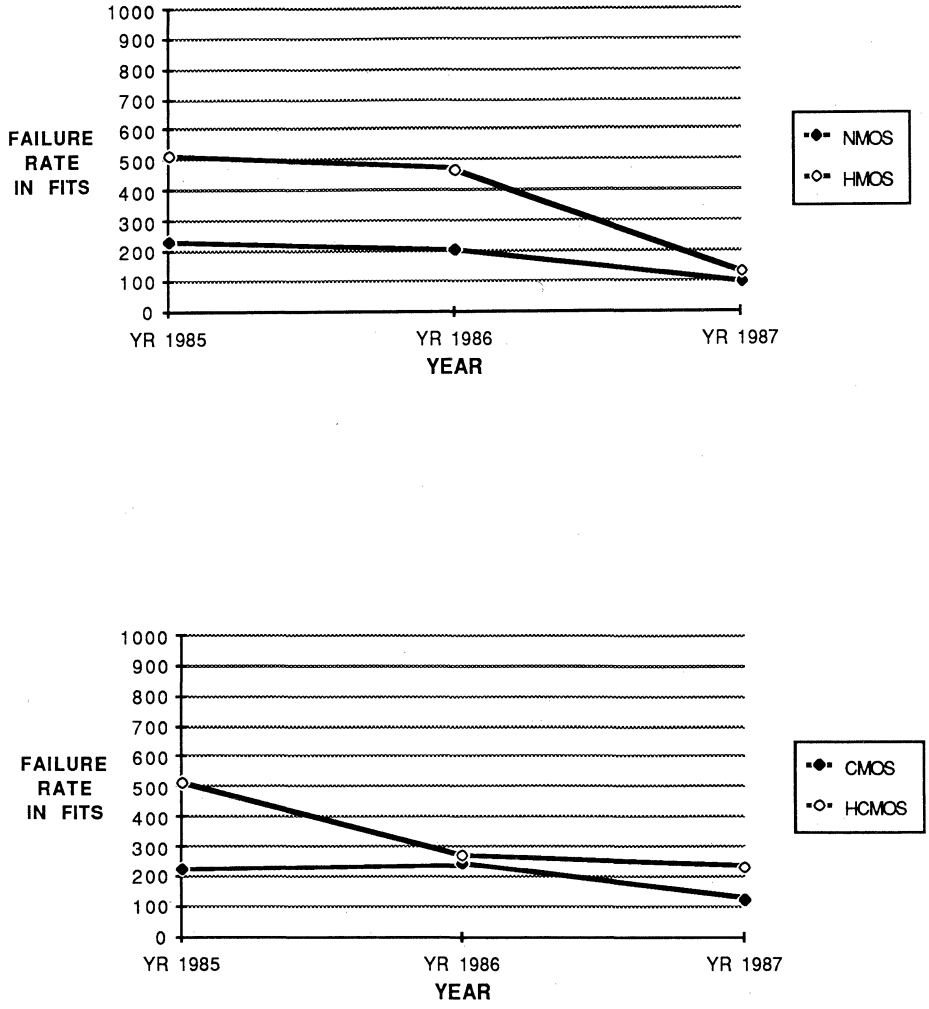


Figure 2-1. High Temperature Operating Life Trend Chart (By Technology)

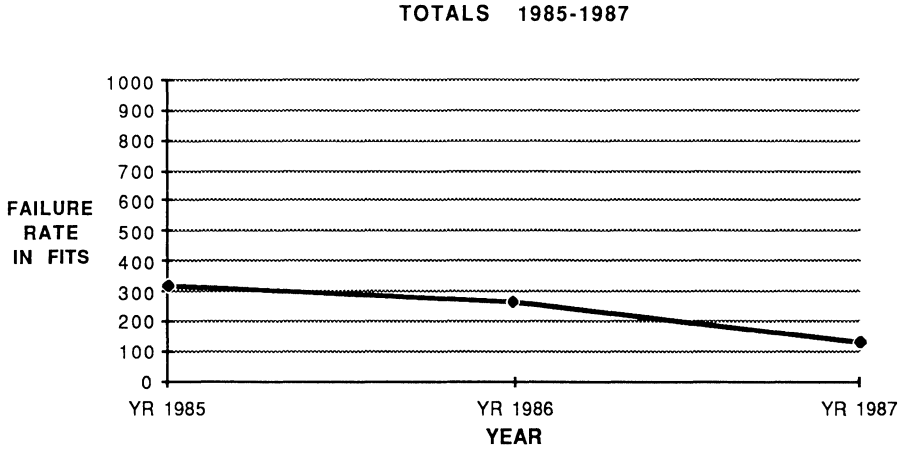


Figure 2-2. High Temperature Operating Life Trend Chart (Total)

Temperature Humidity Bias Test

Temperature humidity bias (THB) is an environmental test performed at a temperature of 85°C and a relative humidity of 85%. The test is designed to measure the moisture reistance of plastic encapsulated circuits. A nominal voltage of 5 volts static bias is applied to the device to create the electrolytic cells necessary to accelerate corrosion of the metallization. Testing is performed to JEDEC Standard 22, Method A101. Most groups are tested to 1008 hours with some groups extended beyond to look for longer term effects.

Table 2-5 shows the results of the temperature humidity bias test. Table 2-6 lists the grand total of the devices tested in Table 2-5. Figure 2-3 shows the trend chart for the temperature humidity bias test.

Table 2-5. Temperature Humidity Bias Test

TEMPERATURE: 85°C
HUMIDITY: 85%
LONGEST STRESS: 1008 Hours

Device Type	— Failures Per Sample —			
	168 Hrs	504 Hrs	1008 Hrs	% Failures
NMOS DIP				
MC6800	0/68	0/68	0/68	0.00
MC6802	0/34	1/34	0/33	2.94
MC6840	0/68	0/68	0/68	0.00
MC6845	0/34	0/34	0/34	0.00
MC6850	0/34	0/34	0/34	0.00
MC6852	0/34	0/34	0/34	0.00
TOTAL	**0/272**	**1/272**	**0/271**	**0.37**
HMOS DIP				
MC6801	0/68	0/67	0/66	0.00
MC6802	0/34	0/34	0/34	0.00
MC6803U4	0/34	0/34	0/34	0.00
MC6804J2	0/222	0/222	0/222	0.00
MC6805P2	0/34	0/34	0/34	0.00
MC6805P4	0/34	2/34	0/32	5.88
MC6805P3	0/68	0/67	0/67	0.00
MC6805T2	0/68	0/65	0/65	0.00
MC6809	0/34	0/34	0/34	0.00
MC6809E	0/102	0/102	0/102	0.00
MC68000	0/136	0/134	0/134	0.00
MC68008	0/94	0/94	1/93	1.08
MC68010	0/34	0/34	0/29	0.00
TOTAL	**0/962**	**2/955**	**1/946**	**0.32**
HMOS PLCC				
MC68000	0/45	0.45	0/45	0.00
MC68705R3	0/254	0.254	0/254	0.00
TOTAL	**0/299**	**0/299**	**0/299**	**0.00**
CMOS DIP				
MC146804E2	0/68	0/68	0/68	0.00
MC146805F2	0/34	0/34	0/34	0.00
MC146805G2	0/68	0/68	0/68	0.00
MC146818A	0/34	0/34	0/34	0.00
MC146823	0/34	0/34	0/34	0.00
TOTAL	**0/238**	**0/238**	**0/238**	**0.00**
HCMOS DIP				
MC68HC05C4	0/102	0/102	0/102	0.00
TOTAL	**0/102**	**0/102**	**0/102**	**0.00**
HCMOS PLCC				
XC68HC11A8	0/231	0/231	0/231	0.00
MC68HC11	0/615	0/615	0/615	0.00
MC68HC000	0/135	0/135	0/135	0.00
MC68605	0/231	1/231	0/230	0.43
TOTAL	**0/1212**	**1/1212**	**0/1211**	**0.083**

Table 2-6. Temperature Humidity Bias Test
GRAND TOTAL

TEMPERATURE: 85°C
HUMIDITY: 85%
LONGEST STRESS: 1008 Hours

Device Type	— Failures Per Sample —			
	168 Hrs	504 Hrs	1008 Hrs	% Failures
NMOS	0/272	1/272	0/272	0.37
HMOS	0/1261	2/1254	1/1245	0.24
HCMOS	0/1314	1/1314	0/1313	0.08
CMOS	0/238	0/238	0/238	0.00
DIP	0/1574	3/1567	1/1558	0.26
PLCC	0/1511	1/1511	0/1510	0.07
GRAND TOTAL	**0/3085**	**4/3078**	**1/3068**	**0.163**

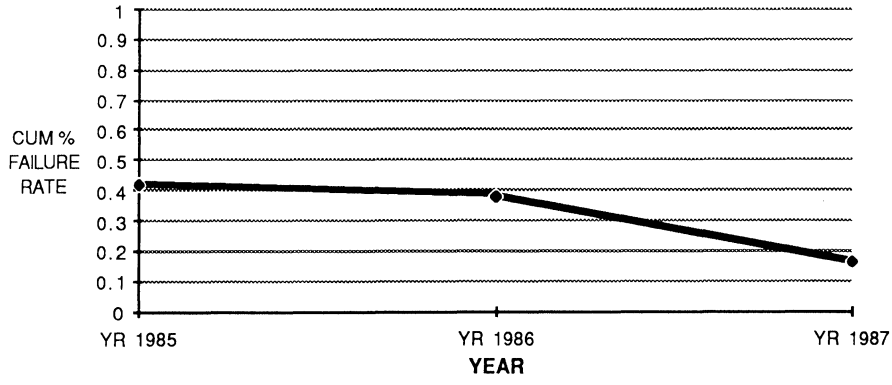


Figure 2-3. Temperature Humidity Bias Trend Chart

Autoclave Test

Autoclave, like THB, is an environmental test which measures device resistance to moisture penetration along the leadframe-plastic interface. Conditions employed during the test include 121°C, 100% relative humidity, and 15 psig. Corrosion of the die is the expected failure mechanism. Autoclave is a highly accelerated and destructive test performed per JEDEC Standard 22, Method A102. Testing is routinely performed for 144 hours.

Table 2-7 lists the results of the autoclave test. Table 2-8 lists the grand total of the devices tested in Table 2-7. Figure 2-4 shows the trend chart for the autoclave test.

Table 2-7. Autoclave Test

TEMPERATURE: 121°C
PRESSURE: 15 psig
LONGEST STRESS: 144 Hours

Device Type	— Failures Per Sample —			
	48 Hrs	96 Hrs	144 Hrs	% Failures
NMOS DIP				
MC6800	0/44	0/44	0/44	0.00
MC6802	0/22	0/22	0/22	0.00
MC6840	0/44	0/44	0/44	0.00
MC6845	0/22	0/22	0/22	0.00
MC6846	0/22	0/22	0/22	0.00
MC6850	0/65	0/65	0/65	0.00
MC6852	0/22	0/22	0/22	0.00
MC6854	0/44	0/44	0/44	0.00
MC68661	0/22	0/22	0/22	0.00
MC68901	0/22	0/22	0/22	0.00
CUSTOM A	0/330	0/327	0/326	0.00
CUSTOM B	0/462	0/462	0/462	0.00
CUSTOM C	0/847	2/846	0/843	0.24
TOTAL	**0/1968**	**2/1964**	**0/1960**	**0.10**
HCMOS DIP				
MC68HC21	0/22	0/22	0/22	0.00
MC68HC05C4	0/114	0/114	0/114	0.00
TOTAL	**0/136**	**0/136**	**0/136**	**0.00**
CMOS DIP				
MC146805F2	0/65	0/65	0/65	0.00
MC146805G2	0/44	0/44	0/44	0.00
MC146818	0/192	0/192	0/192	0.00
MC146818A	0/394	0/394	0/394	0.00
MC146805E2	0/22	0/22	0/22	0.00
MC146805F2	0/43	0/43	0/43	0.00
MC146805G2	0/66	0/66	0/66	0.00
MC146823	0/534	0/534	0/534	0.00
MC1468705F2	0/34	0/34	0/34	0.00
TOTAL	**0/1394**	**0/1394**	**0/1394**	**0.00**
HMOS DIP				
MC2674	0/22	0/22	0/22	0.00
MC6801	0/297	0/297	0/297	0.00
MC6801U4	1/777	0/776	1/776	0.13
MC6802	0/22	0/22	0/22	0.00
MC6803U4	0/44	0/44	0/44	0.00
MC6804J2	0/354	0/354	0/354	0.00
MC6804P2	0/66	0/66	0/66	0.00
MC6805P2	0/44	1/44	0/43	2.32
MC6805R2	0/34	0/34	0/34	0.00
MC6805R3	0/98	0/98	0/98	0.00
MC6805S2	0/44	0/44	0/44	0.00
MC6805S3	0/22	0/22	0/22	0.00
MC6805T2	0/66	0/66	0/66	0.00
MC6809	0/44	0/43	0/43	0.00
MC6809E	0/44	0/44	0/44	0.00
MC68000	0/110	0/110	0/110	0.00
MC68008	0/22	0/22	0/22	0.00
MC68010	0/22	0/22	0/22	0.00
MC68230	0/22	0/22	0/22	0.00
MC68661	0/44	0/44	0/44	0.00
MC68681	0/65	0/64	0/64	0.00
TOTAL	**1/2263**	**1/2260**	**1/2259**	**0.13**

Table 2-7. Autoclave Test (Continued)

TEMPERATURE: 121°C
PRESSURE: 15 psig
LONGEST STRESS: 144 Hours

Device Type	— Failures Per Sample —			
	48 Hrs	96 Hrs	144 Hrs	% Failures
HMOS PLCC				
MC6805R2	0/170	0/170	0/170	0.00
MC68705R3	0/255	0/255	0/255	0.00
CUSTOM D	0/434	0/433	0/432	0.00
CUSTOM E	1/693	0/692	0/691	0.14
TOTAL	**1/1552**	**0/1550**	**0/1548**	**0.06**
HCMOS PLCC				
MC68HC11	1/1150	0/1148	2/1146	0.26
MC68HC11A8	0/320	0/320	0/320	0.00
MC68HC000	0/135	0/135	0/135	0.00
MC68605	0/45	0/45	0/44	0.00
CUSTOM E	1/363	1/361	0/359	0.55
CUSTOM G	0/770	0/770	0/770	0.00
TOTAL	**2/2783**	**1/2779**	**2/2775**	**0.18**
CMOS PLCC				
MC146805F2	0/102	0/102	0/102	0.00
MC146805G2	0/68	0/68	0/68	0.00
MC146818	0/102	0/101	0/101	0.00
MC146818A	0/33	0/33	0/33	0.00
TOTAL	**0/305**	**0/304**	**0/304**	**0.00**

Table 2-8. Autoclave Test GRAND TOTAL

TEMPERATURE: 121°C
PRESSURE: 15 psig
LONGEST STRESS: 144 Hours

Device Type	— Failures Per Sample —			
	48 Hrs	96 Hrs	144 Hrs	% Failures
NMOS	0/1968	2/1963	0/1959	0.10
HMOS	2/3815	1/3810	1/3807	0.10
HCMOS	2/2919	1/2915	2/2909	0.17
CMOS	0/1699	0/1698	0/1698	0.00
DIP	1/5761	3/5753	1/5748	0.09
PLCC	3/4640	1/4633	2/4625	0.13
GRAND TOTAL	**4/10401**	**4/10386**	**3/10373**	**0.106**

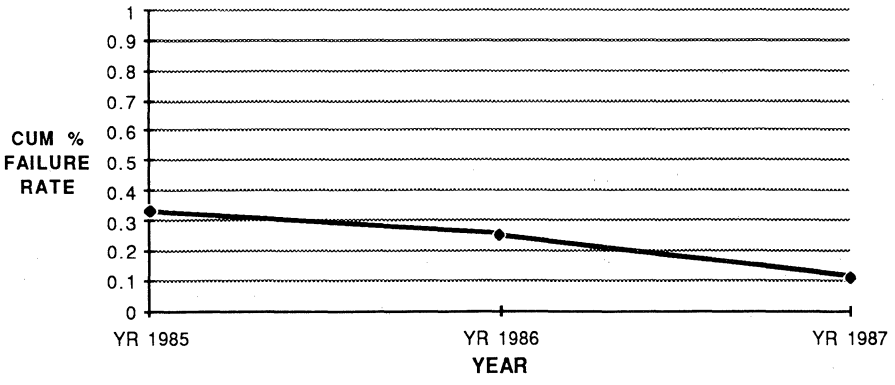


Figure 2-4. Autoclave Trend Chart

Temperature Cycle Test

Temperature cycle testing accelerates the effects of thermal expansion mismatch among the different components within a specific packaging system. During temperature cycle testing, devices are inserted into a cycling system and held at the cold (−65°C) dwell temperature for at least ten minutes. Following this cold dwell, the devices are heated to the hot (+105°C) dwell where they remain for another ten minute minimum time period. The system employs a circulating air environment to assure rapid stabilization at the specified temperature. The dwell at each extreme, plus the two transition times of five minutes each (one up to the hot dwell temperature, another down to the cold dwell temperature), constitutes one cycle. Test duration is for 1000 cycles with some tests extended to look for longer term effects.

Table 2-9 lists the test results of the temperature cycle test testing at a temperature range of a −65°C to 150°C. Table 2-10 lists the grand total of the devices tested in Table 2-9. Table 2-11 lists the test results of the temperature cycle test testing at a temperature range of a −50°C to 150°C. Table 2-12 lists the grand total of the devices and results of Table 2-11. Figure 2-5 shows the trend chart for the temperature cycle test.

Table 2-9. Temperature Cycle Test

TEMPERATURE: −65°C to +150°C
STRESS METHOD: Air to Air
LONGEST STRESS: 1000 Cycles

Device Type	— Failures Per Sample —			
	100 cyc	500 cyc	1K cyc	% Failures
NMOS DIP				
MC3870	0/38	0/38	0/38	0.00
MC6800	0/114	0/114	0/114	0.00
MCM6810	0/38	0/37	0/37	0.00
MC6840	0/114	0/114	0/114	0.00
MC6844	0/38	0/38	0/38	0.00
MC6845	0/38	0/38	1/38	2.63
MC6846	0/38	0/38	0/38	0.00
MC6850	0/114	0/114	1/114	0.88
MC6852	0/76	0/76	0/76	0.00
MC6854	0/38	0/38	0/38	0.00
MC68661	0/38	0/38	0/38	0.00
CUSTOM A	0/307	0/307	0/307	0.00
CUSTOM B	0/115	0/115	0/115	0.00
CUSTOM C	0/1306	0/1306	3/1306	0.23
TOTAL	**0/2412**	**0/2411**	**5/2411**	**0.21**
NMOS CERAMIC				
MC6821S	0/38	0/38	0/38	0.00
MC6821L	0/38	0/38	0/38	0.00
MC6844S	0/114	0/114	0/114	0.00
MC6850	0/114	0/114	0/114	0.00
TOTAL	**0/304**	**0/304**	**0/304**	**0.00**
HMOS PLCC				
MC6805R2	0/152	3/152	4/149	4.61
MC6805R3	0/38	0/38	0/38	0.00
CUSTOM E	0/153	1/153	0/152	0.65
TOTAL	**0/343**	**4/343**	**4/339**	**2.36**
HMOS DIP				
MC2674	0/37	0/37	0/37	0.00
MC6801	0/546	1/546	2/542	0.55
MC6801U4	0/668	0/653	0/638	0.00
MC6802	0/38	0/38	0/38	0.00
MC6803U4	0/114	0/114	0/114	0.00
MC6804J2	0/76	0/76	0/76	0.00
MC6804P2	0/114	0/114	0/114	0.00
MC6805P2	0/76	0/76	0/75	0.00
MC6805P4	0/38	0/38	0/38	0.00
MC6805P6	1/432	2/429	0/427	0.69
MC6805S2	1/38	0/37	0/37	2.63
MC6805R2	0/38	0/38	0/38	0.00
MC6805R3	0/152	0/152	3/152	1.97
MC6805S2	0/38	0/38	0/38	0.00
MC6805S3	0/37	0/37	0/37	0.00
MC6805T2	0/114	0/114	0/114	0.00
MC6809E	1/114	0/113	0/113	0.88
MC6809	0/76	0/76	0/76	0.00
TOTAL	**3/2746**	**3/2726**	**5/2704**	**0.40**

Table 2-9. Temperature Cycle Test (Continued)

TEMPERATURE: −65°C to +150°C
STRESS METHOD: Air to Air
LONGEST STRESS: 1000 Cycles

Device Type	— Failures Per Sample —			
	100 cyc	500 cyc	1K cyc	% Failures
HMOS CERAMIC				
MC6801L	0/38	0/38	0/38	0.00
MC6809EL	0/38	0/38	0/38	0.00
MC6809ES	0/38	0/38	3/38	7.89
MC6809S	0/76	0/76	2/76	2.63
MC68120L	0/76	0/76	0/76	0.00
MC68701	0/38	0/38	0/38	0.00
MC68701U4L	0/38	0/37	0/37	0.00
MC68705P3	0/38	0/38	1/38	2.63
MC68705S3S	0/38	0/38	0/38	0.00
MC68000L	0/113	0/112	0/112	0.00
MC68000R	0/38	0/38	0/37	0.00
MC68010R	0/38	0/38	0/38	0.00
MC68230L	0/38	0/38	0/38	0.00
MC68451L	0/36	0/36	0/36	0.00
MC68901L	0/38	0/38	0/38	0.00
TOTAL	**0/719**	**0/717**	**6/716**	**0.84**
HCMOS PLCC				
MC68HC11	1/538	0/537	2/535	0.56
MC68881	0/78	0/78	0/78	0.00
XC68882	1/135	0/134	0/134	0.74
CUSTOM E	0/253	4/252	5/246	3.56
CUSTOM G	0/1153	1/1153	2/1150	0.26
TOTAL	**2/2157**	**5/2154**	**9/2143**	**0.74**
HCMOS DIP				
MC68HC05C4	0/223	0/223	0/223	0.00
MC68HC05C8	0/338	0/338	3/338	0.89
MC68HC21	0/109	0/109	0/109	0.00
TOTAL	**0/670**	**0/670**	**3/670**	**0.45**
HCMOS CERAMIC				
MC68020R	0/77	0/77	0/77	0.00
MC68605R	0/231	0/231	0/231	0.00
MC68824R	0/231	0/231	1/230	0.43
MC68851R	0/77	0/77	0/77	0.00
MC68881R	0/74	0/74	0/74	0.00
MC68704P2	0/135	0/135	0/135	0.00
TOTAL	**0/825**	**0/825**	**1/824**	**0.12**
CMOS PLCC				
MC146805F2	0/76	0/76	0/76	0.00
MC146805G2	0/114	0/114	0/113	0.00
MC146818	0/38	0/38	0/38	0.00
MC146818A	0/76	0/76	0/76	0.00
TOTAL	**0/304**	**0/304**	**0/303**	**0.00**
CMOS DIP				
MC146805E2	0/38	0/38	0/38	0.00
MC146805F2	0/190	0/189	0/189	0.00
MC146805G2	0/152	0/152	0/152	0.00
MC146818	0/38	0/38	0/38	0.00
MC146818A	0/76	0/76	0/76	0.00
MC146823	0/76	0/76	0/76	0.00
MC1468705F2	0/38	0/38	0/38	0.00
TOTAL	**0/608**	**0/607**	**0/607**	**0.00**

Table 2-10. Temperature Cycle Test
GRAND TOTAL

TEMPERATURE: −65°C to +150°C
STRESS METHOD: Air to Air
LONGEST STRESS: 1000 Cycles

Device Type	— Failures Per Sample —			
	100 cyc	500 cyc	1K cyc	% Failures
NMOS	0/2716	0/2715	5/2715	0.18
HMOS	3/3808	7/3786	15/3759	0.67
CMOS	0/912	0/911	0/910	0.00
HCMOS	2/3652	5/3649	10/3637	0.47
PLCC	2/2804	9/2801	13/2785	0.86
DIP	3/6436	3/6414	10/6392	0.25
PLASTIC	5/9240	12/9215	23/9177	0.44
CERMAIC	0/1848	0/1846	7/1844	0.38
GRAND TOTAL	**5/11088**	**12/11061**	**30/11021**	**0.43**

Table 2-11. Temperature Cycle Test

TEMPERATURE: −50°C to +150°C
STRESS METHOD: Air to Air
LONGEST STRESS: 1000 Cycles

Device Type	— Failures Per Sample —			
	100 cyc	500 cyc	1K cyc	% Failures
NMOS DIP				
MC68661	0/114	0/114	0/114	0.00
MC68901	0/38	0/38	0/38	0.00
TOTAL	**0/152**	**0/152**	**0/152**	**0.00**
HMOS DIP				
MC6801	0/231	0/231	0/231	0.00
MC6801U4	0/77	0/77	0/77	0.00
MC68000	0/114	0/114	0/107	0.00
MC68008	0/38	0/38	1/38	2.63
MC68230	0/38	0/38	1/38	2.63
MC68681	0/112	0/111	0/111	0.00
TOTAL	**0/610**	**0/609**	**2/602**	**0.33**
HMOS PLCC				
MC68HC11	0/384	0/384	0/384	0.00
MC68705R3	0/107	0/107	0/107	0.00
CUSTOM G	0/737	1/737	0/736	0.14
CUSTOM E	0/154	0/154	0/154	0.00
TOTAL	**0/1382**	**1/1382**	**0/1381**	**0.07**
HMOS PLCC				
MC68HC11A8	0/80	0/80	0/80	0.00
MC68881	1/76	0/75	0/75	1.32
CUSTOM E	2/198	2/195	1/191	2.53
CUSTOM G	0/308	0/308	0/308	0.00
TOTAL	**3/662**	**2/658**	**1/654**	**0.92**

Table 2-12. Temperature Cycle Test
GRAND TOTAL

TEMPERATURE: −50°C to +150°C
STRESS METHOD: Air to Air
LONGEST STRESS: 1000 Cycles

Device Type	— Failures Per Sample —			
	100 cyc	500 cyc	1K cyc	% Failures
NMOS	0/152	0/152	0/152	0.00
HMOS	0/1992	1/1991	2/1983	0.15
HCMOS	3/662	2/658	1/654	0.92
PLCC	3/2044	3/2040	1/2035	0.34
DIP	0/762	0/761	2/754	0.27
GRAND TOTAL	**3/2806**	**3/2801**	**3/2789**	**0.32**

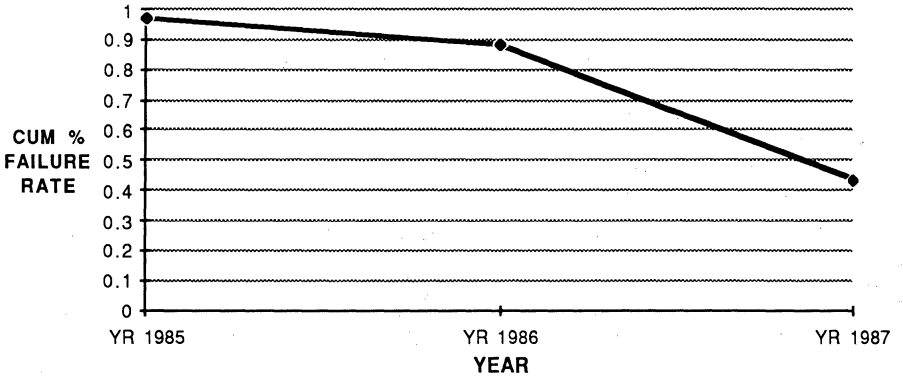


Figure 2-5. Temperature Cycle Trend Chart

Thermal Shock Test

The objective of thermal shock testing is the same as that for temperature cycle testing — to emphasize differences in expansion coefficients for components of the packaging system. However, thermal shock provides additional stress in that the device is exposed to a sudden change in temperature due to the transfer time of ten seconds maximum as well as the increased thermal conductivity of a liquid ambient. Devices are placed in a flourocarbon bath and cooled to −65°C. After being held in the cold chamber for five minutes minimum, the devices are transferred to an adjacent chamber filled with flourocarbon at +150°C for an equivalent time. Two five minute dwells plus two ten second transitions constitute one cycle. Test duration is normally for 1000 cycles with some tests being extended to look for longer term effects.

Table 2-13 lists the results of the thermal shock test and Table 2-14 lists the grand total of Table 2-13. Figure 2-6 shows the trend chart for the thermal chart for the thermal shock test.

Table 2-13. Thermal Shock Test

TEMPERATURE: −65°C to +150°C
STRESS METHOD: Liquid to Liquid
LONGEST STRESS: 1000 Cycles

Device Type	— Failures Per Sample —			
	100 cyc	500 cyc	1K cyc	% Failures
NMOS DIP				
MC2674	0/34	0/34	0/34	0.00
MC3870	0/135	2/135	0/133	1.48
MC6800	0/34	0/34	0/34	0.00
MC6802	0/34	0/34	0/34	0.00
MCM6810	0/34	0/34	0/34	0.00
MC6846	0/34	0/34	0/29	0.00
MC6850	0/68	0/68	0/67	0.00
MC68488	0/34	0/34	0/34	0.00
MC68661	0/68	0/68	1/68	1.47
TOTAL	**0/475**	**2/474**	**1/467**	**0.64**
HMOS DIP				
MC6801	0/432	0/431	0/431	0.00
MC6803U4	0/68	0/68	0/68	0.00
MC6804J2	0/343	0/343	0/343	0.00
MC6804P2	0/34	0/34	0/34	0.00
MC6805P2	0/68	0/68	0/68	0.00
MC6805P4	0/34	0/34	0/34	0.00
MC6805P6	0/432	4/432	0/428	0.93
MC6805R3	0/34	0/34	0/34	0.00
MC6805S2	0/68	0/68	0/68	0.00
MC6805S3	0/34	0/34	2/32	5.88
MC6805T2	0/34	0/34	0/34	0.00
MC68661	0/34	0/34	0/33	0.00
MC68681	0/102	0/102	0/95	0.00
MC68901	0/33	0/34	0/33	0.00
TOTAL	**0/1750**	**4/1750**	**2/1735**	**0.34**
HMOS CERAMIC				
MC6801L	0/38	0/38	0/37	0.00
MC6850	0/38	0/38	0/37	0.00
MC68000L	0/38	0/36	0/36	0.00
MC68010R	0/38	0/38	0/38	0.00
MC68451L	0/38	0/38	0/36	0.00
TOTAL	**0/190**	**0/188**	**0/184**	**0.00**
HCMOS DIP				
XC68HC01	0/34	0/34	0/34	0.00
MC68HC05C8	0/68	0/68	0/68	0.00
TOTAL	**0/102**	**0/102**	**0/102**	**0.00**
HMOS PLCC				
MC68HC11	0/615	2/614	0/612	0.33
HCMOS CERAMIC				
MC68020R	0/462	0/461	1/459	0.22
MC68881R	0/205	0/205	0/199	0.00
TOTAL	**0/667**	**0/666**	**1/658**	**0.15**
CMOS DIP				
MC146805G2	0/34	0/34	0/31	0.00
MC146805F2	0/68	0/68	0/68	0.00
MC146818	0/34	0/34	0/33	0.00
MC146823	0/34	0/34	0/34	0.00
TOTAL	**0/170**	**0/170**	**0/166**	**0.00**

Table 2-14. Thermal Shock Test
GRAND TOTAL

TEMPERATURE: −65°C to +150°C
STRESS METHOD: Liquid to Liquid
LONGEST STRESS: 1000 Cycles

Device Type	— Failures Per Sample —			
	100 cyc	500 cyc	1K cyc	% Failures
NMOS	0/475	2/474	1/467	0.64
HMOS	0/1941	4/1938	2/1919	0.31
CMOS	0/170	0/170	0/166	0.00
HCMOS	0/769	0/768	1/760	0.13
PLCC	0/615	2/614	0/612	0.33
DIP	0/3355	6/3350	4/3312	0.30
PLASTIC	0/3113	8/3110	3/3082	0.35
CERAMIC	0/857	0/854	1/842	0.12
GRAND TOTAL	**0/3970**	**8/3964**	**4/3924**	**0.30**

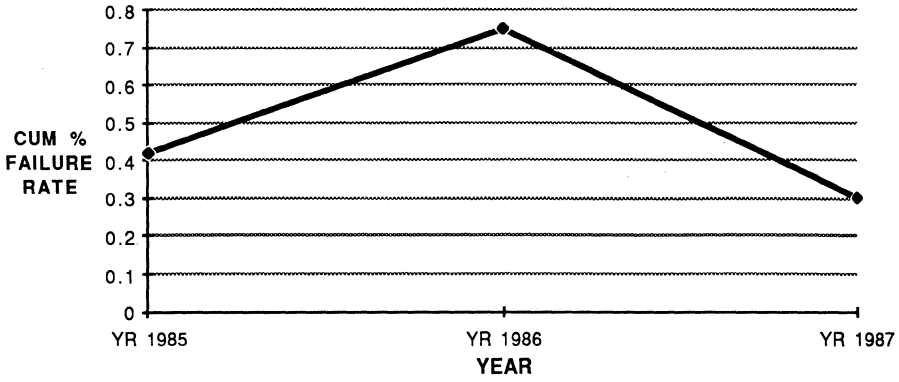


Figure 2-6. Thermal Shock Trend Chart

Data Retention Test

Data retention testing or high temperature storage is performed to measure the stability of the programmed EPROM and EEPROM devices during storage at elevated temperatures with no electrical stress applied. The devices are stored at an ambient of 150°C. An acceleration of charge loss from the storage cell is the expected result. All groups are typically tested to 1008 hours.

Table 2-15 lists the results and the grand total of the data retention test. Figure 2-7 shows the trend chart for the data retention bake test.

Table 2-15. Data Retention Test

TEMPERATURE: 150°C
LONGEST STRESS: 1008 Hours

Device Type	— Failures Per Sample —			
	168 Hrs.	504 Hrs.	1008 Hrs.	% Failures
HMOS CERAMIC				
MC1468705F2	0/78	0/78	0/78	0.00
MC68701U4	0/44	0/44	0/44	0.00
MC68705R3	0/45	0/45	0/45	0.00
MC68704P2	0/442	0/442	0/442	0.00
MC68701	0/45	0/45	0/45	0.00
TOTAL	**0/654**	**0/654**	**0/654**	**0.00**
HMOS DIP				
MC68000	0/100	0/100	0/100	0.00
HMOS PLCC				
MC68705R3	1/2044	0/2043	1/2043	0.10
HCMOS PLCC				
XC68HC11A8*	1/385	0/381	0/377	0.26
MC68HC11A8*	2/1668	2/1667	2/1665	0.36
TOTAL	**3/2053**	**2/2048**	**2/2042**	**0.34**
HMOS	1/2798	0/2797	1/2797	0.07
HCMOS	3/2053	2/2048	2/2042	0.3
				45
GRAND TOTAL	**4/4851**	**2/4845**	**3/4839**	**0.19**

*These EEPROM units were prestressed through 10K write/erase cycles.

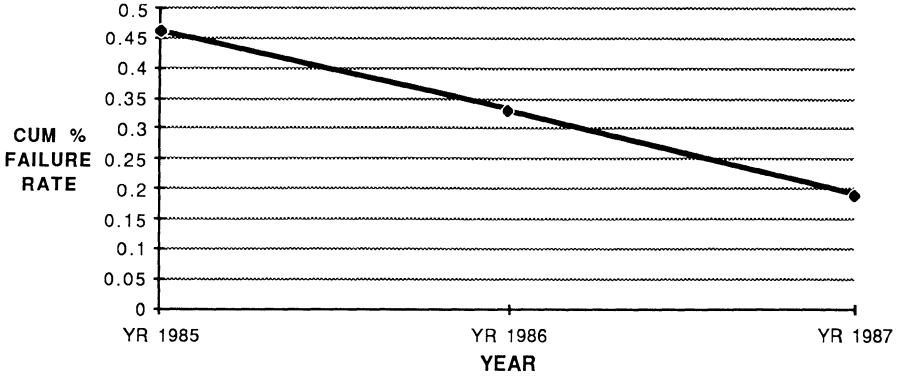


Figure 2-7. Data Retention Bake Trend Chart

EEPROM Write/Erase Cycling Test

The write/erase endurance test measures EEPROM cell operation over an expected life time. All cells are alternately cycled for 10,000 cycles between an erased state "1" and a write state "0" at the device high temperature specification of 85°C. The most common failure mode is failure to write a "0" within the 10 msec specification limit.

Table 2-16 lists the results and grand total of the EEPROM write/erase cycling test. Table 2-17 lists the average outgoing quality from year 1979 through 1987.

Table 2-16. EEPROM Write/Erase Cycling Test

2

VOLTAGE: 5.5 Volts
TEMPERATURE: 85°C
LONGEST STRESS: 10K Cycles

Device Type	— Failures Per Sample —					
	1K cyc	2K cyc	5K cyc	8K cyc	10K cyc	Failure
HCMOS PLCC						
XC68HC11A8 (Mask: 1B96D)	1/288	1/287	3/286	0/283	1/283	2.10
XC68HC11A8 (Mask: 2B96D)	3/642	2/633	2/629	2/627	0/625	1.42
MC68HC11A8 (Mask: 2B96D)	1/314	1/313	0/312	0/312	0/312	0.64
MC68HC11A8 (Mask: 7B96D)	3/1289	0/1286	0/1286	1/1286	1/1285	0.39
TOTAL	**8/2533**	**4/2519**	**5/2513**	**3/2508**	**2/2505**	**0.87**

Write/Erase Cycling Failure Rate Calculation

Device Type	Test Device	85°C Device Hrs.	70°C Equiv. Device cyc[1]	Failures	% 1K cyc 0.53 eV[2]
HCMOS PLCC					
MC68HC11A8 (Mask: 1 & 2B96D)	1244	12,440,000	2.58×10^7	17	0.090
MC68HC11A8 (Mask: 7B96D (Current Mask))	1289	12,900,000	2.67×10^7	5	0.035
GRAND TOTAL	**2533**	**25,340,000**	**5.24×10^7**	**22**	**0.056**

1) Activation energy used in equivalent device cycle calculation is 0.53 eV.
2) 90% confidence.

Table 2-17. Average Outgoing Quality

Time Frame	Goal (PPM)	Electrical AOQ (PPM) Actual	Visual/Mech. AOQ (PPM) Actual
Year 1979	3000	(~)4000	(~)4500
Year 1980	2500	(~)2000	(~)2500
Year 1981	1500	1725	1920
Year 1982	900	717	1103
Year 1983	425	383	380
Year 1984	200	419	403
Year 1985	80	272	137
Year 1986	50	291	509
Year 1987	50	232	190

RESULTS AND CONCLUSION

The 1987 Microprocessor Reliability results indicate that the major product lines have excellent overall reliability performance. The reliability performance of our products is evaluated through extensive stress/testing which includes life test, temperature cycle, thermal shock, THB, autoclave, and data retention bake. This year's results indicate there are many areas where significant gains were made in reliability performance as compared to the 1986 results.

The overall High Temperature Operating Life test result for the year was excellent with a failure rate of **117** FITs compared to the 1986 yearly total of **264** FITs (based on 0.7 eV). Failure rate improvements were seen in all of the key process technologies during the year. The life test failure rate for NMOS was **91** FITs which is a **55**% improvement compared to the previous yearly results. The HMOS failure rate improved to a **127** FIT level as compared to the **467** FIT level this technology achieved in 1986. The HCMOS failure rate was **224** FITs which is a **16**% improvement in the 1986 figure. The 5 micron CMOS technology achieved a failure rate of **125** FITs which is excellent and a significant improvement over 1986.

The environmental results for 1987 indicate that our products lines are capable of meeting rigid environmental extremes with very low failure rates. The actual stress results for the various thermal cycling and moisture tests are detailed below.

The temperature cycle results for 1987 improved to an overall **0.43**% cumulative failure rate through 1000 cycles. This is a **51**% gain over the 1986 figures. Thermal shock results for this period also improved substantially to a **0.30**% level. These figures are excellent.

Both temperature humidity bias and autoclave produced improved failure rate performance during 1987. Temperature humidity bias achieved a **0.16**% cumulative failure rate through 1008 hours. The autoclave test for this time frame resulted in a **0.11**% figure which is a **56**% improvement over 1986.

Data retention bake, which is used to evaluate the ability of the MCU EPROM and EEPROM devices to store charge over an extended period of time, has a cumulative percent fallout of **0.19**%. This figure has improved **42**% during 1987 as compared to the previous years results.

Write/erase cycling, which was begun this year to measure the MCU EEPROM arrays operational endurance over an expected life time, resulted in an overall failure rate of 0.056%/1K cycles at 70°C. The most recent material evaluated in the 4th quarter of 1987 achieved a 0.035%/1K cycles failure rate.

Average Outgoing Quality levels for both electrical and visual/mechanical performance improved for 1987. The yearly figures are **232** ppm for electrical and **190** ppm for visual/mechanical.

In summary, the Motorola Microprocessor Product Group's products are achieving very high levels of reliability and quality performance. Improvements in many key areas have been made during the course of the year, and as a group our goal will be to continually upgrade the Reliability and Quality of our products.

For more information, contact Microprocessor Reliability Engineering at 512/440-2530 or write to:

Microprocessor Reliability Engineering
Motorola Inc.
6501 William Cannon Drive West
Austin, Texas 78735-8598

FAILURE RATE CALCULATIONS

Environmental tests are designed to measure device resistance to unusual and severe stress not expected under normal operating conditions. Device performance under these conditions is expressed as a percent of devices defective and compared to previous results. Life tests, on the other hand, accelerate the use conditions of the device with temperature and voltage in a manner which is more quantitatively correlatable to system operation. Life test failure rates are expressed as failures per unit time and are calculated using established principles or probability and statistics.

The principles of reliability engineering have indicated that failure rates for semiconductor devices will take the form of the"bathtub" curve (Figure 2-8).

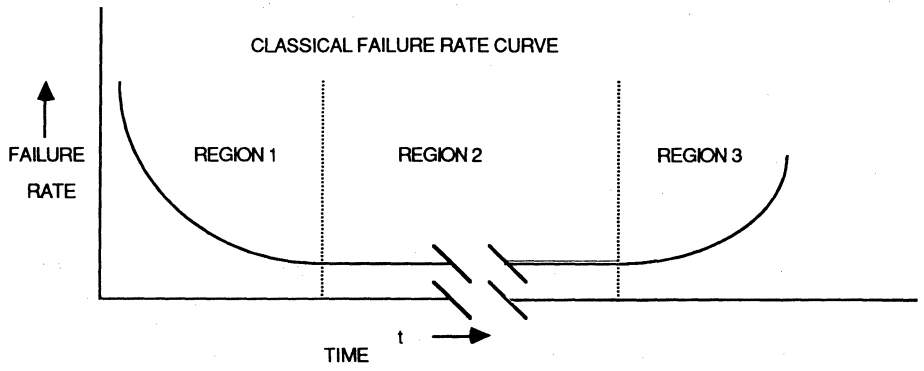


Figure 2-8. Device Failure Rate as a Function of Time

The following three regions are represented in the curve:

1. Infant Mortality — a region of high bur rapidly declining failure rates, usually associated with manufacturing defects.
2. Random Failures — a region of low, random failures caused by more subtle defects. This area of the curve represents the useful part of device life.
3. Wearout — a region of rapidly rising failure rates related to device wearout. Most semiconductors will not reach this stage before they are replaced because of changes in technology.

Techniques for calculating life test failure rates assume that the devices being tested have passed infant mortality and entered the stable random failure portion of the life curve. Failures which occur in this area are few and are known to approximate specific probability distributions. These probability distributions are used to calculate sample failure rates which can be projected to the population in general through the application of confidence limits. Techniques used to calculate life test failure rates for micorprocessors are discussed below.

A failure rate for any sample of life tested devices can be determined by dividing the number of failures by the number of device hours. However, this rate will apply to that sample only. Ff you are interested in projecting from the sample to the population in general, you must establish confidence limits. The application of confidence limits is a statement of how "confident" you are that the sample failure rate approximates that for the population in general. To obtain rates with different confidence levels, it is necessary to make use of specific probability distributions which take the same form as the actual failure distribution.

It has been determined that failures in semiconductors that have entered the middle portion of the bathtub curve will approximate a Poisson distribution; this distribution applies when one has a large sample with an extremely small number of events of interest, such as device failures. Given a Poisson failure process, a Chi-Square distribution can be used to establish confidence limits for failure rates. R&QA Engineering has determined that the following general formula, which utilizes values from a Chi-Square table, can be used to calculate failure rates for semiconductors:

$$\lambda \leq \frac{X^2(\alpha, \text{d.f})}{2t} \quad (1)$$

where:

λ = Failure Rate
x^2 = Chi-Square Function
$\alpha = \frac{100 - \text{Confidence Level}}{100}$
d.f. = Degrees of Freedom = 2r + 2
r = Number of Rejects
t = Device Hours

To calculate the failure rate, first determine the level of confidence you require and calculate degrees of freedom. Select the Chi-Square value for a Chi-Square distribution table with the appropriate degrees of freedom and confidence level. Divide that value by twice the actual device hours, at the temperature of interest.

The above formula applies for calculating a device failure rate, provided that the test is conducted at system temperature. However, since we are unable to observe long-term effects which develop over time, the test is accelerated through the application of a high temperature. In oder to calculate a failure rate at the ambient temperature of a system, a factor must be supplied to compensate for the acceleration. The factor (Fa) which equates test temperature with rated temperature is derived from the Arrhenius relationship:

$$Fa = \exp\left((\phi/k) \cdot \left(\frac{1}{Tr} - \frac{1}{Tt} \right) \right) \quad (2)$$

where:

Fa = Acceleration Factor
ϕ = Activation Energy, eV
k = Boltzman's Constant, 8.62×10^{-5} eV/K
Tr = Junction Temperature, K at the Rated Ambient of 70°C
Tt = Junction Temperature, K at the Life Test Ambient of 125°C

Motorola uses 70°C for the system temperature (To) to more closely approximate the actual temperature of the device during system operation and to supply a degree of conservatism to the failure rate calculation.

Motorola uses an activation energy value of 0.7 electron volt. A 0.7 eV was selected as an average value because a variety of different failure mechanisms exist for microprocessor and other VLSI

devices, with activation energies ranging from 0.40 eV for oxide related failures to 1.0 eV or greater for contamination and metal related failures. Tr and Tt of the equation are the average junction temperatures present at the rated and test ambients. Motorola uses junction, rather than ambient temperature, because they produce acceleration factors that are more conservative and representative of actual conditions. These temperatures are calculated as follows:

$$T_J = T_A + P_D \cdot \theta_{JA} \quad (3)$$

where:

T_J = Junction Temperature, °C
T_A = Ambient Temperature, °C
P_D = Average Power Dissipation, Watts
θ_{JA} = Thermal Resistance — Junction to Ambient, °C Per Watt

Once this step has been completed, the acceleration factor can be calculated and applied as a multiplier to the number of device test hours under accelerated test conditions to determine the equivalent number of hours at rated operating conditions. To determine the failure rate at the operating temperature use equation (1) substituting the equivalent device hours at rated temperature for t in the equation.

Equation (1) provides a failure rate expressed in percent per thousand hours. This number, stated as a percentage per each thousand hours of operation, is one way Motorola R&QA Engineering expresses failure rates for Microprocessors. One other way of expressing failure rates is Failures In Time (FITs) which refers to failed units per 10^9 device hours ($1\ FIT = \lambda \times 10^4$).

Mean Time To Failure (MTTF) is another parameter frequently used to express failure rates. MTTF is the average time to a failure of a nonrepairable item such as a semiconductor and is expressed as the reciprocal of the failure rate:

$$MTTF = \frac{1}{\lambda} \quad (4)$$

Data Sheets
Volume I and II

3

This chapter (found in both Volume I and Volume II) contains the data sheets for the Microprocessors, Microcontrollers, and Peripheral devices. For information on packaging, refer to Chapter 4. Ordering forms are located in Chapter 6.

3

MC2672

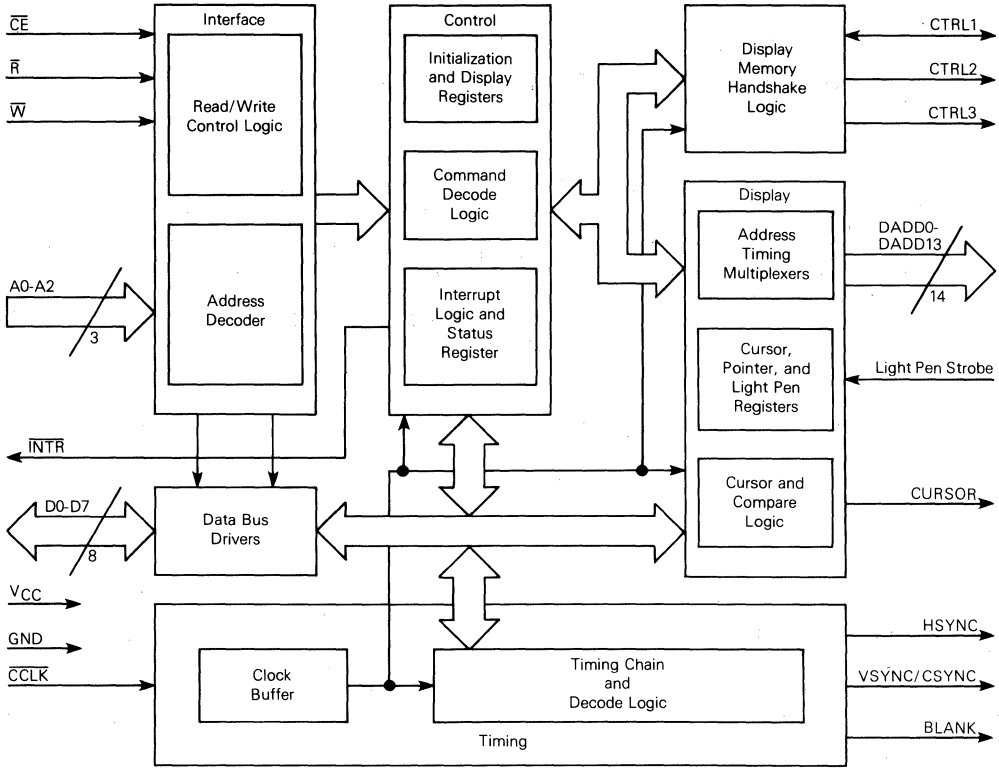
Advance Information

Programmable Video Timing Controller (PVTC)

The MC2672 programmable video timing controller (PVTC) is a programmable device designed for use in CRT terminals and displays systems that employ raster scan techniques. The PVTC generates the vertical and horizontal timing signals necessary for the display of interlaced or non-interlaced data on a CRT monitor. It provides consecutive addressing to a user specified display buffer memory domain and controls the CPU-display buffer interface for various buffer configuration modes. A variety of operating modes, display formats, and timing profiles can be implemented by programming the control registers in the PVTC. Applications include CRT terminals, word-processing systems, small business computers, and home computers.

- 4 MHz Character Rate
- Up to 256 Characters Per Row
- 1 to 16 Raster Lines Per Character Row
- Up to 128 Character Rows Per Frame
- Programmable Horizontal and Vertical Sync Generators
- Interlaced or Non-Interlaced Operation
- Up to 16K RAM Addressing for Multiple Page Operation
- Automatic Wraparound of RAM
- Addressable, Incrementable, and Readable Cursor
- Programmable Cursor Size, Position, and Blink
- Split Screen and Horizontal Scroll Capability
- Light Pen Register
- Selectable Buffer Interfce Modes
- Dynamic RAM Refresh
- Completely TTL Compatible
- Single +5-Volt Power Supply
- Power-On Reset Circuit

BLOCK DIAGRAM



3

ABSOLUTE MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range	T_A	0 to 70	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Rating
Thermal Resistance Plastic Package	θ_{JA}	50	°C/W

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $GND \leq (V_{in}$ or $V_{out}) \leq V_{CC}$. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (e.g., either GND or V_{CC}).

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A

3

DC ELECTRICAL CHARACTERISTICS ($T_A = 0°C$ to 70°C, $V_{CC} = 5.0$ V ±5%)

Parameter	Symbol	Min	Max	Unit
Input Low Voltage	V_{IL}	−0.3	0.8	V
Input High Voltage	V_{IH}	2.0	V_{CC}	V
Output Low Voltage ($I_{Load} = 2.4$ mA)	V_{OL}	–	0.4	V
Output High Voltage (Except $\overline{INTR}$ Output) $I_{Load} = -200$ μA	V_{OH}	2.4	–	V
Input Leakage Current $V_{in} = 0$ to V_{CC}	I_{in}	−10	10	μA
Hi-Z (Offstate) Input Current $V_{in} = 0.4$ to 2.4 V	I_{TSI}	−10	10	μA
$\overline{INTR}$ Open-Drain Output Leakage Current $V_{OH} = 2.4$ V_{CC}	I_{LOH}	–	10	μA
Internal Power Dissipation	P_{INT}	–	800	mW

NOTE: All voltage measurements are referenced to ground. All time measurements are at the 0.8 V to 2.0 V level for inputs and outputs. Input levels are 0.4 V to 2.4 V.

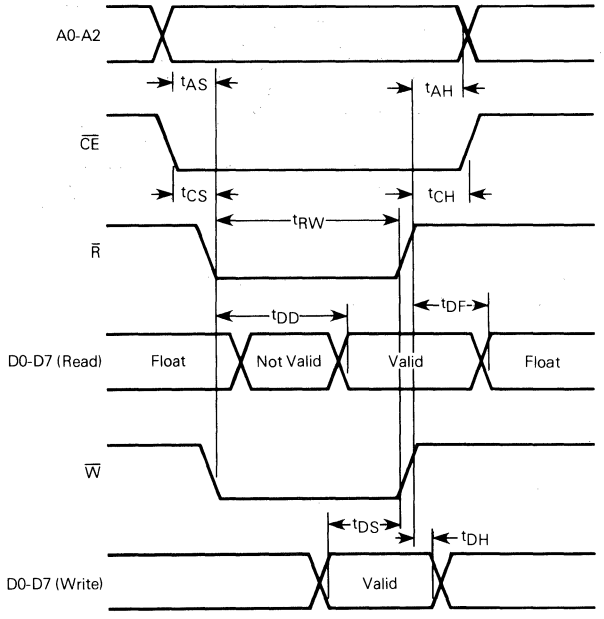
AC ELECTRICAL CHARACTERISTICS — BUS TIMING (T_A=0° to 70°C, V_{CC}=5.0 V ±5%, See Note 1)

Parameter	Symbol	MC2672B3 Min	MC2672B3 Max	MC2672B4 Min	MC2672B4 Max	Unit
A0-A2 Setup Time to $\overline{W}$, $\overline{R}$ Low	t_{AS}	30	–	30	–	ns
A0-A2 Hold Time from $\overline{W}$, $\overline{R}$ High	t_{AH}	0	–	0	–	ns
$\overline{CE}$ Setup Time to $\overline{W}$, $\overline{R}$ Low	t_{CS}	0	–	0	–	ns
$\overline{CE}$ Hold Time from $\overline{W}$, $\overline{R}$ High	t_{CH}	0	–	0	–	ns
$\overline{W}$, $\overline{R}$ Pulse Width	t_{RW}	250	–	250	–	ns
Data Valid after $\overline{R}$ Low	t_{DD}	–	200	–	200	ns
Data Bus Floating after $\overline{R}$ High	t_{DF}	–	100	–	100	ns
Data Setup Time to $\overline{W}$ High	t_{DS}	150	–	150	–	ns
Data Hold Time from $\overline{W}$ High	t_{DH}	10	–	5	–	ns
High Time from $\overline{CE}$ to $\overline{CE}$ (see Note 2) Consecutive Commands	t_{CC}	600	–	600	–	ns
Other Commands		300	–	300	–	ns

NOTES:
1. Timing is illustrated and specified referenced to $\overline{W}$ and $\overline{R}$ inputs. Device may also be operated with $\overline{CE}$ as the "strobing" input. In this ase, all timing specifications apply referenced to falling and rising edges of $\overline{CE}$.
2. This specification requires that the $\overline{CE}$ input be negated (high) between read and/or write cycles.
3. All voltage measurements are referenced to ground. All time measurements are at the 0.8 V to 2.0 V level for inputs and outputs. Input levels are 0.4 V to 2.4 V.

3

BUS TIMING DIAGRAM

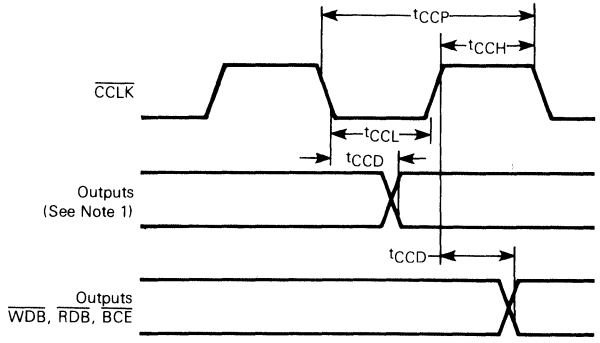


AC ELECTRICAL CHARACTERISTICS — CHARACTER CLOCK TIMING (T_A = 0°C to 70°C, V_{CC} = 5.0 V ±5%, See Note 1)

Parameter	Symbol	MC2672B3		M2672B4		Unit
		Min	Max	Min	Max	
$\overline{CCLK}$ Period	t_{CCP}	370	–	250	–	ns
$\overline{CCLK}$ High Time	t_{CCH}	125	–	100	–	ns
$\overline{CCLK}$ Low Time	t_{CCL}	125	–	100	–	ns
Output Delay Time from $\overline{CCLK}$ Edge	t_{CCD}					ns
DADD0-DADD13, $\overline{BCE}$, $\overline{WDB}$, $\overline{RDB}$, MBC		40	175	40	150	
BLANK, HSYNC, VSYNC/CSYNC, CURSOR, $\overline{BEXT}$, $\overline{BREQ}$, $\overline{BACK}$		40	225	40	200	

NOTES:
1. $\overline{BCE}$, $\overline{WDB}$, and $\overline{RDB}$ delays track each other within 10 nanoseconds. Also, these output delays will tend to follow the direction (minimum/maximum) of DADD0-DADD13 delays.
2. All voltage measurements are referenced to ground. All time measurements are at the 0.8 V to 2.0 V level for inputs and outputs. Input levels are 0.4 V to 2.4 V.

CHARACTER CLOCK TIMING DIAGRAM



NOTES:
1. DADD0-DADD13, BLANK, HSYNC, CSYNC/VSYNC, CURSOR, $\overline{BEXT}$, $\overline{BREQ}$, $\overline{BCE}$, MBC, $\overline{BACK}$.
2. $\overline{BCE}$ changes state on both $\overline{CCLK}$ edges.

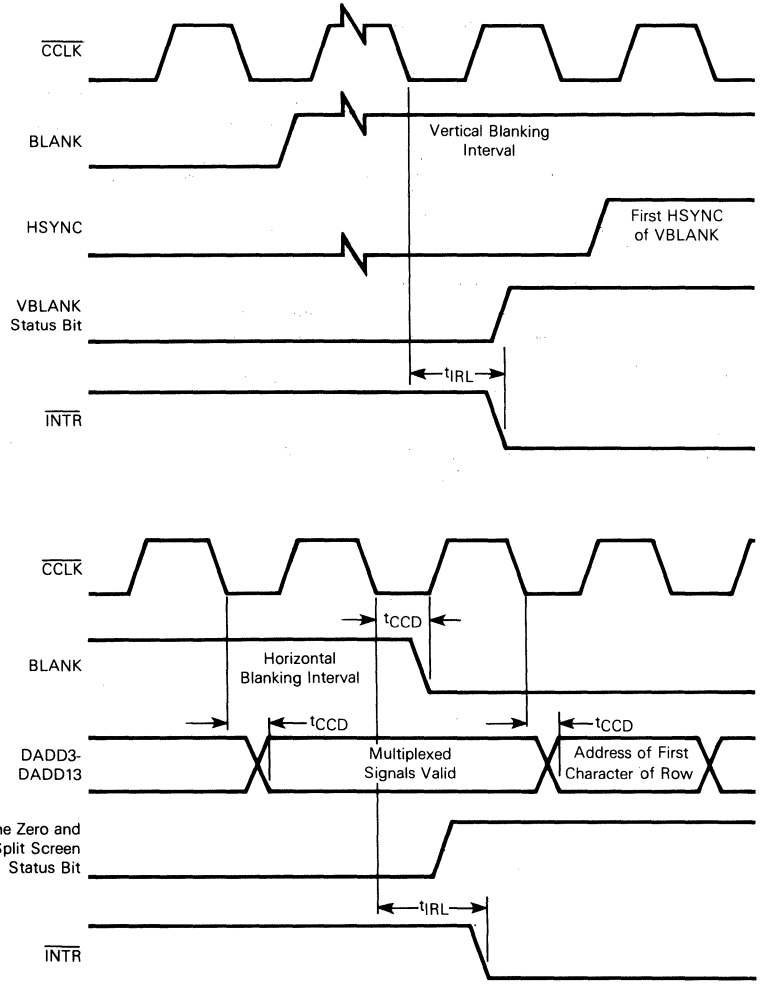
3

AC ELECTRICAL CHARACTERISTICS — OTHER TIMINGS (T_A = 0°C to 70°C, V_{CC} = 5.0 V ±5%)

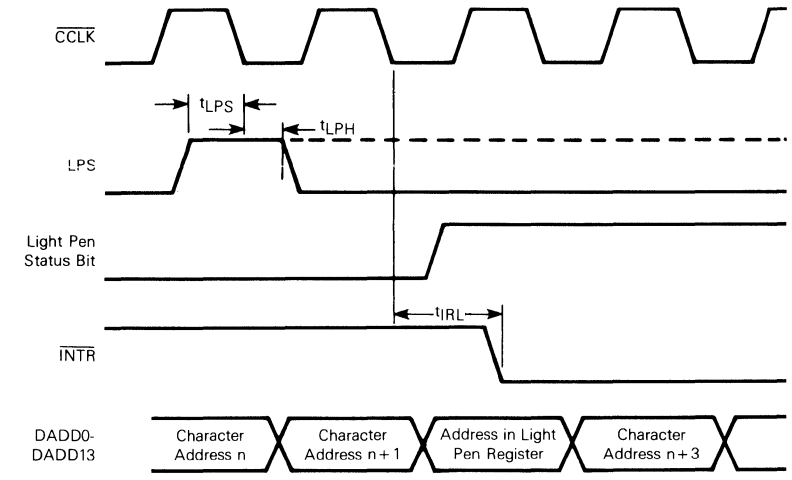
Parameter	Symbol	MC2672B3 Min	MC2672B3 Max	MC2672B4 Min	MC2672B4 Max	Unit
READY/RDFLG Low from $\overline{W}$ HIGH	t_{RDL}	–	t_{CCP} + 30	–	t_{CCP} + 30	ns
$\overline{BACK}$ High from $\overline{PBREQ}$ Low	t_{BAK}	–	225	–	200	ns
$\overline{BEXT}$ High from $\overline{PBREQ}$ High	t_{BXT}	–	225	–	200	ns
Light Pen Strobe Setup Time to $\overline{CCLK}$ Low	t_{LPS}	120	–	120	–	ns
Light Pen Strobe Hold Time from $\overline{CCLK}$ Low	t_{LPH}	−10	–	−10	–	ns
$\overline{INTR}$ Low from $\overline{CCLK}$ Low	t_{IRL}	–	225	–	200	ns
$\overline{INTR}$ High from $\overline{W}$, $\overline{R}$ High	t_{IRH}	–	600	–	600	ns

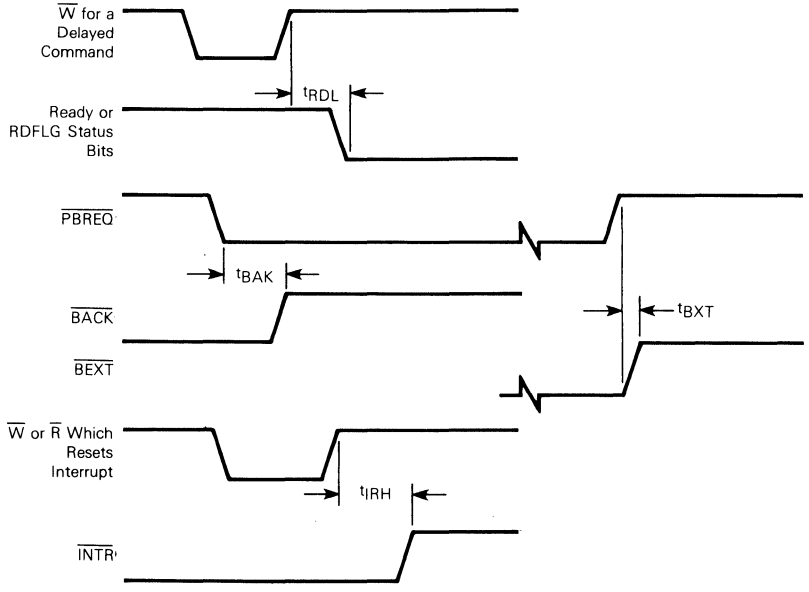
NOTES:
1. Timing is illustrated and specified referenced to $\overline{W}$ and $\overline{R}$ inputs. Device may also be operated with $\overline{CE}$ as the "strobing" input. In this case, all timing specifications apply referened to falling and rising edges of $\overline{CE}$.
2. All voltage measurements are referenced to ground. All time measurements are at the 0.8 V to 2.0 V level for inputs and outputs. Input levels are 0.4 V to 2.4 V.

OTHER TIMING DIAGRAMS

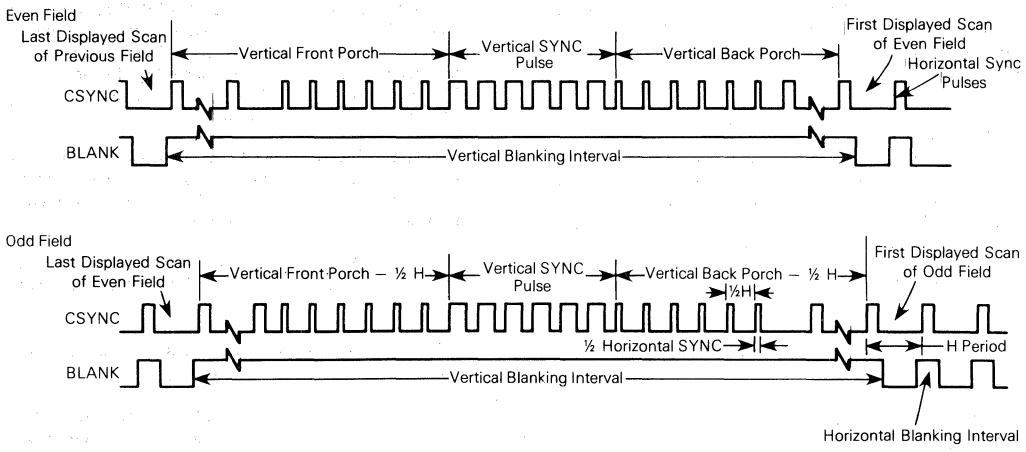


OTHER TIMING DIAGRAMS (Continued)





COMPOSITE SYNC TIMING DIAGRAM



3

NOTES:
1. In non-interlaced operation the even field is repeated continuously, and the odd field is not.
2. Interlaced operation the even field alternates with the odd field.
3. All voltage measurements are referenced to ground. All time measurements are at the 0.8 V to 2.0 V level for inputs and outputs. Input levels are 0.4 V to 2.4 V.

SIGNAL DESCRIPTION

The input and output signals for the PVTC are described in the following paragraphs.

V_{CC} AND GND

Power is supplied to the PVTC using these two pins. V_{CC} is the +5 volts ±5% power input and GND is the ground connection.

ADDRESS LINES (A0-A2)

These lines are used to select PVTC internal registers for read/write operations and for commands.

DATA BUS (D0-D7)

These lines comprise the 8-bit bidirectional three-state data bus. Bit 0 is the least significant bit and bit 7 is the most significant bit. All data, command, and status transfers between the CPU and the PVTC take place over this bus. The direction of the transfer is controlled by the read and write inputs when the chip enable input is low. When the chip enable input is high the data bus is in the high-impedance state.

READ STROBE ($\overline{R}$)

This pin is an active low input. A low on this pin while chip enable is low causes the contents of the register selected by A0-A2 to be placed on the data bus. The read cycle begins on the falling edge of $\overline{R}$.

WRITE STROBE ($\overline{W}$)

This pin is an active low input. A low on this pin while chip enable is also low causes the contents of the data bus to be transferred to the register selected by A0-A2. The transfer occurs on the rising edge of $\overline{W}$.

CHIP ENABLE ($\overline{CE}$)

This pin is an active low input. When low, data transfers between the CPU and the PVTC are enabled on D0-D7 as controlled by the $\overline{W}$, $\overline{R}$, and A0-A2 inputs. When $\overline{CE}$ is high, the PVTC is effectively isolated from the data bus and D0 through D7 are placed in the high-impedance state.

CHARACTER CLOCK ($\overline{CCLK}$)

This pin is the timing signal derived from the video dot clock which is used to synchronize the PVTC's timing functions.

HORIZONTAL SYNC (HSYNC)

This pin is an active high output which provides video horizontal sync pulses. The timing parameters are programmable.

VERTICAL SYNC/COMPOSITE SYNC (VSYNC/CSYNC)

A control bit selects either vertical or composite sync pulses on this active high output. When CSYNC is selected, equalization pulses are included. The timing parameters are programmable.

BLANK (BLANK)

This active high output defines the horizontal and vertical borders of the display. Display control signals which are output on DADD3 through DADD13 are valid on the trailing edge of BLANK.

CURSOR GATE (CURSOR)

This active high output becomes active for a specified number of scan lines when the address contained in the cursor registers matches the address output on the display address (DADD0 through DADD13). The first and last lines of the cursor and a blink option are programmable.

INTERRUPT REQUEST ($\overline{\text{INTR}}$)

This pin is an open-drain output which supplies an active low interrupt request from any of five maskable sources. This pin is inactive after power-on reset or a master reset command.

LIGHT PEN STROBE (LPS)

This positive edge triggered input indicates a light pen 'hit' causing the current value of the display address to be strobed into the light pen register.

HANDSHAKE CONTROL 1 (CTRL1)

In independent mode, this pin provides an active low write data buffer ($\overline{\text{WDB}}$) output which strobes data from the interface latch into the display memory. In transparent and shared modes, this is an active low processor bus request ($\overline{\text{PBREQ}}$) input which indicates that the CPU desires to access the display memory. This pin must be tied high when operating in row-buffer mode.

HANDSHAKE CONTROL 2 (CTRL2)

In independent mode, this pin provides an active low read data buffer ($\overline{\text{RDB}}$) output which strobes data from the display memory into the interface latch. In transparent and shared modes, CTRL2 is an active low bus external enable ($\overline{\text{BEXT}}$) output which indicates that the PVTC has relinquished control of the display memory (DADD0-DADD13 are in the high-impedance state) in response to a CPU bus request. $\overline{\text{BEXT}}$ also goes low in response to a "display off and float DADD" command. In row-buffer mode, CTRL2 is an active low bus request ($\overline{\text{BREQ}}$) output which halts the CPU during a line DMA.

HANDSHAKE CONTROL 3 (CTRL3)

In independent mode, this pin provides the active low buffer chip enable ($\overline{\text{BCE}}$) signal to the display memory. In transparent and shared modes, CTRL3 provides an active low bus acknowledge ($\overline{\text{BACK}}$) output which serves as a ready signal to the CPU in response to a processor bus request. In row buffer mode, CTRL3 is an active high memory bus control (MBC) output which configures the system for the DMA transfer of one row of character codes from system memory to the row display buffer.

DISPLAY ADDRESS (DADD0-DADD13)

The display address is used by the PVTC to address up to 16K of display memory. These outputs are floated at various times depending on the buffer mode. Various control signals are multiplexed on DADD3 through DADD13 and are valid at the trailing edge of BLANK. The following paragraphs describe these control signals.

LINE INTERLACE (DADD3/LI) — Replaces DADD4/LA0 as the least significant line address for interlaced sync and video applications. A low indicates an even row of an even field or an odd row of an odd field.

LINE ADDRESS (DADD4-DADD7/LA0-LA3) — Provides the number of the current scan line within each character row.

LINE ZERO (DADD8/LNZ) — Asserted before the first scan line in each character row.

LIGHT PEN LINE (DADD9/LPL) — Asserted before the scan line which matches the programmed light pen line position (line three, five, seven, or nine).

UNDERLINE (DADD10/UL) — Asserted before the scan line which matches the programmed underline position (line 0 through 15).

BLINK FREQUENCY (DADD11/BLINK) — Provides an output divided down from the vertical sync rate.

ODD FIELD (DADD12/ODD) — Active high signal which is asserted before each scan line of the odd field when interlace is specified.

LAST LINE (DADD13/LL) — Asserted before the last scan line of character row.

FUNCTIONAL DESCRIPTION

The following paragraphs describe the major blocks (data-bus buffer, interface logic, operation control, timing, display control, and buffer control) which comprise the PVTC.

DATA-BUS BUFFER

The data-bus buffer provides the interface between the external and internal data buses. It is controlled by the operation control block to allow read and write operations to take place between the controlling CPU and the PVTC.

INTERFACE LOGIC

The interface logic contains address decoding and read and write circuits to permit communications with the microprocessor via the data-bus buffer. The functions performed by the CPU read and write operations are as shown in Table 1.

TABLE 1 — PVTC ADDRESSING

A2	A1	A0	Read ($\overline{R}$=0)	Write ($\overline{W}$=0)
0	0	0	Interrupt Register	Initialization Registers*
0	0	1	Status Register	Command Register
0	1	0	Screen Start Address Lower Register	Screen Start Address Lower Register
0	1	1	Screen Start Address Upper Register	Screen Start Address Upper Register
1	0	0	Cursor Address Lower Register	Cursor Address Lower Register
1	0	1	Cursor Address Upper Register	Cursor Address Upper Register
1	1	0	Light Pen Address Lower Register	Display Pointer Address Lower Register
1	1	1	Light Pen Address Upper Register	Display Pointer Address Upper Register

*There are 11 initialization registers which are accessed sequentially via a simple address. The PVTC maintains an internal pointer to these registers which is incremented after each write at this address until the last register (IR10, the split-screen register) is accessed. The pointer then continues to point to the split-screen register. Upon power-up or a master reset command, the internal pointer is reset to point to the first register (IR0) of the initialization register group. The internal pointer can also be preset to any register of the group via the "load IR address pointer" command.

3

OPERATION CONTROL

The operation control section decodes configuration and operation commands from the CPU and generates appropriate signals to other internal sections to control the overall device operation. It contains the timing and display registers which configure the display format and operating modes, the interrupt logic, and the status register which provides operational feedback to the CPU.

TIMING

The timing section contains the cursors and decoding logic necessary to generate and monitor timing outputs and to control the display format. These timing parameters are selected by programming of the initialization registers.

DISPLAY CONTROL

The display control section generates linear addressing of up to 16K bytes of display memory. Internal comparators limit the portion of the memory which is displayed to programmed values. Additional functions performed in this section include cursor positioning, storage of light pen "hit" locations, and address comparisons required for generation of timing signals and the split-screen interrupt.

BUFFER CONTROL

The buffer control section generates three signals which control the transfer of data between the CPU and the display buffer memory. Four system configurations requiring four different handshaking schemes are supported. These are described in **SYSTEM CONFIGURATIONS.**

SYSTEM CONFIGURATIONS

Figure 1 illustrates the block diagram of a typical display terminal that uses an MC2672, character ROM, a keyboard interface, and an attribute controller. In this system, the CPU examines inputs from the data communications line and the keyboard and places the data to be displayed in the display buffer memory. The buffer is typically a RAM wich holds the data for a single or multiple screenload (page) or for a single character row.

The PVTC supports four common system configurations of display buffer memory, designated the independent, transparent, shared, and row-buffer modes. The first three modes utilize a single or multiple page RAM and differ primarily in the means used to transfer display data between the RAM and the CPU. The row-buffer mode makes use of a single row buffer (which can be shift register or a small RAM) that is updated in real time to contain the appropriate display data.

The user program bits 0 and 1 of IR0 to select the mode best suited for the system environment. The CNTRL1-CNTRL3 outputs perform different functions for each mode and are named accordingly in the description of each mode given in the following paragraphs.

INDEPENDENT MODE

The CPU-to-RAM interface configuration for this mode is illustrated in Figure 2. Transfer of data between the CPU and display memory is accomplished via a bidirectional latched port and is controlled by the signals read data buffer ($\overline{RDB}$), write data buffer ($\overline{WDB}$), and buffer chip enable ($\overline{BCE}$). This mode provides a non-contention type of operation that does not address the memory directly. The read or write operation is performed at the address contained in the cursor address register or the pointer address register as specified by the CPU. The PVTC enacts the data transfers during blanking intervals in order to prevent visual disturbances of the displayed data.

The CPU manages the data transfers by supply commands to the PVTC. The commands used are:

1. Read/write at pointer address.
2. Read/write at cursor address (with optional increment of address).
3. Write from cursor address to pointer address.

The operational sequence for a write operation is:

1. CPU checks RDFLG status bit to assure that any previous operation has been completed.
2. CPU loads data to be written to display memory into the interface latch.
3. CPU writes address into cursor or pointer registers.
4. CPU issues "write at cursor with/without increment" or "write at pointer" command.
5. PVTC generates control signals and outputs specified address to perform requested operation. Data is copied from the interface latch into the memory.

6. PVTC sets RDFLG status to indicate that the write is completed.

Similarly, a read operation proceeds as follows:

1. Steps 1. and 3. as above
2. CPU issues "read at cursor with/without increment" or "read at pointer" command.
3. PVTC generates control signals and outputs specified address to perform requested operation. Data is copied from memory to the interface latch and PVTC sets RDFLG status to indicate that the read is complete.
4. CPU checks RDFLG status to see if operation is completed.
5. CPU reads data from interface latch.

Loading the same data into a block of display memory is accomplished via the "write from cursor-to-pointer" command:

1. CPU checks RDFLG status bit to assure that any previous operation has been completed.
2. CPU loads data to be written to display memory into the interface latch.
3. CPU writes beginning address of memory block into cursor address register and ending address of block into pointer address register.
4. CPU issues "write from cursor-to-pointer" command.
5. PVTC generates control signals and outputs block addresses to copy data from the interface latch into the specified block of memory.
6. PVTC sets RDFLG status to indicate that the block write is completed.

Similar sequences can be implemented on an interrupt driven basis using the READY interrupt output to advise the CPU that a previously requested command has been completed.

Two timing sequences are possible for the "read/write at cursor/pointer" commands. If the command is given during the active display window (defined as first scan line of the first character row to the last scan line of the last character row), the operation takes place during the next horizontal blanking interval, as illustrated in Figure 3. If the command is given during the vertical blanking interval, or while the display has been commanded blanked, the operation takes place immediately. In the latter case, the execution time for the command is approximately one microsecond plus six character clocks (see Figure 4).

Timing for the "write from cursor-to-pointer" operation is shown in Figure 5. The BLANK output is asserted automatically and remains asserted until the vertical retrace interval following completion of the command. The memory is filed at a rate of one location per two character times, plus a small amount of overhead.

FIGURE 1 — CRT TERMINAL BLOCK DIAGRAM

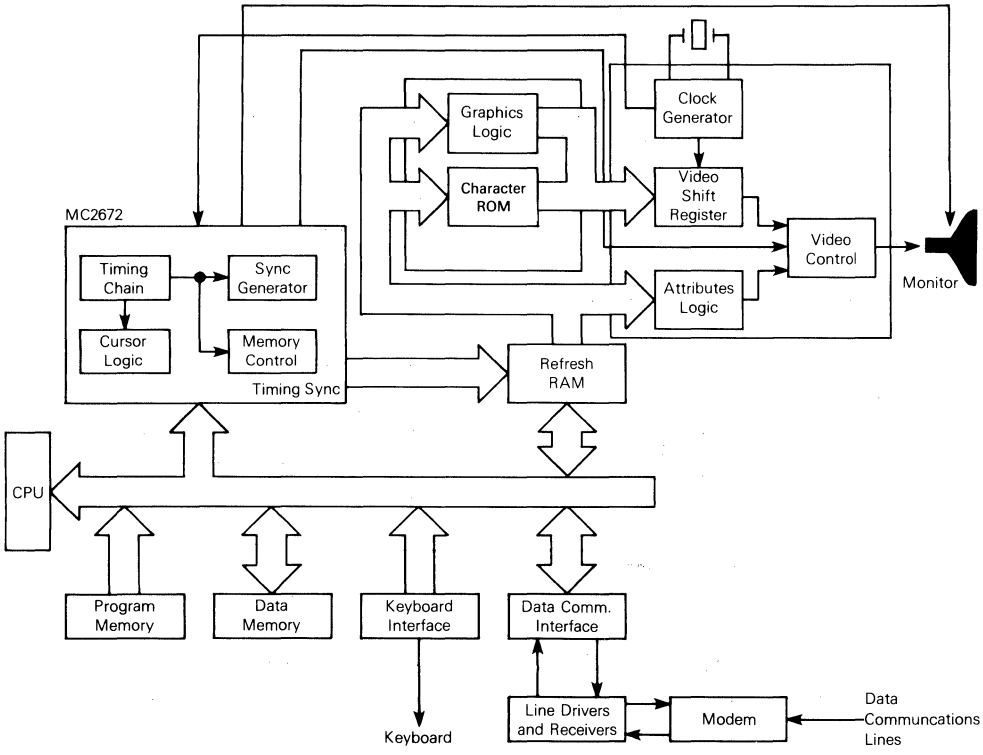


FIGURE 2 — INDEPENDENT BUFFER-MODE CONFIGURATION

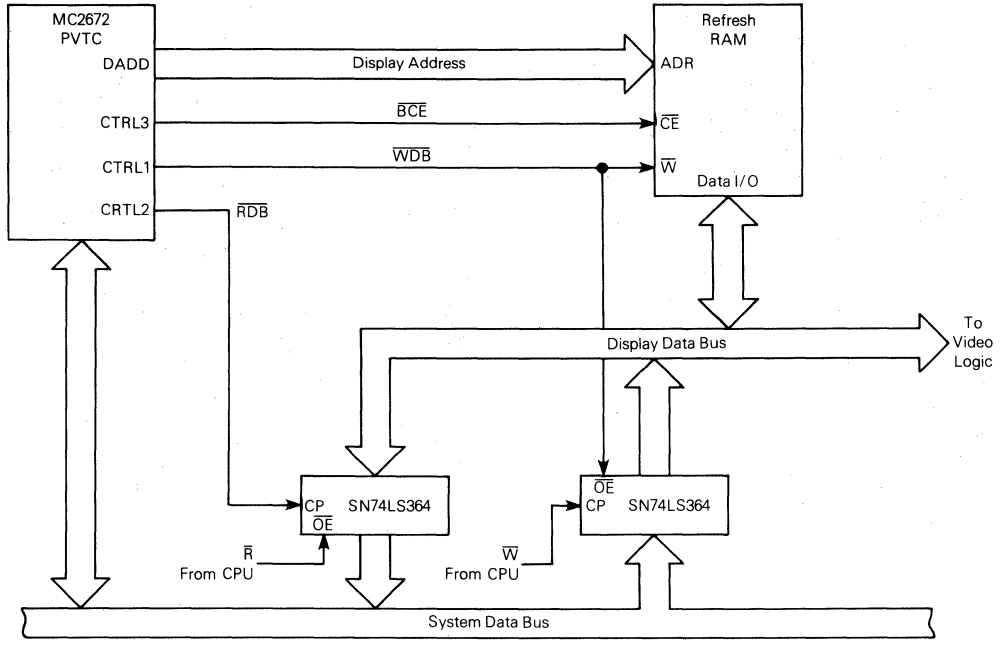
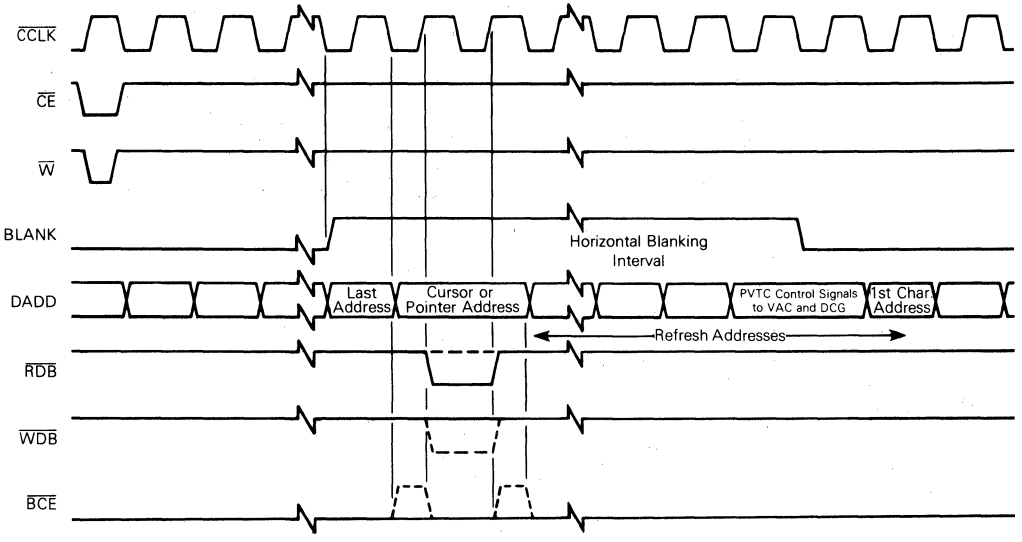


FIGURE 3 — READ/WRITE AT CURSOR/POINTER COMMAND TIMING DIAGRAM
(Command Received During Active Display Window)



NOTE: Write waveforms shown in dotted lines.

FIGURE 4 — READ/WRITE AT CURSOR/POINTER COMMAND TIMING DIAGRAM
(Command Received While Display is Blanked)

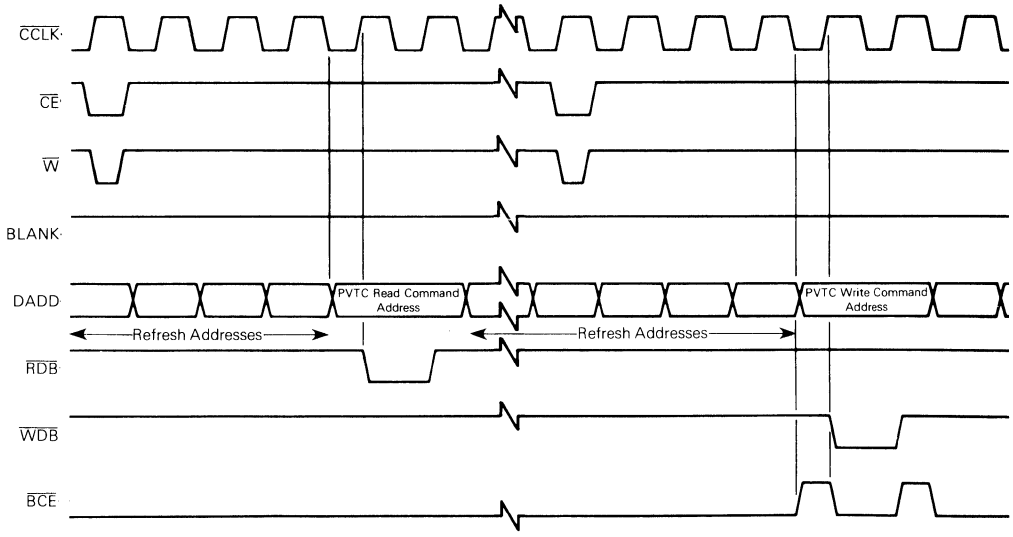
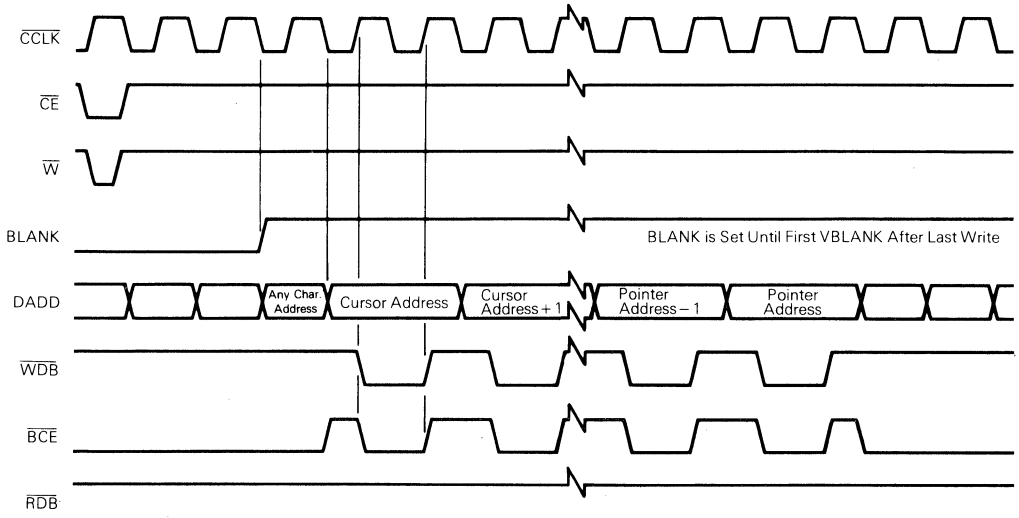


FIGURE 5 — WRITE FROM CURSOR-TO-POINTER COMMAND TIMING



SHARED AND TRANSPARENT BUFFER MODES

In these modes the display buffer RAM is a part of the CPU memory domain and is addressed directly by the CPU. Both modes use the same hardware configuration with the CPU accessing the display buffer via three-state drivers (see Figure 6). The processor bus request ($\overline{PBREQ}$) control signal informs the PVTC that the CPU is requesting access to the display buffer. In response to this request, the PVTC raises bus acknowledge ($\overline{BACK}$) until its bus external ($\overline{BEXT}$) output has freed the display address and data buses for CPU accesses. $\overline{BACK}$, which can be used as a "hold" input to the CPU, is then lowered to indicate that the CPU can access the buffer.

In transparent mode, the PVTC delays the granting of the buffer to the CPU until a vertical or horizontal blanking interval, thereby causing minimum disturbance of the display. In shared mode, the PVTC will blank the display and grant immediate access to the CPU. Timing for these modes is illustrated in Figures 7, 8, and 9.

FIGURE 6 — PVTC SHARED OR TRANSPARENT BUFFER MODES

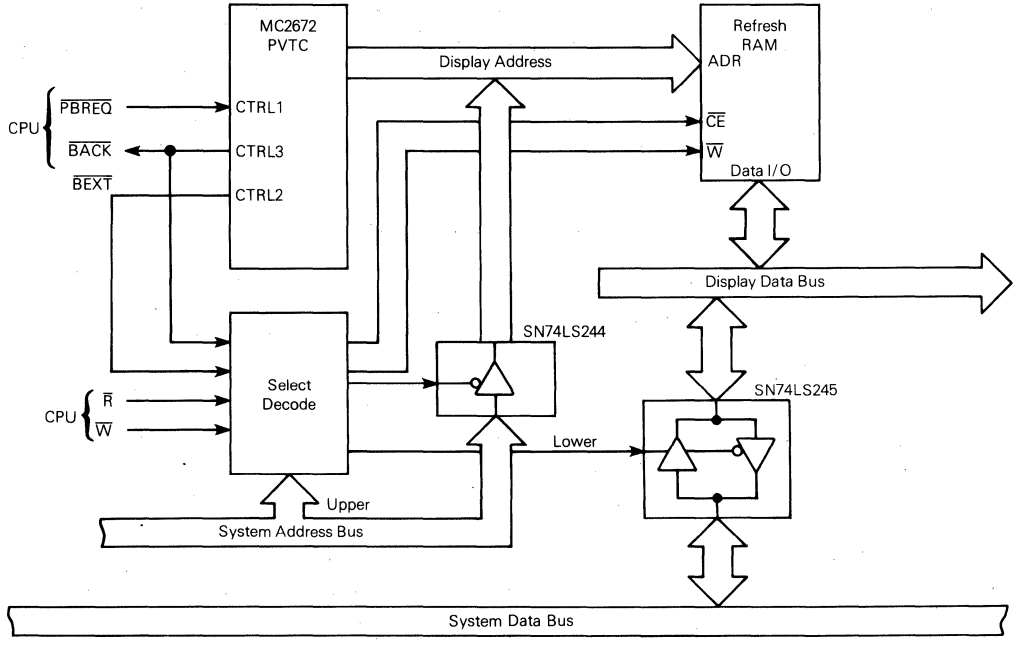
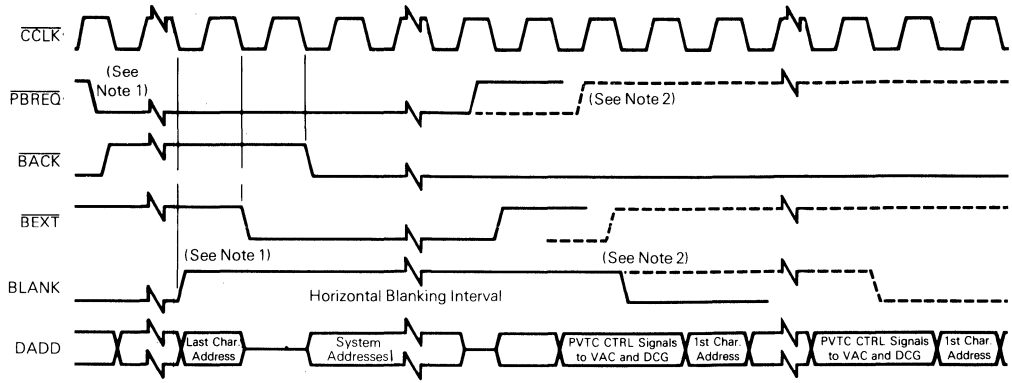


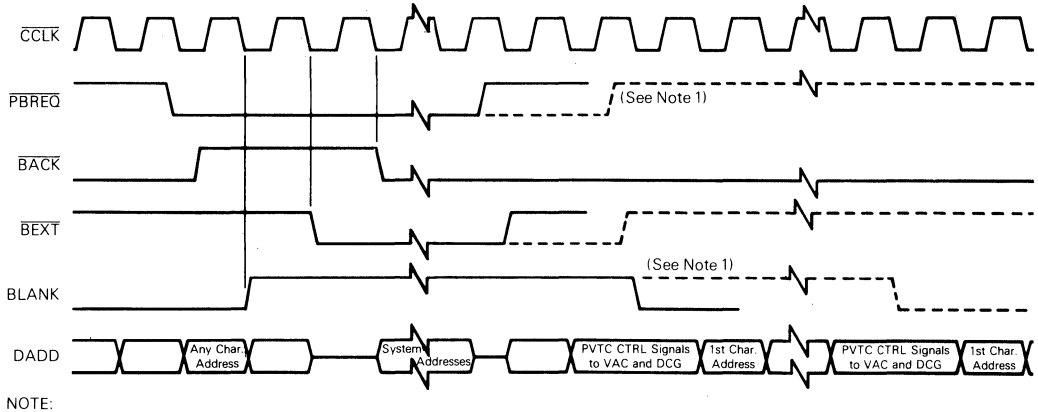
FIGURE 7 — TRANSPARENT-BUFFER MODE TIMING



NOTES:
1. $\overline{\text{PBREQ}}$ must be asserted prior to the rising edge of BLANK in order for sequence to begin during that blanking period.
2. If $\overline{\text{PBREQ}}$ is negated after the next to last $\overline{\text{CCLK}}$ of the horizontal blanking interval, the next scan line will also be blanked.

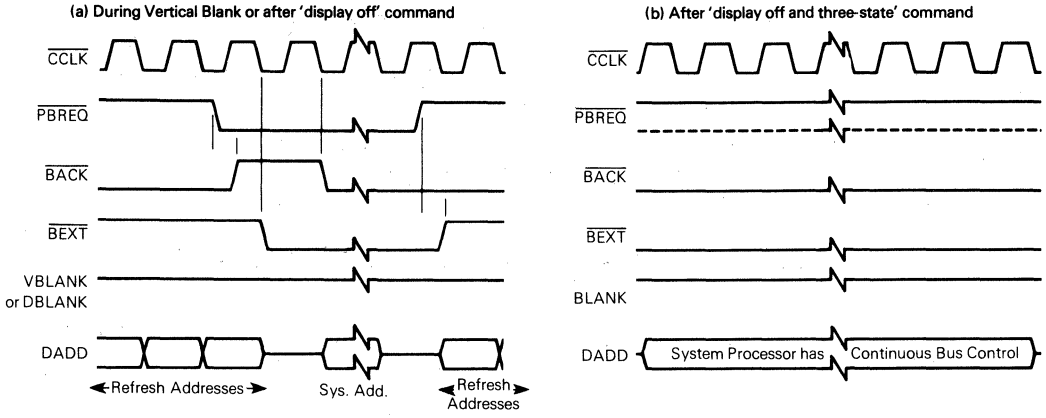
3

FIGURE 8 — SHARED-BUFFER MODE TIMING



NOTE:
1. If $\overline{\text{PBREQ}}$ is negated after the next to last $\overline{\text{CCLK}}$ of the horizontal blanking interval, the next scan line will also be blanked.

FIGURE 9 — SHARED AND TRANSPARENT MODE TIMING



3

ROW-BUFFER MODE

Figures 10 and 11 show the timing and a typical hardware implementation for the row-buffer mode. During the first scan line (line 0) of each character row, the PVTC halts the CPU and DMA's the next row of character data from the system memory to row-buffer memory. The PVTC then releases the CPU and displays the row-buffer data for the programmed number of scan lines. The bus-request control ($\overline{BREQ}$) signal informs the CPU that character addresses and the memory bus control (MBC) signal will start at the next falling edge of BLANK. The CPU must release the address and data buses before this time to prevent bus contention. After the row of character data is transferred to the CPU, $\overline{BREQ}$ returns high to grant memory control back to the CPU.

FIGURE 10 — ROW-BUFFER MODE CONFIGURATION

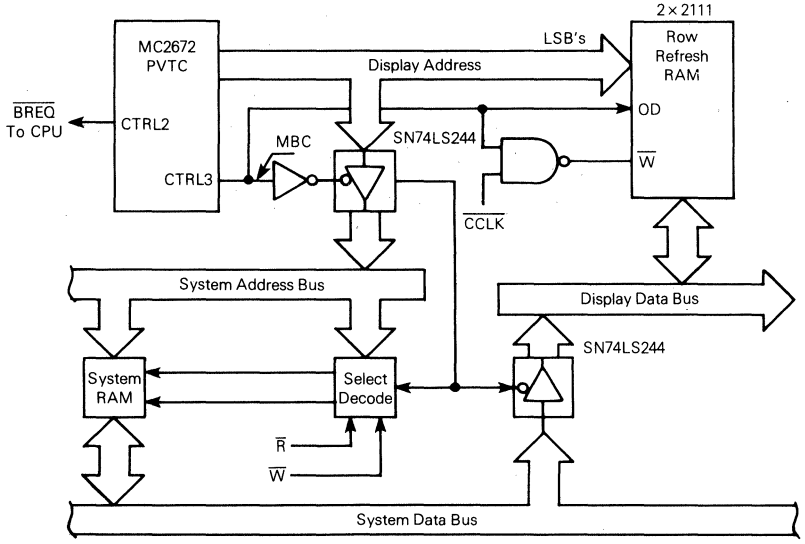
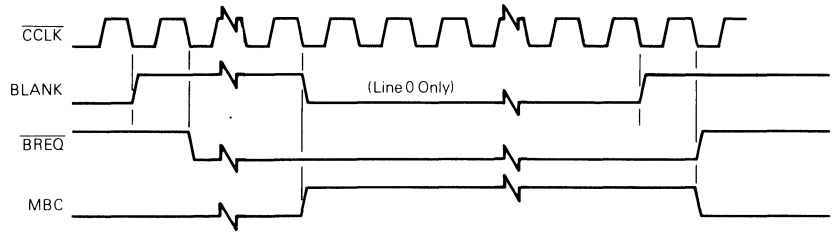


FIGURE 11 — ROW-BUFFER MODE TIMING



OPERATION

After power is applied, the PVTC will be in an inactive state. Two consecutive "master reset" commands are necessary to release this circuitry and ready the PVTC for operation. Two register groups exist within the PVTC: the initialization registers and the display control registers. The initialization registers select the system configuration, monitor timing, cursor shape, display memory domain, and screen format. These are loaded first and normally require no modification except for certain special visual effects. The display control registers specify the memory address of the base character (upper left corner of screen), the cursor position, and the pointer address for independent memory access mode. These usually require modification during operation.

After initial loading of the two register groups, the PVTC is ready to control the monitor screen. Prior to executing the PVTC commands which turn on the display and cursor, the user should load the display memory with the first data to be displayed. During operation, the PVTC will sequentially address the display memory within the limits programmed into its registers. The memory outputs character codes to the system character and graphics generation logic, where they are converted to the serial video stream necessary to display the data on the CRT. The user effects changes to the display by modifying the contents of the display memory, the PVTC display control and command registers, and the initialization registers, if required. Interrupts and status conditions generated by the PVTC supply the "handshaking" information necessary for the CPU to effect the display changes in the proper time frame.

INITIALIZATION REGISTERS

There are 11 initialization registers (IR0-IR10) which are accessed sequentially via a single address. The PVTC maintains an internal pointer to these registers which is incremented after each write at this address until the last register (IR10, the split-screen register) is accessed. The pointer then continues to point to the split-screen register. Upon power-up or a master reset command, the internal pointer is reset to point to the first register (IR0) of the initialization register group. The internal pointer can also be preset to any register of the group via the "load IR address pointer" command. These registers are write only and are used to specify parameters such as the system configuration, display format, cursor shape, and monitor timing. Register formats are shown in Figure 12 and described in the following paragraphs.

FIGURE 12 — INITIALIZATION REGISTER FORMATS (Page 1 of 3)

	7	6	5	4	3	2	1	0
IR0	Not Used	Scan Lines Per Character Row				Sync Select	Buffer-Mode Select	
		Non-Interlaced		Interlaced				
		0000 = 1 Line 0001 = 2 Lines 0010 = 3 Lines • • 1110 = 15 Lines 1111 = 16 Lines		0000 = Undefined 0001 = 5 Lines 0010 = 7 Lines • • 1110 = 31 Lines 1111 = Undefined		0 = VSYNC 1 = CSYNC	00 = Independent 01 = Transparent 10 = Shared 11 = Row	

FIGURE 12 — INITIALIZATION REGISTER FORMATS (Page 2 of 3)

IR1

7	6 5 4 3 2 1 0
Interlace Enable	Equalizing Constant
0 = Non-Interlace 1 = Interlace	0000000 = 1 $\overline{CCLK}$ 0000001 = 2 $\overline{CCLK}$ • • • 1111110 = 127 $\overline{CCLK}$ 1111111 = 128 $\overline{CCLK}$ Calculated from: $EC = 0.5\ (H_{ACT} + H_{FP} + HSYNC + H_{BP}) - 2(HSYNC)$

IR2

7	6 5 4 3	2 1 0
Not Used	Horizontal Sync Width	Horizontal Back Porch
	0000 = 2 $\overline{CCLK}$ 0001 = 4 $\overline{CCLK}$ • • 1110 = 30 $\overline{CCLK}$ 1111 = 32 $\overline{CCLK}$	000 = 1 $\overline{CCLK}$ 001 = 5 $\overline{CCLK}$ • • 110 = 25 $\overline{CCLK}$ 111 = 29 $\overline{CCLK}$

IR3

7 6 5	4 3 2 1 0
Vertical Front Porch	Vertical Back Porch
000 = 4 Scan Lines 001 = 8 Scan Lines • • 110 = 28 Scan Lines 111 = 32 Scan Lines	00000 = 4 Scan Lines 00001 = 6 Scan Lines • • 11110 = 64 Scan Lines 11111 = 66 Scan Lines

IR4

7	6 5 4 3 2 1 0
Character Blink Rate	Active Character Rows Per Screen*
0 = 1/16 VSYNC 1 = 1/32 VSYNC	0000000 = 1 Row 0000001 = 2 Rows • • • 1111110 = 127 Rows 1111111 = 128 Rows

*In interlace mode with odd total character rows per screen the last character row will be the programmed scan lines per character row minus one.

IR5

7	6 5 4 3 2 1 0
Active Characters Per Row	
	00000010 = 2 Characters 00000011 = 4 Characters • • 11111110 = 255 Characters 11111111 = 256 Characters

FIGURE 12 — INITIALIZATION REGISTER FORMATS (Page 3 of 3)

IR6

7	6	5	4	3	2	1	0
First Line of Cursor				Last Line of Cursor			
0000 = Scan Line 0 0001 = Scan Line 1 • • 1110 = Scan Line 14 1111 = Scan Line 15				0000 = Scan Line 0 0001 = Scan Line 1 • • 1110 = Scan Line 14 1111 = Scan Line 15			

IR7

7	6	5	4	3	2	1	0
Light Pen Line		Cursor Blink	Double Height Char.	Underline Position			
00 = Scan Line 3 01 = Scan Line 5 10† Scan Line 7 11† Scan Line 9		0 = No 1 = Yes	0 = No 1 = Yes	0000 = Scan Line 0 0001 = Scan Line 1 • • 1110 = Scan Line 14 1111 = Scan Line 15			

IR8

7	6	5	4	3	2	1	0
Display Buffer First Address LSBs							
H"000" = 0 H"001" = 1 • • H"FFE" = 4,094 H"FFF" = 4,095				NOTE: MSBs are in IR9[3:0]			

IR9

7	6	5	4	3	2	1	0
Display Buffer Last Address				Display Buffer First Address MSBs			
0000 = 1,023 0001 = 2,047 • • 1110 = 15,359 1111 = 16,383				See IR8			

IR10

7	6	5	4	3	2	1	0
Cursor Blink Rate	Split-Screen Interrupt Row						
0 = 1/16 VSYNC 1 = 1/32 VSYNC	0000000 = Row 0 0000001 = Row 1 • • 1111110 = Row 126 1111111 = Row 127						

3

SCAN LINES PER CHARACTER ROW (IR0[6:3]) – Both interlaced and non-interlaced scanning are supported by the PVTC. For interlaced mode, two different formats can be implemented, depending on the interconnection between the PVTC and the character generator (see IR1[7]). This field defines the number of scan lines used to compose a character row for each technique. As scanning occurs, the scan line count is output on the LA0-LA3 and LI pins.

VS/CS ENABLE (IR0[2]) – This bit selects either vertical sync pulses or composite sync pulses on the VSYNC/CSYNC output (pin 18). The composite sync waveform conforms to EIA RS170 standards, with the vertical interval composed of six equalizing pulses, six vertical sync pulses, and six more equalizing pulses.

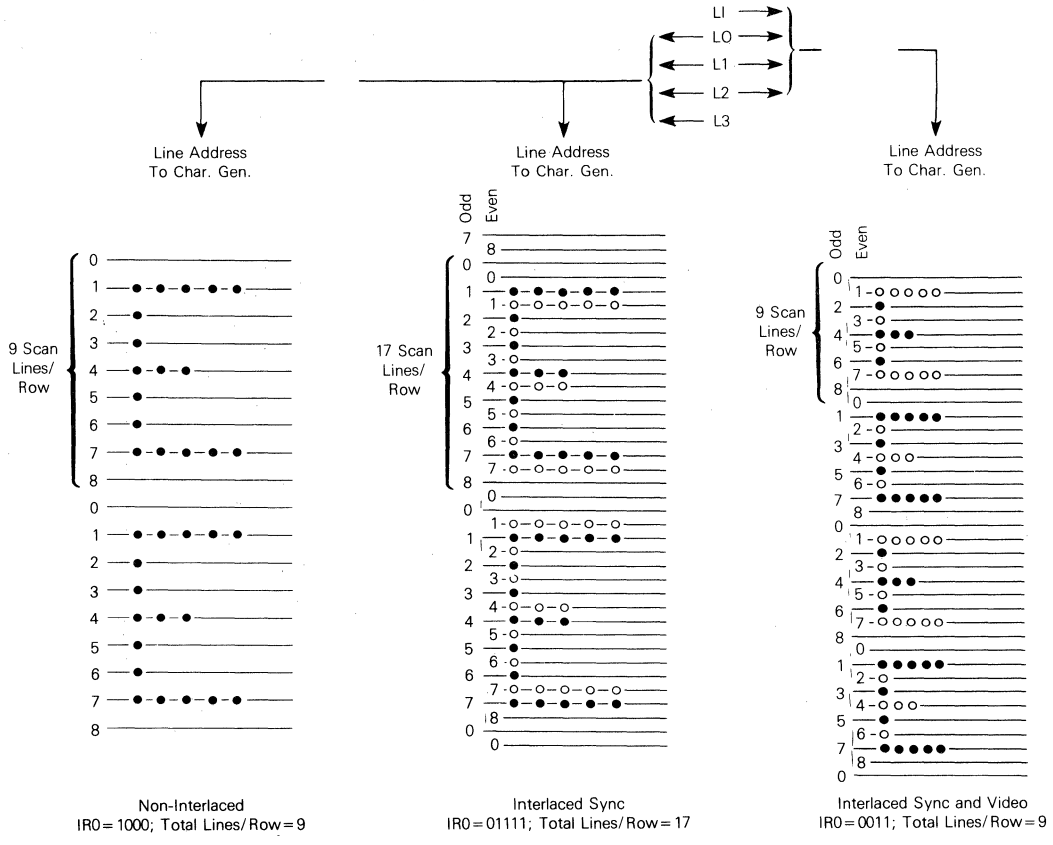
BUFFER MODE SELECT (IR0[1:0]) – Four buffer memory modes may be selectively enabled to accommodate the desired system configuration. See **SYSTEM CONFIGURATION.**

INTERLACE ENABLE (IR1[7]) – Specifies interlaced or non-interlaced timing operation. Two modes of interlaced operation are available, depending on whether L0-L3 or LI, L0-L2 are used as the line address for the character generator. The resulting displays are shown in Figure 13.

For "interlaced sync" operation, the same information is displayed in both odd and even fields, resulting in enhanced readability. The PVTC outputs successive line numbers in ascending order on the LA0-LA3 lines, one per scan line for each field.

The "interlaced sync and video" format doubles the character density on the screen. The PVTC outputs successive line numbers in ascending order on the LI, LA0-LA2 lines, one per scan line for each field, but alternates beginning the count with even and odd line numbers. This displays the odd field with even scan lines in even character rows and odd scan lines in odd character rows, and the even field with odd scan lines in even character rows and even scan lines on odd character rows. This provides balanced beam currents in the odd and even fields, thus minimizing character variations due to different loading of the CRT anode supply between fields.

FIGURE 13 – INTERLACED DISPLAY MODES



EQUALIZING CONSTANT (IR1[6:0]) — This field indirectly defines the horizontal front porch and is used internally to generate the equalizing pulses for the RS170 compatible CSYNC. The value for this field is the total number of character clocks ($\overline{CCLK}$) during a horizontal line period divided by two, minus two times the number of character clocks in the horizontal sync pulse:

$$EC = \frac{H_{ACT} + H_{FP} + H_{SYNC} + H_{BP}}{2} - 2(HSYNC)$$

The definition of the individual parameters is illustrated in Figure 14. The minimum value of H_{FP} is two character clocks.

Note that when using the attributes controller the blank pulse is delayed three $\overline{CCLK}$s relative to the HSYNC pulse.

HORIZONTAL SYNC PULSE WIDTH (IR2[6:3]) — This field specifies the width of the HSYNC pulse in $\overline{CCLK}$ periods.

HORIZONTAL BACK PORCH (IR2[2:0]) — This field defines the number of $\overline{CCLK}$s between the trailing edge of HSYNC and the trailing edge of BLANK.

VERTICAL FRONT PORCH (IR3[7:5]) — Programs the number of scan line periods between the rising edges of BLANK and VSYNC during a vertical retrace interval. The width of the VSYNC pulse is fixed at three scan lines.

VERTICAL BACK PORCH (IR3[4:0]) — This field determines the number of scan line periods between the falling edges of the VSYNC and BLANK outputs.

CHARACTER BLINK RATE (IR4[7]) — Specifies the frequency for the character blink attribute timing. The blink rate can be specified as 1/16 or 1/32 of the vertical field rate. The timing signal has a duty cycle of 75% and is multiplexed onto the DADD11/BLINK output at the falling edge of each BLANK.

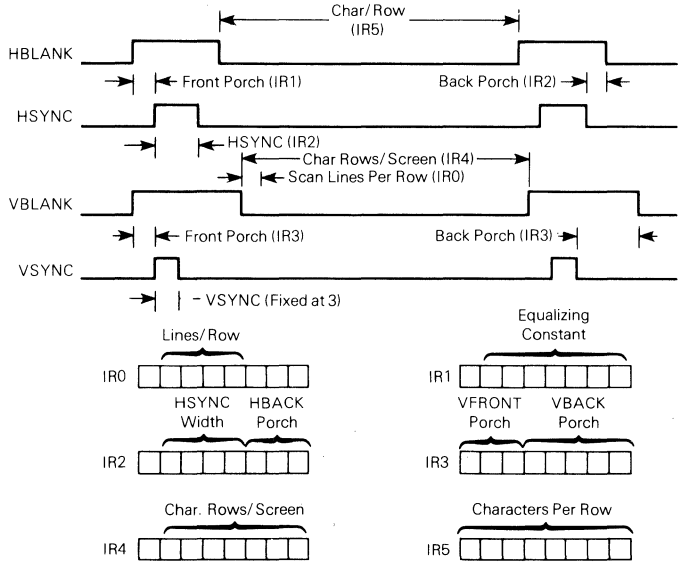
CHARACTER ROWS PER SCREEN (IR4[6:0]) — This field defines the number of character rows to be displayed. This value multiplied by the scan lines per character row, plus the vertical front and back porch values, and the vertical sync pulse width (three scan lines) is the vertical scan period in scan lines.

ACTIVE CHARACTERS PER ROW (IR5[7:0]) — This field determines the number of characters to be displayed on each row of the CRT screen. The sum of this value, the horizontal front porch, the horizontal sync width, and the horizontal back porch is the horizontal scan period is $\overline{CCLK}$s.

FIRST AND LAST SCAN LINE OF CURSOR (IR6[7:4] AND IR6[3:0]) — These two fields specify the height and position of the cursor on the character block. The "first" line is the topmost line when scanning from the top to the bottom of the screen.

3

FIGURE 14 — HORIZONTAL AND VERTICAL TIMING



LIGHT PEN LINE POSITION (IR7[7:6]) — This field defines which of four scan lines of the character row will be used for the light pen strike — through attribute by the MC2673 VAC. The timing signal is multiplexed onto the DADD9/LPL output during the falling edge of BLANK.

CURSOR BLINK ENABLE (IR7[5]) — This bit controls whether or not the cursor output pin will be blinked at the selected rate (IR10[7]). The blink duty cycle for the cursor is 50%.

DOUBLE HEIGHT CHARACTER ROW ENABLE (IR7[4]) — If enabled, the number of each scan line will be repeated twice in succession, causing the height of the character row to double. This bit can be changed at any time but will only become effective at the beginning of the character row following the time it is changed. This allows selected character rows to be of double height. The split-screen interrupt can be used to notify the CPU when the effectuate changes to this bit. For each double height row which replaces a normal row, one row count should be subtracted from the "character rows per screen" field (IR4) to maintain the same total number of scan lines per field.

UNDERLINE POSITION (IR7[3:0]) — This field defines which scan line of the character row will be used for the underline attributes by the attributes controller. The timing signal is multiplexed onto the DADD10/UL output during the falling edge of BLANK.

DISPLAY BUFFER FIRST ADDRESS (IR9[3:0] AND IR8[7:0]) AND DISPLAY BUFFER LAST ADDRESS (IR9[7:4]) — These two fields define the area within the buffer memory where the display data will reside. When the data at the "display buffer last address" is displayed, the PVTC will wrap-around and obtain the data to be displayed at the next screen position from the "display buffer first address" If "last address" is the end of a character row and a new screen start address has been loaded into the screen start register, or if "last address" is the last character position of the screen, the next data is obtained from the address contained in the screen start register.

Note that there is no restriction in displaying data from other areas of the addressable memory. Normally, the area between these two bounds is used for data which can be overwritten (e.g., as a result of scrolling), while data that is not to be overwritten would be contained outside these bounds and accessed by means of the split-screen interrupt feature of the PVTC.

CURSOR BLINK RATE (IR10[7]) — The cursor blink rate can be specified at 1/16 or 1/32 of the vertical scan frequency. Blink is effective only if blink is enabled by IR7[5].

SPLIT-SCREEN INTERRUPT (IR10[6:0]) — The split-screen interrupt can be used to provide special screen effects such as a row of double height characters or to change the normal addressing sequence of the display memory. The contents of this field is compared, in real time, to the current character row number. Upon a match, the PVTC sets the split-screen status bit, and issues an interrupt request if so programmed. The status change/interrupt request is made at the beginning of scan line zero of the split-screen character row.

TIMING CONSIDERATIONS

Normally, the contents of the initialization registers are not changed during operation. However, this may be necessary to implement special display features such as multiple cursors, smooth scrolling, horizontal scrolling, and double height character rows. Table 2 describes the timing details for these registers which should be considered when implementing these features.

TABLE 2 — TIMING CONSIDERATIONS

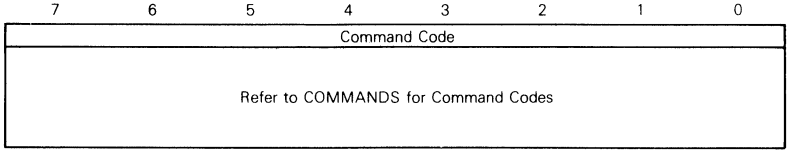
Parameter	Timing Considerations
Field Line of Cursor Last Line of Cursor Light Pen Line Underline	These parameters must be established at a minimum of two characters times prior to their occurrence.
Double Height Characters	Set/reset during the character row prior to the row which is to be/not to be double height.
Cursor Blink Cursor Blink Rate Character Blink Rate	New values become effective within one field after values are changed.
Split-Screen Interrupt Row	Change anytime prior to line zero of desired row.
Character Rows Per Screen	Change only during vertical blanking period.
Vertical Front Porch	Change prior to first line of VFP.
Vertical Back Porch	Change prior to fourth line after VSYNC.
Screen-Start Register	Change prior to the horizontal blanking interval of the last line of character row before row where new value is to be used.

DISPLAY CONTROL REGISTERS

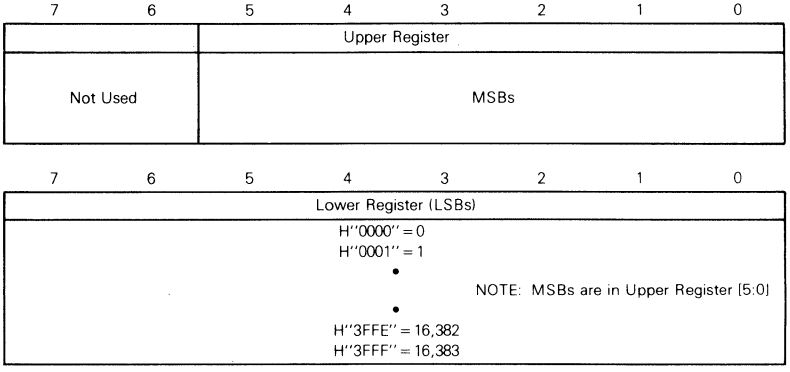
There are nine registers in this group, each with an individual address. Their formats are illustrated in Figure 15. The command register is used to invoke one of 16 possible PVTC commands as described in COMMANDS. The remaining registers in the group store address values which specify the cursor and buffer pointer locations, the location of the first character to be displayed on the screen, and the location of a light pen "hit". With the exception of the light pen register, the user initializes these registers after powering on the system and changes their values to control the data which is displayed.

FIGURE 15 — DISPLAY CONTROL REGISTER FORMATS

(a) Command Register (Write Only)



(b) Screen Start Registers (Read and Write), Cursor Address Registers (Read and Write), Pointer Address Register (Write Only), and Light Pen Address Register (Read Only)

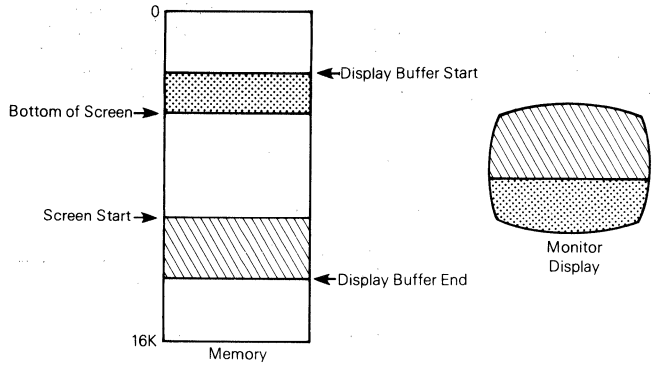


SCREEN-START REGISTERS

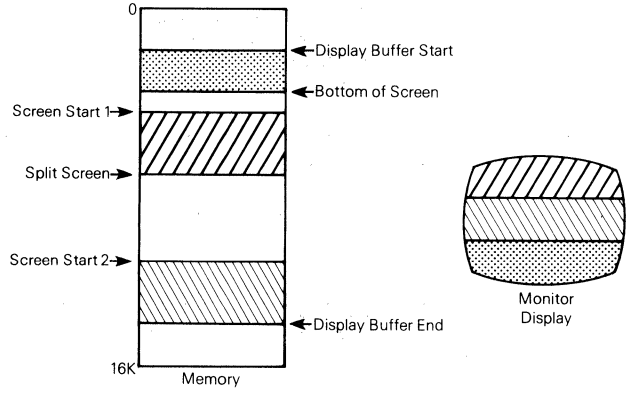
The screen-start registers contain the address of the first character of the first row (upper left corner of the active display). At the beginning of the first scan line of the first row, this address is transferred to the row-start register (RSR) and into the memory-address counter (MAC). The counter is then advanced sequentially at the character rate the number of times programmed into the active characters-per-row register (IR5) thus reaching the address of the last character of the row plus one. At the beginning of each subsequent scan line of the first row, the MAC is reloaded from the RSR and the above sequence is repeated. At the end of the last scan line of the first row, the contents of the MAC is loaded into the RSR to serve as the starting memory address for the second character row. This process is repeated for the programmed number of rows per screen. Thus, the data in the display memory is displayed sequentially starting from the address contained in the screen start register. After the ensuing vertical retrace interval, the entire process repeats again.

The sequential operation described above will be modified upon the occurrence of either of two events. First, if during the incrementing of the memory address counter the "display buffer last address" (IR9[7:4]) is reached, the MAC will be loaded from the "display buffer first address" register (IR9[3:0]), (IR8[7:0]) at the next character clock. Sequential operation will then resume starting form this address. This wraparound operation allows portions of the display buffer to be used for purposes other than storage of displayable data and is completely automatic without any CPU intervention (see Figure 16a).

FIGURE 16 — DISPLAY ADDRESSING OPERATION



(a) Display Memory Wraparound



(b) Display Memory Split Screen With Wraparound

The sequential row-to-row addressing can also be modified under CPU control. If the contents of the screen-start register (upper, lower, or both) are changed during any character row (say row "n"), the starting address of the next character row (row "n+1") will be the next value of the screen-start register and addressing will continue sequentially from there. This allows features such as split-screen operation, partial scroll, or status line display to be implemented. The split-screen interrupt feature of the PVTC is useful in controlling this type of operation. Note that in order to obtain the correct screen display, the screen-start register must be reloaded with the original value prior to the end of the vertical retrace. See Figure 16b.

During vertical blanking the address counter operation is modified by stopping the automatic load of the contents of the RSR into the counter, thereby allowing the address outputs to free-run. This allows dynamic memory refresh to occur during the vertical retrace interval. The refresh addressing starts at the last address displayed on the screen and increments by one for each character clock during the retrace interval. If the display buffer last address is encountered refreshing continues from the display buffer first address.

CURSOR ADDRESS REGISTERS

The contents of these registers define the buffer memory address of the cursor. If enabled, the cursor output will be asserted when the memory address counter matches the value of the cursor address registers. The cursor address registers may be read or written by the CPU or incremented via the "increment cursor address" command. In independent buffer mode, these registers define a buffer memory address for PVTC controlled access in response to "read/write at cursor with/without increment" commands, or the first address to be used in executing the "write for cursor to pointer" command.

DISPLAY POINTER ADDRESS REGISTERS

These registers define a buffer memory address for PVTC controlled accesses in response to "read/write at pointer" commands. They also define the last buffer memory address to be written for the "write from cursor to pointer" command.

LIGHT PEN ADDRESS REGISTERS

If the light pen input is enabled, these registers are used to

store the current character address upon receipt of a light pen strobe input. Several sources of delay between the display of a character upon the screen and the receipt of a light pen hit can be expected to exist in a system environment. These delays include address pipelining in the character generation circuits, delays in the video generation circuits, and delays in the light detection circuitry itself. These delays cause the value stored in the light pen register to differ from the actual address of the character at which the light pen hit actually was detected. Software must be used to correct this condition.

INTERRUPT/STATUS REGISTERS

The interrupt and status registers provide information to the CPU to allow it to interface with the PVTC to effect desired changes to implement various display operations. The interrupt register provides information on five possible interrupting conditions, as shown in Figure 17. These conditions may be selectively enabled or disabled (masked) from causing interrupts by certain PVTC commands. An interrupt condition which is enabled (mask bit equal to one) will cause the $\overline{\text{INTR}}$ output to be asserted and will cause the corresponding bit in the interrupt register to be set upon occurrence of interrupt condition. An interrupt condition which is disabled (mask bit equal to zero) has no effect on either the $\overline{\text{INTR}}$ output or the interrupt register.

The status register provides six bits of status information; the five possible interrupting conditions plus the NOT BUSY bit. For this register, however, the contents are not effected by the state of the mask bits.

Descriptions of each interrupt/status register bit follows. Unless otherwise indicated, a bit, once set, will remain set until reset by the CPU by issuing a "reset interrupt/status bits" command. The bits are also reset by a "master reset" command and upon power-up.

RDFLG (SR[5]) — This bit is present in the status register only. A zero indicates that the PVTC is currently executing the previously issued command. A one indicates that the PVTC is ready to accept a new command.

VBLANK (I/SR[4]) — Indicates the beginning of a vertical blanking interval, is set to a one at the beginning of the first scan line of the vertical front porch.

LINE ZERO (I/SR[3]) — Is set to a one at the beginning of the first scan line (line zero) of each active character row.

SPLIT SCREEN (I/SR[2]) — This bit is set when a match occurs between the current character row number and the value contained in the split-screen interrupt register, IR10[6:0]. The equality condition is only checked at the beginning of line zero of each character row. This bit is reset when either of the screen-start registers is loaded by the CPU.

READY (I/SR[1]) — Certain PVTC commands affect the display and may require the PVTC to wait for a blanking interval before enacting the command. This bit is set to one when execution of the command has been completed. No command should be invoked until the prior command is completed.

LIGHT PEN (I/SR[0]) — A one indicates that a light pen hit has occurred and that the contents of the light pen register have been updated. This bit will be reset when either of the light pen registers is read.

COMMANDS

The PVTC commands are divided into two classes: the instantaneous commands, which are executed immediately after they are invoked, and the delayed commands which may need to wait for a blanking interval prior to their execution. Command formats are shown in Table 3. The commands are asserted by performing a write operation to the command register with the appropriate bit pattern as the data byte.

FIGURE 17 — INTERRUPT AND STATUS REGISTER FORMAT

7	6	5	4	3	2	1	0
Not Used Always Read as Zero		RDFLG	VBLANK	Line Zero	Split Screen	Ready	Light Pen
		0 = Busy 1 = Ready	0 = No 1 = Yes	0 = No 1 = Yes	0 = No 1 = Yes	0 = Busy 1 = Ready	0 = No 1 = Yes

TABLE 3 — PVTC COMMAND FORMATS

D7	D6	D5	D4	D3	D2	D1	D0	Hex	Command
Instantaneous Commands									
0	0	0	0	0	0	0	0		Master Reset
0	0	0	1	V	V	V	V		Load IR Pointer with Value V (V = 0 to 10)
0	0	1	d	d	d	1	0*		Disable Light Pen
0	0	1	d	d	d	1	1*		Enable Light Pen
0	0	1	d	1	N	d	0*		Display Off — Float DADD Bus If N = 1
0	0	1	d	1	N	d	1*		Display On — Next Field (N = 1) or Scan Line (N = 0)
0	0	1	1	d	d	d	0*		Cursor Off
0	0	1	1	d	d	d	1*		Cursor On
0	1	0	N	N	N	N	N		Reset Interrupt/Status — Bit Reset where N = 1
1	0	0	N	N	N	N	N		Disable Interrupt — Disable where N = 1
0	1	1	N	N	N	N	N		Enable Interrupt — Enables Interrupts and Resets the Corresponding Interrupt/Status Bits where N = 1
			V B	L Z	S S	R D	L P		
Delayed Commands									
1	0	1	0	0	1	0	0	A4	Reset at Pointer Address
1	0	1	0	0	0	1	0	A2	Write at Pointer Address
1	0	1	0	1	0	0	1	A9	Increment Cursor Address
1	0	1	0	1	1	0	0	AC	Read at Cursor Address
1	0	1	0	1	0	1	0	AA	Write at Cursor Address
1	0	1	0	1	1	0	1	AD	Read at Cursor Address and Increment Address
1	0	1	0	1	0	1	1	AB	Write at Cursor Address and Increment Address
1	0	1	1	1	0	1	1	BB	Write from Cursor Address to Pointer Address

*Any combination of these three commands is valid.
d = Don't Care

3

INSTANTANEOUS COMMANDS

The instantaneous commands are executed immediately after the trailing edge of the write pulse during which the command is issued. These commands do not affect the state of the RDFLG or READY interrupt/status bits. However, a command should not be invoked if the RDFLG bit is low.

MASTER RESET

This command initializes the PVTC and may be invoked at any time to return the PVTC to its initial state. Upon power-up, two successive master reset commands must be applied to release the PVTC's internal power on circuits. In transparent and shared buffer modes, the CNTRL1 input must be high when the command is issued. The command causes the following:

1. VSYNC and HSYNC are driven low for the duration of reset and BLANK goes high. BLANK remains high until a "display on" command is received.
2. The interrupt and status bits and masks are set to zero, except for the RDFLG flag which is set to a one.
3. The transparent mode, cursor off, display off, and light pen disable states are set.
4. The initialization register pointer is set to address IR0.

LOAD IR ADDRESS

This command is used to preset the initialization register pointer with the value "V" defined by D3-D0. Allowable values are 0 to 10.

ENABLE LIGHT PEN

After invoking this command, receipt of a light pen strobe input will cause the light pen register to be loaded with the current buffer memory address and the corresponding interrupt and status flag to be set. Once loaded, further loads are inhibited until either one of the light pen registers are read or a reset function is performed.

DISABLE LIGHT PEN

Light pen hits will not be recognized.

DISPLAY OFF

Asserts the BLANK output. The DADD0 through DADD13 display address bus outputs may be optionally placed in the high-impedance state by setting bit 2 to a one when invoking the command.

DISPLAY ON

Restores normal blanking operation either at the beginning of the next field (bit 2 = 1) or at the beginning of the next scan line (bit 2 = 0). Also returns the DADD0-DADD13 drivers to their active state.

CURSOR OFF

Disables cursor operation. Cursor output is placed in the low state.

CURSOR ON

Enables normal cursor operation.

RESET INTERRUPT/STATUS BITS

This command resets the designated bits in the interrupt and status registers. The bit positions correspond to the bit positions in the registers:

Bit 0 – Light Pen
Bit 1 – Ready
Bit 2 – Split Screen
Bit 3 – Line Zero
Bit 4 – Vertical Blank

DISABLE INTERRUPTS

Sets the interrupt mask to zeros for the designated conditions, thus disabling these conditions from asserting the $\overline{INTR}$ output. Bit position correspondence is as above.

ENABLE INTERRUPTS

Resets the selected interrupt and status register bits and writes the associated interrupt mask bits to a one. This enables the corresponding conditions to assert the $\overline{INTR}$ output. Bit position correspondence is as above.

DELAYED COMMANDS

This group of commands is utilized for the independent buffer mode of operation, although the "increment cursor" command can also be used in other modes. With the exception of the "write from cursor to pointer" and "increment cursor" commands, all the commands of this type will be executed immediately or will be delayed depending on when the command is invoked. If invoked during the active screen time, the command is executed at the next horizontal blanking blanking interval. If invoked during a vertical retrace interval or a "display off" state, the command is executed immediately.

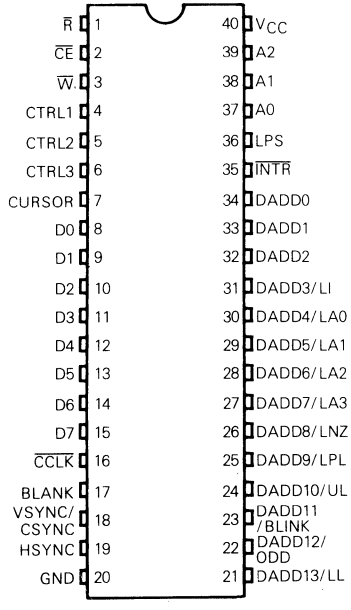
MECHANICAL DATA

3

ORDERING INFORMATION (T_A = 0°C to 70°C)

Package Type	Frequency	Order Number
Plastic P Suffix	2.7 MHz	MC2672B3P
	4.0 MHz	MC2672B4P

PIN ASSIGNMENTS



MC2674

Advance Information

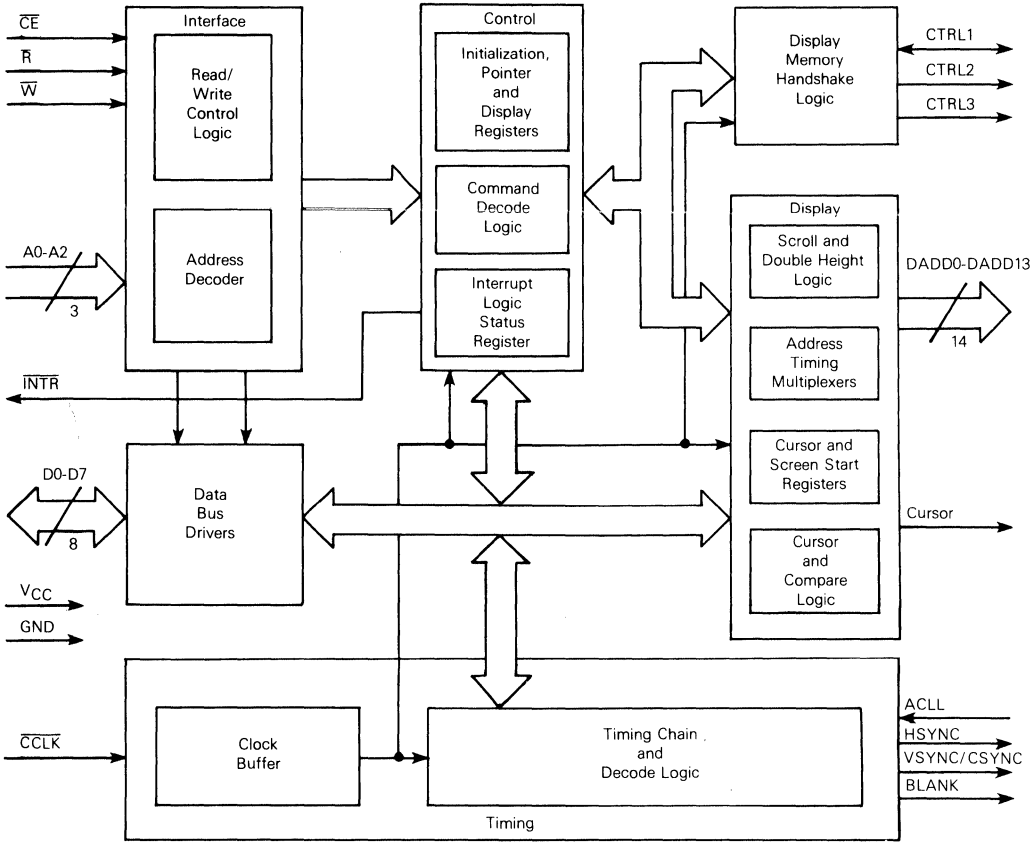
Advanced Video Display Controller (AVDC)

The MC2674 advanced video display controller (AVDC) is a programmable device designed for use in CRT terminals and display systems that employ raster-scan techniques. The AVDC generates the vertical and horizontal timing signals necessary for the display of interlaced or non-interlaced data on a CRT monitor. It provides consecutive addressing to a user specified display buffer memory domain and controls the CPU-display buffer interface for various buffer configuration modes. A variety of operating modes, display formats, and timing profiles can be implemented by programming the control registers in the AVDC.

A minimum CRT terminal system configuration consists of an AVDC, a keyboard controller, an asynchronous communications interface adapter, character ROM, and an attributes controller. Other necessary parts of the system are a single-chip microcomputer such as the MC6809, display buffer RAM, and a small amount of TTL for miscellaneous address decoding, interface, and control. System complexity can be enhanced by upgrading the microprocessor and expanding via the system address and data buses.

- 4 MHz Character Rate
- 1 to 256 Characters Per Row
- 1 to 16 Raster Lines Per Character Row
- Bit Mapped Graphics Mode
- Programmable Horizontal and Vertical Sync Generators
- Interlaced or Non-Interlaced Operation
- Up to 64K RAM Address for Multiple-Page Operation
- Readable, Writeable, and Incrementable Cursor
- Programmable Cursor Size and Blink
- AC Line Lock
- Automatic Wraparound of RAM
- Automatic Split Screen
- Automatic Bidirectional Soft Scrolling
- Programmable Scan Line Increment
- Row Table Addressing Mode
- Double Height Tops and Bottoms
- Double Width Control Output
- Selectable Buffer Interface Modes
- Dynamic RAM Refresh
- Completely TTL Compatible
- Single +5-Volt Power Supply
- Power-On Reset Circuit
- Applications Include: CRT Terminals, Word Processing Systems, Small Business Computers, and Home Computers

BLOCK DIAGRAM



3

ABSOLUTE MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range	T_A	0 to 70	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range GND≤(V_{in} or V_{out})≤V_{CC}. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (e.g., either GND or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Rating
Thermal Resistance Plastic Package	θ_{JA}	50	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part, K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A

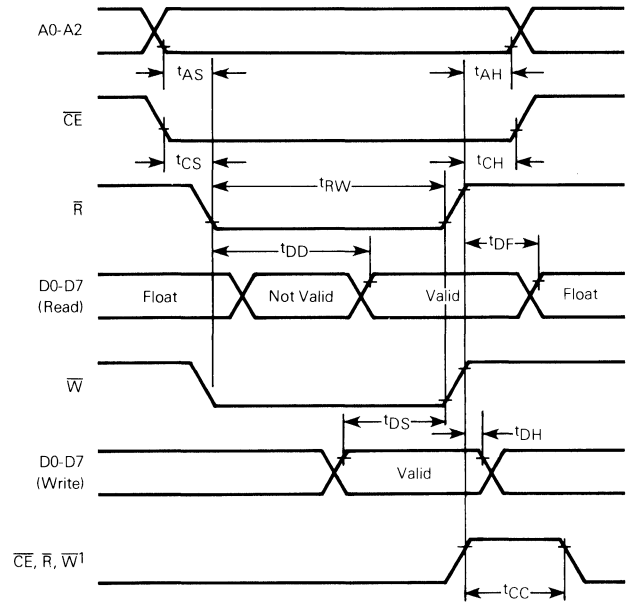
DC ELECTRICAL CHARACTERISTICS (T_A = 0°C to 70°C, V_{CC} = 5.0 V ± 5%)

Parameter	Symbol	Min	Max	Unit
Input Low Voltage	V_{IL}	−0.3	0.8	V
Input High Voltage	V_{IH}	2.0	V_{CC}	V
Output Low Voltage (I_{OL} = 2.4 mA)	V_{OL}	–	0.4	V
Output High Voltage (Except $\overline{INTR}$ Output) (I_{OH} = −200 μA)	V_{OH}	2.4	–	V
Input Leakage Current (V_{in} = 0 to V_{CC})	I_{in}	−10	10	μA
Hi-Z (Off-State) Leakage Current (V_{CC} = 5.25 V, V_{in} = 0.4 to 2.4 V)	I_{TSI}	−10	10	μA
$\overline{INTR}$ Open-Drain Output Leakage Current (V_O = 0 to V_{CC})	I_{OD}	–	10	μA
Internal Power Dissipation (Measured at T_A = 0°C)	P_{INT}	–	800	mW

AC ELECTRICAL CHARACTERISTICS — BUS TIMING ($T_A = 0°C$ to $70°C$, $V_{CC} = 5\ V \pm 5\%$)

Parameter	Symbol	2.7 MHz		4.0 MHz		Unit
		Min	Max	Min	Max	
A0-A2 Setup Time to $\overline{W}$, $\overline{R}$ Low	t_{AS}	30	–	30	–	ns
A0-A2 Hold Time from $\overline{W}$, $\overline{R}$ High	t_{AH}	0	–	0	–	ns
$\overline{CE}$ Setup Time to $\overline{W}$, $\overline{R}$ Low	t_{CS}	0	–	0	–	ns
$\overline{CE}$ Hold Time from $\overline{W}$, $\overline{R}$ High	t_{CH}	0	–	0	–	ns
$\overline{W}$, $\overline{R}$ Pulse Width	t_{RW}	250	–	200	–	ns
Data Valid after $\overline{R}$ Low	t_{DD}	–	200	–	200	ns
Data Bus Floating after $\overline{R}$ High	t_{DF}	–	100	–	100	ns
Data Setup Time to $\overline{W}$ High	t_{DS}	150	–	150	–	ns
Data Hold Time from $\overline{W}$ High	t_{DH}	10	–	5	–	ns
High Time from $\overline{CE}$ to $\overline{CE}$	t_{CC}					
Consecutive Commands		t_{CCP}	–	t_{CCP}	–	ns
Other Accesses		300	–	300	–	ns

BUS TIMING DIAGRAM



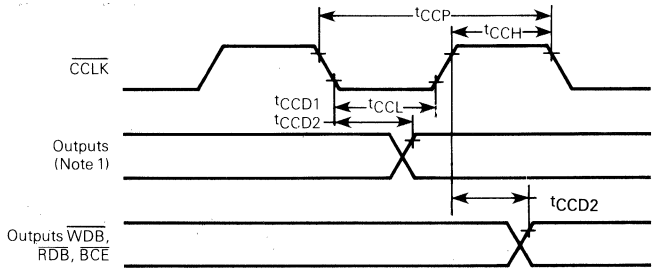
NOTES:
1. Any two must be high for t_{CC}.
2. All voltage measurements are referenced to ground. All time measurements are at the 0.8 V to 2.0 V level for inputs and outputs. Input levels are 0.4 V to 2.4 V.

AC ELECTRICAL CHARACTERISTICS — CHARACTER CLOCK ($\overline{CCLK}$) TIMING ($T_A = 0°C$ to $70°C$, $V_{CC} = 5\ V \pm 5\%$)

Parameter	Symbol	2.7 MHz Min	2.7 MHz Max	4.0 MHz Min	4.0 MHz Max	Unit
$\overline{CCLK}$ Period	t_{CCP}	370	10000	250	10000	ns
$\overline{CCLK}$ High Time	t_{CCH}	125	—	100	—	ns
$\overline{CCLK}$ Low Time	t_{CCL}	125	—	100	—	ns
Output Delay Time from $\overline{CCLK}$ Edge						
DADD0-13, MBC	t_{CCD1}	40	175	40	150	ns
BLANK, HSYNC, VSYNC/CSYNC, CURSOR, $\overline{BEXT}$, $\overline{BREQ}$, $\overline{BACK}$, $\overline{BCE}$, $\overline{WDB}$, $\overline{RDB}$*	t_{CCD2}	40	225	40	200	ns

* $\overline{BCE}$, $\overline{WDB}$, and $\overline{RDB}$ delays track each other within 10 nanoseconds. Also, these output delays will tend to follow direction (minimum/maximum) of DADD0-DADD13 delays.

CCLK TIMING DIAGRAM



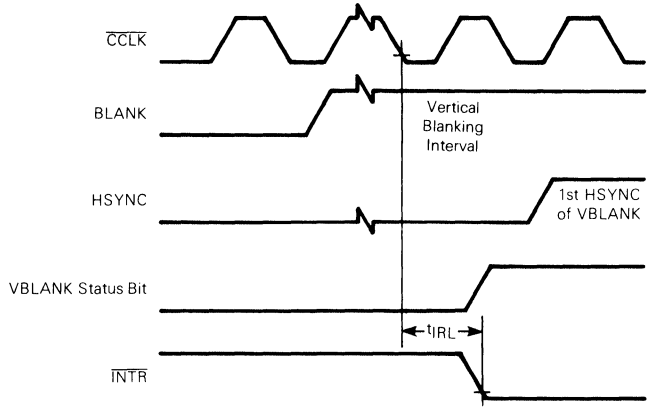
NOTES:
1. DADD0-DADD13, BLANK, HSYNC, CSYNC/VSYNC, CURSOR, $\overline{BEXT}$, $\overline{BREQ}$, $\overline{BCE}$, MBC, BACK.
2. $\overline{BCE}$ changes state on both $\overline{CCLK}$ edges.
3. All voltage measurements are referenced to ground. All time measurements are at the 0.8 V to 2.0 V level for inputs and outputs. Input levels are 0.4 V to 2.4 V.

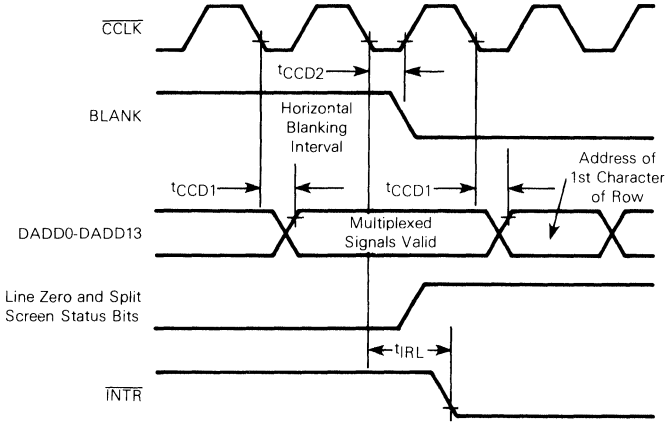
AC ELECTRICAL CHARACTERISTICS – OTHER TIMING ($T_A = 0°C$ to $70°C$, $V_{CC} = 5\ V \pm 5\%$)

Parameter	Symbol	2.7 MHz Min	2.7 MHz Max	4.0 MHz Min	4.0 MHz Max	Unit
READY/RDFLG low from $\overline{W}$ High*	t_{RDL}	—	$t_{CCP}+30$	—	$t_{CCP}+30$	ns
$\overline{BACK}$ High from $\overline{PBREQ}$ Low	t_{BAK}	—	225	—	200	ns
$\overline{BEXT}$ High from $\overline{PBREQ}$ High	t_{BXT}	—	225	—	200	ns
$\overline{INTR}$ Low from $\overline{CCLK}$ Low	t_{IRL}	—	225	—	200	ns
$\overline{INTR}$ High from $\overline{W}$, $\overline{R}$ High*	t_{IRH}	—	600	—	600	ns
ACLL from HSYNC	t_{AC}	$3\times t_{CCP}$	—	$3\times t_{CCP}$	—	ns

*Timing is illustrated and specified referenced to $\overline{W}$ and $\overline{R}$ inputs. Device may also be operated with $\overline{CE}$ as the "strobing" input. In this case, all timing specifications apply referenced to falling and rising edges of $\overline{CE}$.

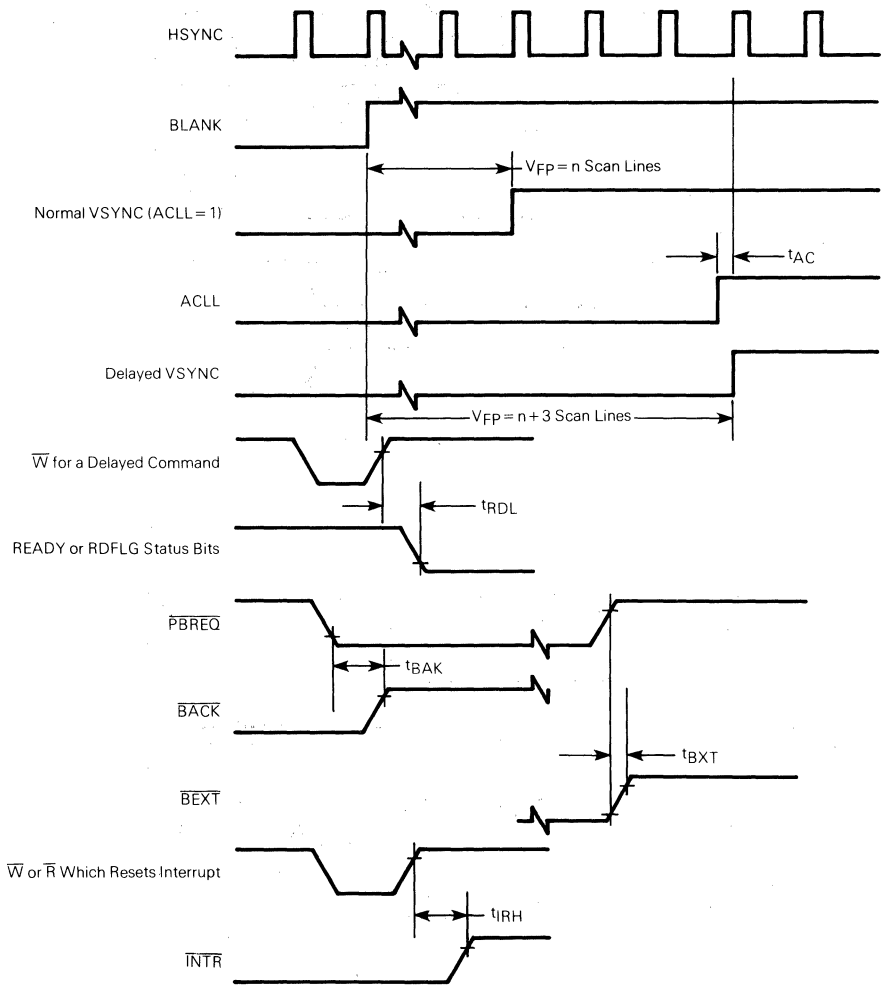
OTHER TIMING DIAGRAMS (Sheet 2 of 2)





NOTE: All voltage measurements are referenced to ground. All time measurements are at the 0.8 V to 2.0 V level for inputs and outputs. Input levels are 0.4 V to 2.4 V.

OTHER TIMING DIAGRAMS (Sheet 2 of 2)

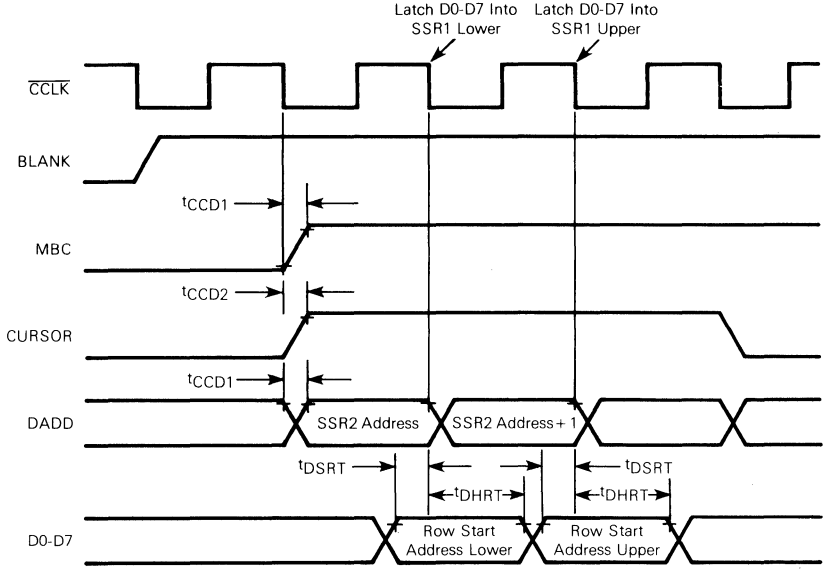


NOTE: All voltage measurements are referenced to ground. All time measurements are at the 0.8 V to 2.0 V level for inputs and outputs. Input levels are 0.4 V to 2.4 V.

AC ELECTRICAL CHARACTERISTICS — ROW TABLE INPUT TIMING ($T_A = 0°C$ to $70°C$, $V_{CC} = 5\ V \pm 5\%$)

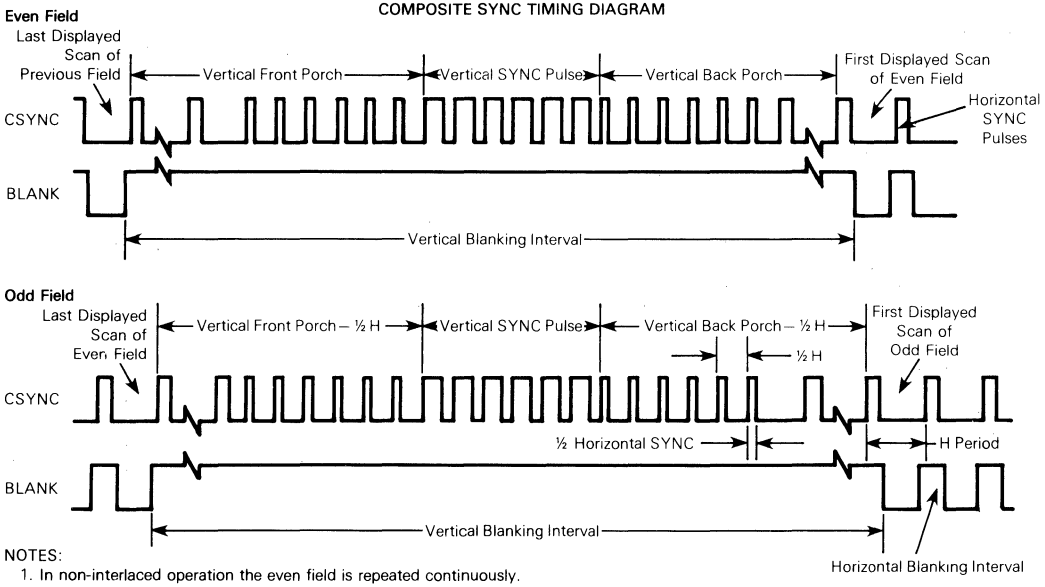
Parameter	Symbol	2.7 MHz		4.0 MHz		Unit
		Min	Max	Min	Max	
Data Setup Time to $\overline{CCLK}$ Low	t_{DSRT}	100	–	60	–	ns
Data Hold Time from $\overline{CCLK}$ Low	t_{DHRT}	60	–	60	–	ns

ROW TABLE FETCH I/O TIMING DIAGRAM



NOTE: All voltage measurements are referenced to ground. All time measurements are at the 0.8 V to 2.0 V level for inputs and outputs. Input levels are 0.4 V to 2.4 V.

COMPOSITE SYNC TIMING DIAGRAM



NOTES:
1. In non-interlaced operation the even field is repeated continuously.
2. In interlaced operation the even field alternates with the odd field.

SIGNAL DESCRIPTION

The input and output signals for the AVDC are described in the following paragraphs.

ADDRESS LINES (A0-A2)

These input lines are used to select AVDC internal register for read/write operations and for commands.

DATA BUS (D0-D7)

The 8-bit bidirectional three-state data bus controls all data, command, and status transfers between the CPU and the AVDC. Bit 0 is the least significant bit and bit 7 is the most significant bit. The direction of the transfer is controlled by the read ($\overline{R}$) and write ($\overline{W}$) inputs when chip enable ($\overline{CE}$) input is low. When the $\overline{CE}$ input is high, the data bus is in the three-state condition.

READ STROBE ($\overline{R}$)

This pin is an active low input. A low on this pin while $\overline{CE}$ is low causes the contents of the register selected by the address lines to be placed on the data bus. The read cycle begins on the leading (falling) edge of $\overline{R}$.

WRITE STROBE ($\overline{W}$)

This is an active low input. A low on this pin while $\overline{CE}$ is also low causes the contents of the data bus to be transferred to the register selected by the address lines. The transfer occurs on the trailing (rising) edge of $\overline{W}$.

CHIP ENABLE ($\overline{CE}$)

This is an active low input. When low, data transfers between the CPU and the AVDC are enabled on the data bus as controlled by the write strobe, read strobe, and address lines. When $\overline{CE}$ is high, effectively, the AVDC is isolated from the data bus and D0-D7 are placed in the three-state condition.

CHARACTER CLOCK ($\overline{CCLK}$)

This input is the timing signal derived from the video dot clock which is used to synchronize the AVDC's timing functions.

HORIZONTAL SYNC (HSYNC)

This active high output provides video horizontal sync pulses. The timing parameters are programmable.

VERTICAL SYNC/COMPOSITE SYNC (VSYNC/CSYNC)

A control bit selects either vertical or composite sync pulses on this active high output. When CSYNC is selected, equalization pulses are included. The timing parameters are programmable.

BLANK (BLANK)

This active high output defines the horizontal and vertical borders of the display. Display control signals which are output on display addresses DADD0 and DADD3 through DADD13 are valid on the trailing edge of BLANK.

CURSOR GATE (CURSOR)

This output becomes active for a specified number of scan lines when the address continued in the cursor register matches the address output on DADD0 through DADD13 for displayable character addresses. The first and last lines of the cursor and a blink option are programmable. When the row table addressing mode is enabled, this output is active for a portion of the blanking interval prior to the first scan line of a character row, while the AVDC is fetching the starting address for that row.

INTERRUPT REQUEST ($\overline{INTR}$)

This is an open-drain output which supplies an active low interrupt request from any of five maskable sources. This pin is inactive after a power-on reset or a master reset command.

AC LINE LOCK (ACLL)

If this input is low after the programmed vertical front porch interval, the vertical front porch will be lengthened by increments of horizontal scan line times until this input goes high.

HANDSHAKE CONTROL 1 (CTRL1)

In independent mode, provides an active low write data buffer ($\overline{WDB}$) output which strobes data from the interface latch into the display memory. In transparent and shared modes, this is an active low processor bus request ($\overline{PBREQ}$) input which indicates that the CPU desires to access the display memory.

HANDSHAKE CONTROL 2 (CTRL2)

In independent mode, provides an active low read data buffer ($\overline{RDB}$) output which strobes data from the display memory into the interface latch. In transparent and shared modes, this is an active low bus external enable ($\overline{BEXT}$) output which indicates that the AVDC has relinquished control of the display memory (DADD0-DADD13 are in the three-state condition) in response to a CPU bus request. $\overline{BEXT}$ also goes low in response to a 'display off and float DADD' command. In row buffer mode, it is an active low bus request ($\overline{BREQ}$) output which halts the CPU during a line DMA.

HANDSHAKE CONTROL 3 (CTRL3)

In independent mode, provides the active low buffer chip enable ($\overline{BCE}$) signal to the display memory. In transparent and shared modes, provides an active low bus acknowledge ($\overline{BACK}$) output which serves as a ready signal to the CPU in response to a processor bus request. In row buffer mode, this is an active high memory bus control (MBC) output which configures the system for the DMA transfer of one row of character codes from system memory to the row display buffer.

DISPLAY ADDRESS (DADD0-DADD13)

These outputs are used by the AVDC to address up to 16K of display memory directly, or to 64K of memory by demultiplexing DADD14 and DADD15. These outputs are floated at various times depending on the buffer mode. Various control

signals are multiplexed on DADD0 through DADD13 and are valid at the trailing edge of BLANK. The following paragraphs describes the control signals.

LINE GRAPHICS (DADD0/LG) – This is the output which denotes bit-mapped graphics mode.

DISPLAY ADDRESS 14 (DADD1/DADD14) – This is the multiplexed address bit used to extend addressing to 64K.

DISPLAY ADDRESS 15 (DADD2/DADD15) – This is the multiplexed address bit used to extend addressing to 64K.

LAST ROW (DADD3/LR) – This is the output which indicates the last active character row of each field.

LINE ADDRESS (DADD4-DADD7/LA0-LA3) – These outputs provide the number of the current scan line count for each character row.

FIRST LINE (DADD8/FL) – This output is asserted during the blanking interval just prior to the first scan line of each character row.

DOUBLE WIDTH (DADD9/DW) – This output denotes a double width character row.

UNDERLINE (DADD10/UL) – This output is asserted during the blanking interval just prior to the scan line which matches the programmed underline position (line 0 through 15).

BLINK FREQUENCY (DADD11/BLINK) – Blink frequency provides an output divided down from the vertical sync rate.

ODD FIELD (DADD12/ODD) – This active high signal is asserted before each scan line of the odd field when interlace is specified. Replaces DADD4/LA0 as the least significant line address for interlaced sync and video applications.

LAST LINE (DADD13/LL) – This output is asserted during the blanking interval just prior to the last scan line of each character row.

V_{CC} AND GND

Power is supplied to the AVDC using these two pins. V_{CC} is the +5 volts ±5% power input and GND is the ground connection.

FUNCTIONAL DESCRIPTION

As shown in the block diagram, the AVDC contains the following major blocks: data bus buffer, interface logic, operation control, timing, display control, and buffer control. The major blocks are described in the following paragraphs.

DATA BUS BUFFER

The data bus buffer provides the interface between the external and internal data buses. It is controlled by the operation control block to allow read and write operations to take place between the controlling CPU and the AVDC.

INTERFACE LOGIC

The interface logic contains address decoding and read and write circuits to permit communications with the microprocessor via the data buffer. The functions performed by the CPU read and write operations are shown in Table 1.

OPERATION CONTROL

The operation control section decodes configuration and operation commands from the CPU and generates appropriate signals to other internal sections to control the overall device operation. It contains the timing and display registers which configure the display format and operating mode, the interrupt logic, and the status register which provides operational feedback to the CPU.

TIMING

The timing section contains the counters and decoding logic necessary to generate the monitor timing outputs and to control the display format. These timing parameters are selected by programming of the initialization registers.

DISPLAY CONTROL

The display control section generates linear addressing of up to 16K bytes of display memory. Internal comparators limit the portion of the memory which is displayed to programmed values. Additional functions performed in this section include cursor positioning and address comparisons required for generation of timing signals, double-height tops and bottoms, smooth scrolling, and the split-screen interrupts.

BUFFER CONTROL

The buffer control section generates three signals which control the transfer of data between the CPU and the display buffer memory. Four system configurations requiring four different 'handshaking' schemes are supported. These are described in **SYSTEM CONFIGURATIONS.**

TABLE 1 – AVDC ADDRESSING

A2	A1	A0	Read ($\overline{R}=0$)	Write ($\overline{W}=0$)
0	0	0	Interrupt Register	Initialization Registers*
0	0	1	Status Register	Command Register
0	1	0	Screen Start 1 Lower Register	Screen Start 1 Lower Register
0	1	1	Screen Start 1 Upper Register	Screen Start 1 Upper Register
1	0	0	Cursor Address Lower Register	Cursor Address Lower Register
1	0	1	Cursor Address Upper Register	Cursor Address Upper Register
1	1	0	Screen Start 2 Lower Register	Screen Start 2 Lower Register
1	1	1	Screen Start 2 Upper Register	Screen Start 2 Upper Register

*There are 15 initialization registers which are accessed sequentially via a single address. The AVDC maintains an internal pointer to these registers which is incremented after each write at this address until the last register (IR14) is accessed. The pointer then continues to point to IR14 for additional accesses. Upon a power-on or a master reset command, the internal pointer is reset to point to the first register (IR0) of the initialization register group. The internal pointer can also be preset to any register of the group via the 'load IR address pointer' command.

SYSTEM CONFIGURATIONS

Figure 1 illustrates the block diagram of a typical display terminal that uses an MC2674, character ROM, a keyboard interface, and an attribute controller. In this system, the CPU examines inputs from the data communications line and the keyboard and places the data to be displayed in the display buffer memory. This bufferis typically a RAM which holds the data for a single or multiple screenload (page) or for a single character row.

The AVDC supports four common system configurations of display-buffer memory, designated the independent, transparent, shared, and row-buffer modes. The first three modes utilize a single or multiple page RAM and differ primarily in the means used to transfer display data between the RAM and the CPU. The row-buffer mode makes use of a single row buffer (which can be a shift register or a small RAM) that is updated in real time to contain the appropriate display data.

The user programs IR0 bits 0 and 1 select the mode best suited for the system environment. The CTRL1, CTRL2, and CTRL3 outputs perform different functions for each mode and are named accordingly in the description of each mode.

3

INDEPENDENT MODE

The CPU-to-RAM interface configuration for this mode is illustrated in Figure 2. Transfer of data between the CPU and display memory is accomplished via a bidirectional latched port and is controlled by read data buffer ($\overline{RDB}$), write data buffer ($\overline{WDB}$), and buffer chip enable ($\overline{BCE}$). This mode provides a non-contention type of operation that does not require address multiplexers. The CPU does not address the memory directly — the read or write operation is performed at the address contained in the cursor address register or the pointer address register as specified by the CPU. The AVDC enacts the data transfers during blanking intervals in order to prevent visual disturbances of the displayed data.

The CPU manages the data transfers by supplying commands to the AVDC. The commands used are:

1. Read/write at pointer address,
2. Read/write at cursor address (with optional increment of address), and
3. Write from cursor address to pointer address.

The operational sequence for a write operation is:

1. CPU checks RDFLG status bit to assure that any delayed commands have been completed.
2. CPU loads data to be written to display memory into the interface latch.
3. CPU writes address into cursor or pointer registers.
4. CPU issues "write at cursor with/without increment" or "write at pointer" command.
5. AVDC generates control signals and outputs specified address to perform requested operation. Data is copied from the interface latch into the memory.
6. AVDC sets RDFLG status to indicate that the write is completed.

Similarly, a read operation proceeds as follows:

1. Steps 1. and 3. as above.
2. CPU issues "read at cursor with/without increment" or "read at pointer" command.
3. AVDC generates control signals and outputs specified address to perform requested operation. Data is copied from memory to the interface latch and AVDC sets RDFLG status to indicate that the read is completed.
4. CPU checks RDFLG status to see if operation is completed.
5. CPU reads data from interface latch.

Loading the same data into a block of display memory is accomplished via the "write from cursor to pointer" command:

1. CPU checks RDFLG status bit to assure that any delayed commands have been completed.
2. CPU loads data to be written to display memory into the interface latch.
3. CPU writes beginning address of memory block into cursor address register and ending address of block into pointer address register.
4. CPU issues "write from cursor to pointer" command.
5. AVDC generates control signals and outputs block addresses to copy data from the interface latch into the specified block of memory.
6. AVDC sets RDFLG status to indicate that the block write is completed.

Similar sequences can be implemented on an interrupt driven basis using the READY interrupt output to advise the CPU that a previously asserted delayed command has been completed.

Two timing sequences are possible for the "read/write at cursor/pointer" commands. If the command is given during the active display window (defined as first scan line of the first character row to the last scan line of the last character row), the operation takes place during the next horizontal blanking interval, as illustrated in Figure 3. If the command is given during the vertical blanking interval, or while the display has been commanded blanked, the operation takes place immediately. In the latter case, the execution time for the command is approximately five character clocks (see Figure 4).

Timing for the "write from cursor to pointer" operation is shown in Figure 5. The memory is filled at a rate of one location per two character times. The command will execute only during blanking intervals and may require many horizontal or vertical blanking intervals to complete. Additional delayed commands can be asserted immediately after this command has completed.

Immediately commands can be asserted at any time regardless of the state of the ready state/interrupt.

FIGURE 1 — CRT TERMINAL BLOCK DIAGRAM

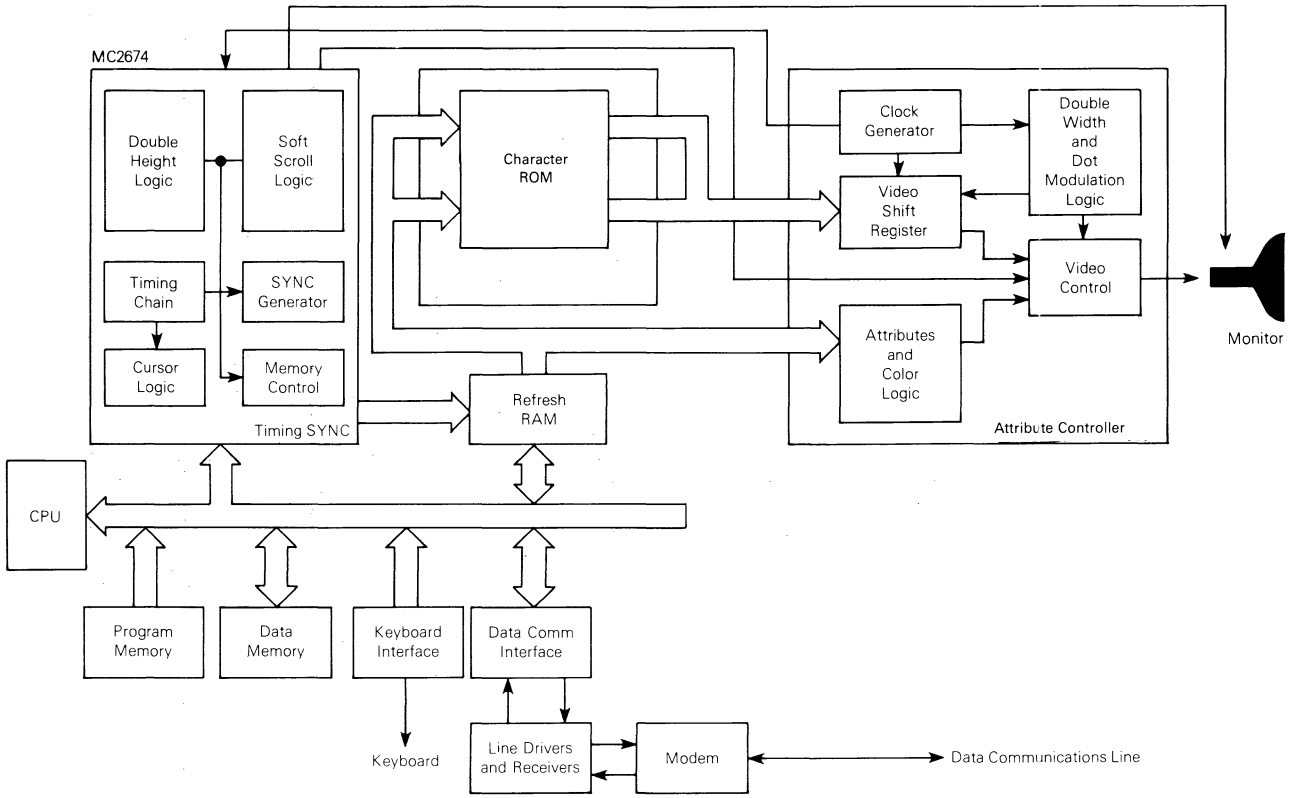


FIGURE 2 — INDEPENDENT BUFFER MODE CONFIGURATION

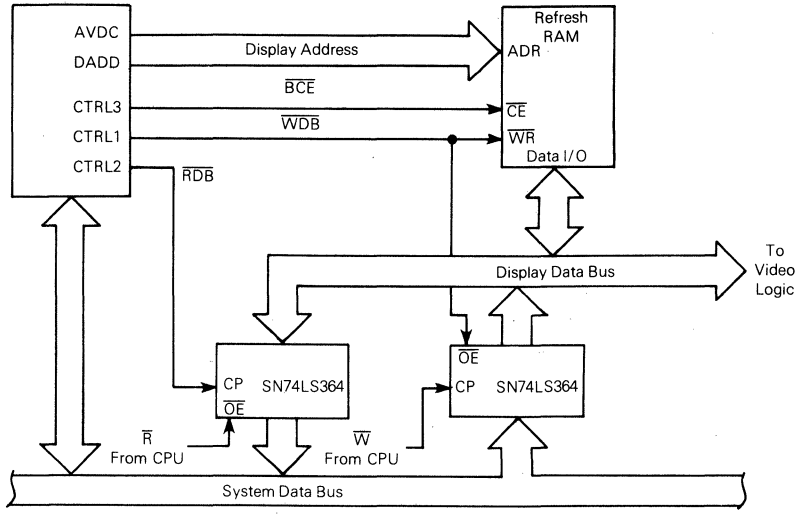
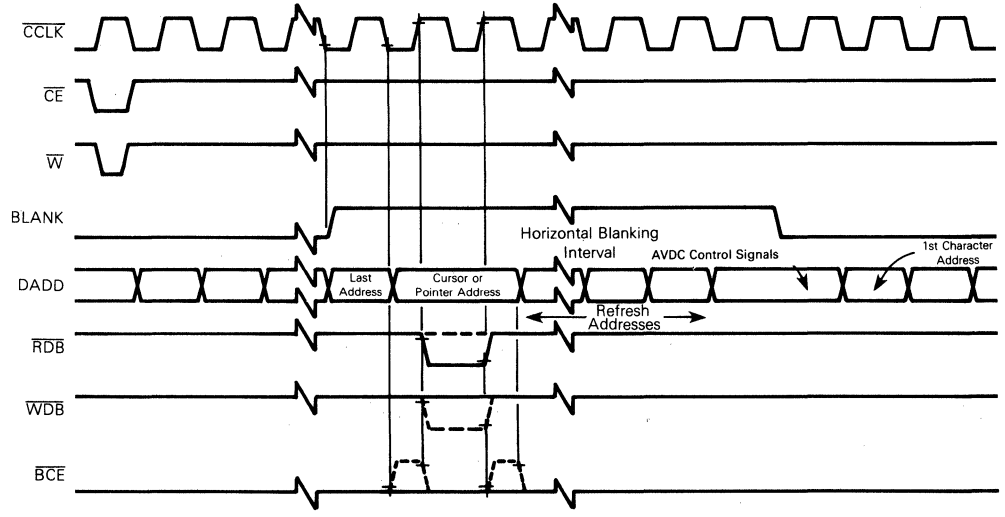
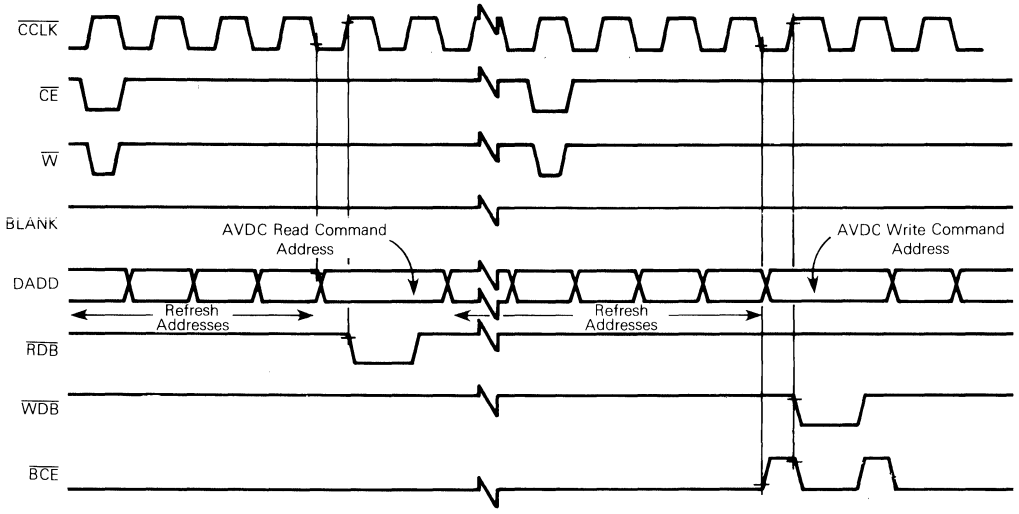


FIGURE 3 — READ/WRITE AT CURSOR/POINTER COMMAND TIMING
(Command Received During Active Display Window)



NOTES:
1. Write waveforms shown in dotted lines.
2. If command execution occurs just prior to the first scan line of a character row and row table addressing mode is enabled, execution of the command is delayed by two character clocks from the timing illustrated.
3. All voltage measurements are referenced to ground. All time measurements are at the 0.8 V to 2.0 V level for inputs and outputs. Input levels are 0.4 V to 2.4 V.

FIGURE 4 — READ/WRITE AT CURSOR/POINTER COMMAND TIMING
(Command Received While Display Is Blanked)



NOTE: All voltage measurements are referenced to ground. All time measurements are at the 0.8 V to 2.0 V level for inputs and outputs. Input levels are 0.4 V to 2.4 V.

SHARED AND TRANSPARENT BUFFER MODES

In these modes, the display buffer RAM is a part of the CPU memory domain and is addressed directly by the CPU. Both modes use the same hardware configuration with the CPU accessing the display buffer via three-state drivers (see Figure 6). The processor bus request ($\overline{PBREQ}$) control signal informs the AVDC that the CPU is requesting access to the display buffer. In response to this request, the AVDC raises bus acknowledge ($\overline{BACK}$) until its bus external ($\overline{BEXT}$) output has freed the display address and data buses for CPU access. $\overline{BACK}$, which can be used as a "hold" input to the CPU, is then lowered to indicate that the CPU can access the buffer.

In transparent mode, the AVDC delays the granting of the buffer to the CPU until a vertical or horizontal blanking interval, thereby causing minimum disturbance of the display. In shared mode, the AVDC will blank the display and grant immediate access to the CPU. Timing for these modes is illustrated in Figures 7, 8, and 9.

ROW BUFFER MODE

Figures 10 and 11 show the timing and a typical hardware implementation for the row buffer mode. During the first scan line (line 0) of each character row, the AVDC halts the CPU and DMA's the next row of character data from the system memory to the row buffer memory. The AVDC then releases the CPU and displays the row buffer data for the programmed number of scan lines. The control signal $\overline{BREQ}$ informs the CPU that character addresses and the MBC signal will start at the next falling edge of BLANK. The CPU must release the address and data buses before this time to prevent bus contention. After the row of character data is transferred to the CPU, $\overline{BREQ}$ returns high to grant memory control back to the CPU.

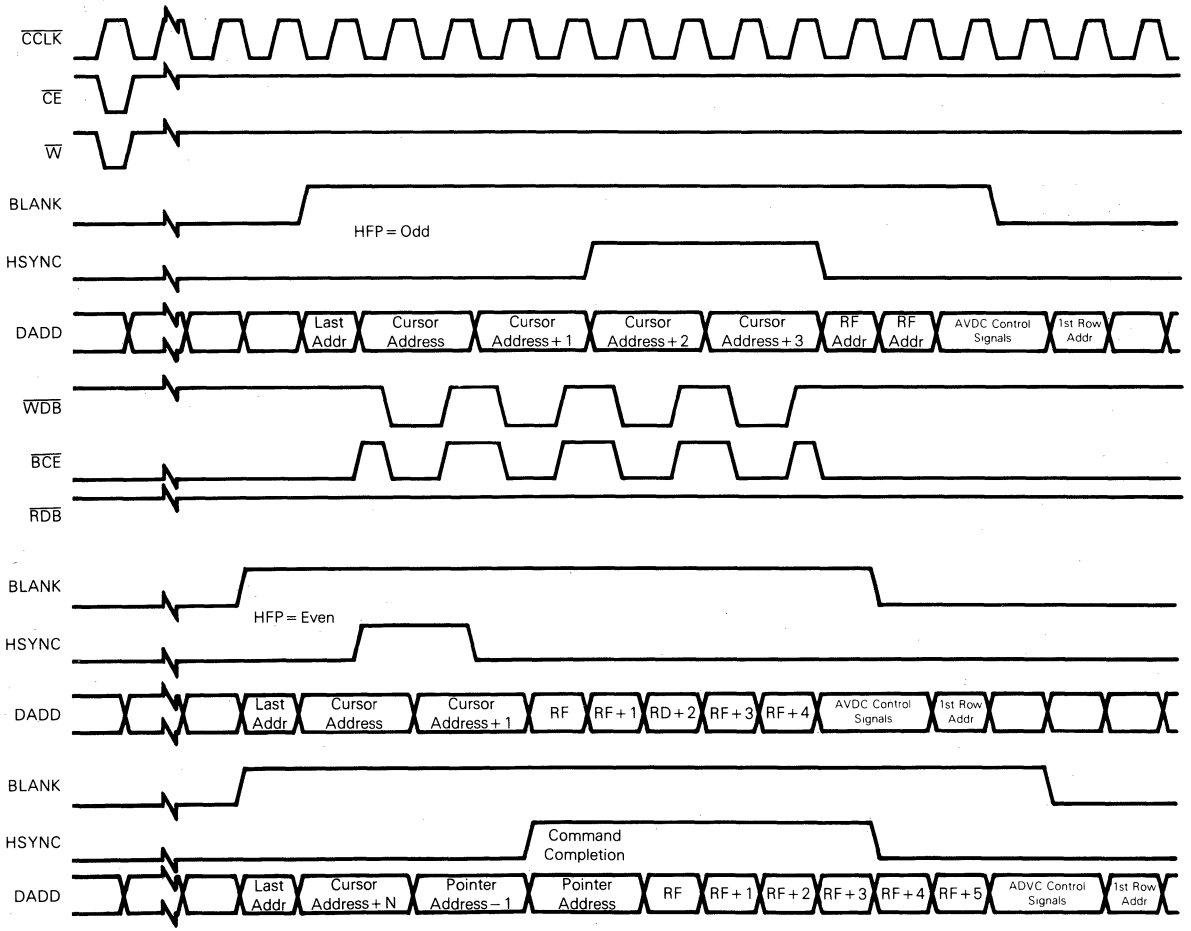
ROW TABLE ADDRESS MODE

In this mode, each character row in the screen image memory has a unique starting address. This provides greater flexibility with respect to screen operations, such as editing, than the sequential addressing mode. The row table, Figure 12, is a list of starting addresses for each character row and may reside anywhere in the AVDC's addressable memory space. Each entry in the table consists of two bytes: the first byte contains the eight least significant bits of the row starting address and the second byte contains, in its six least significant bits, the six most significant bits of the row starting address. The function of the two most significant bits of the second byte is selected by programming IR0[7]. They may be used either as row attribute bits to control double width and double height for that character row, or as an additional two address bits to extend the usable display memory to 64K.

The first address of the row table operation is designated in screen start register 2 (SSR2). If row table addressing is enabled via IR2[7], the AVDC fetches the next row's starting address from the table during the blanking interval prior to the first scan line of each character row, while simultaneously incrementing the contents of SSR2 by two so as to point to the next table entry. The fetching of the row starting address from the row table is indicated by the assertion of the CURSOR output during BLANK. The address read from the table by the AVDC is loaded into screen start register 1 (SSR1) for use internally. Since the contents of SSR2 changes as the table entries are fetched, it must be reinitialized to point to the first table entry during each vertical retrace interval.

Row table addressing is intended primarily for use in conjunction with the row buffer mode of operation and requires no additional circuitry in that case. It may also be used with

FIGURE 5 — WRITE FROM CURSOR TO POINTER COMMAND TIMING



NOTE:
If command execution occurs just prior to the first scan line of a character row and row table addressing mode is enabled, execution of the command is delayed by two character clocks from the timing illustrated.

FIGURE 6 – AVDC SHARED OR TRANSPARENT BUFFER MODES

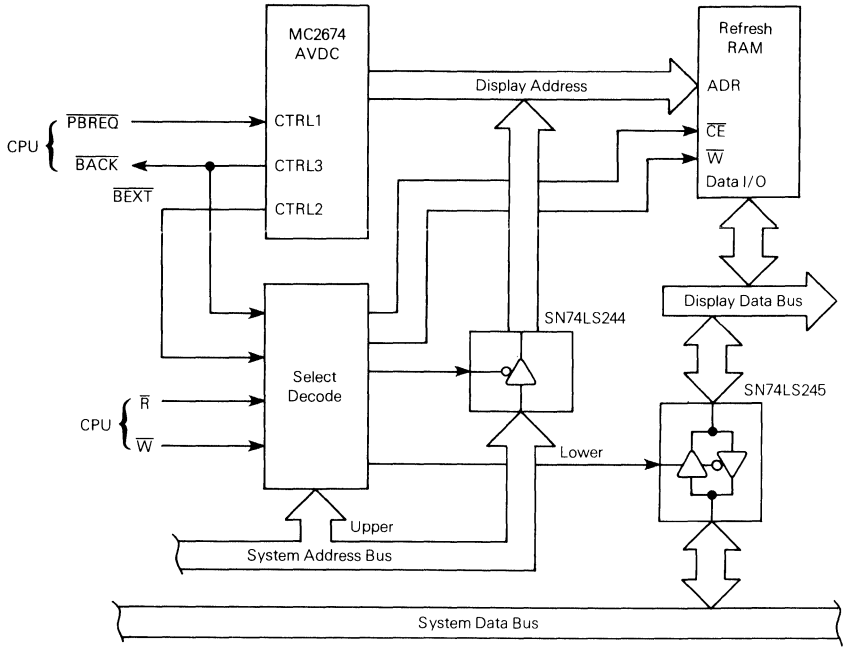
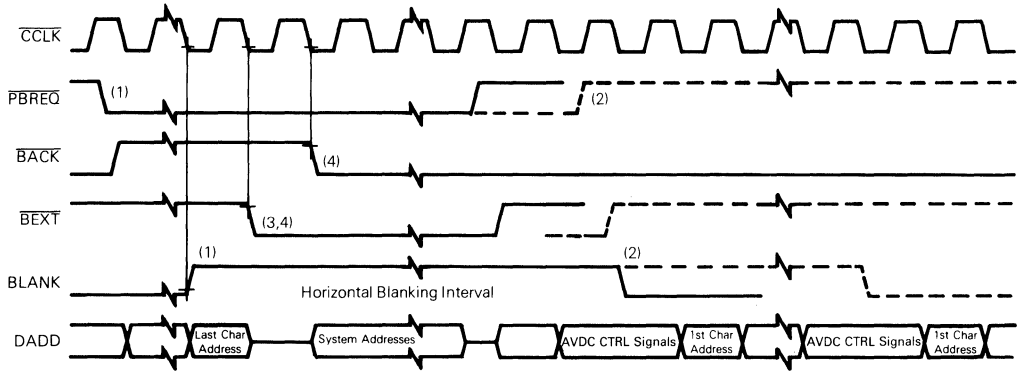


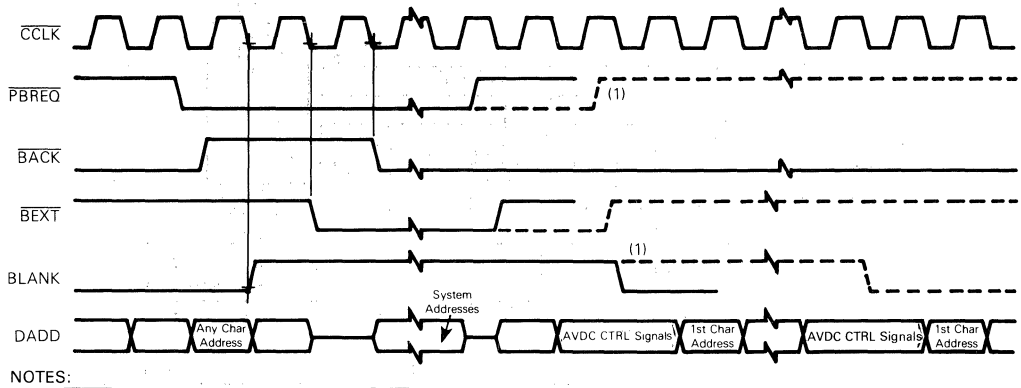
FIGURE 7 – TRANSPARENT BUFFER MODE TIMING



NOTES:

1. PBREQ must be asserted prior to the rising edge of BLANK in order for sequence to begin during that blanking period.
2. If PBREQ is negated after the next to last CCLK of the horizontal blanking interval, the next scan line will also be blanked.
3. Accesses during vertical blank or "display off" are granted only at the beginning of the horizontal front porch.
4. If row table addressing is enabled, CPU access is delayed by two character clocks prior to the first scan line of each character row.
5. All voltage measurements are referenced to ground. All time measurements are at the 0.8 V to 2.0 V level for inputs and outputs. Input levels are 0.4 V to 2.4 V.

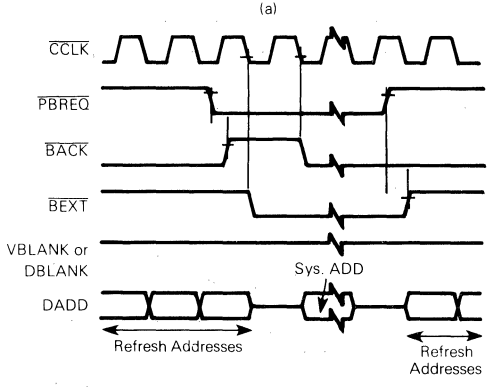
FIGURE 8 — SHARED BUFFER MODE TIMING



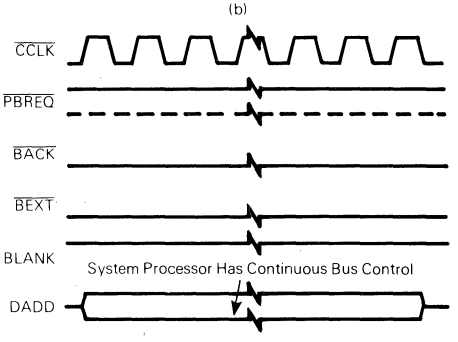
NOTES:
1. If PBREQ is negated after the next to last CCLK of the horizontal blanking interval, the next scan line will also be blanked.
2. All voltage measurements are referenced to ground. All time measurements are at the 0.8 V to 2.0 V level for inputs and outputs. Input levels are 0.4 V to 2.4 V.

3

FIGURE 9 — SHARED AND TRANSPARENT MODE TIMING



a) During Vertical Blank or after 'display off' command in shared mode only. See Figure 7 for transparent timing.



b) After 'display off and 3-state' command.

the other modes, but circuitry must be added to route the data from the display memory to the data bus inputs of the AVDC. Additionally, when not operating in row buffer mode, care must be taken to assure that the CPU does not attempt to access the AVDC while it is reading the row table. One way of preventing this is to latch prior to reading or writing the AVDC. The AVDC should only be accessed if the latch is low, indicating that the last line of the row is not active.

Figure 13 illustrates a typical hardware implementation for use in conjunction with independent and transparent modes, and Figure 14 shows the timing for row table operation.

OPERATION

After power is applied, the AVDC will be in an inactive state. Two consecutive "master reset" commands are necessary to release this circuitry and ready the AVDC for operation. Two register groups exist within the ADC; the initialization registers and the display control registers. The initialization registers select the system configuration, monitor timing, cursor shape, display memory domain, pointer address, scrolling region, double height and width condition, and screen format. These are loaded first and normally require no modification except for certain special visual

FIGURE 10 — ROW BUFFER MODE CONFIGURATION

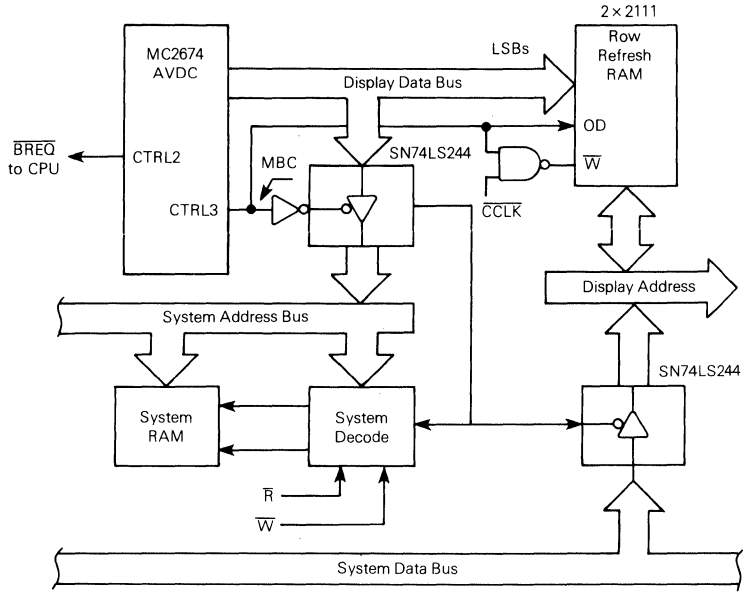
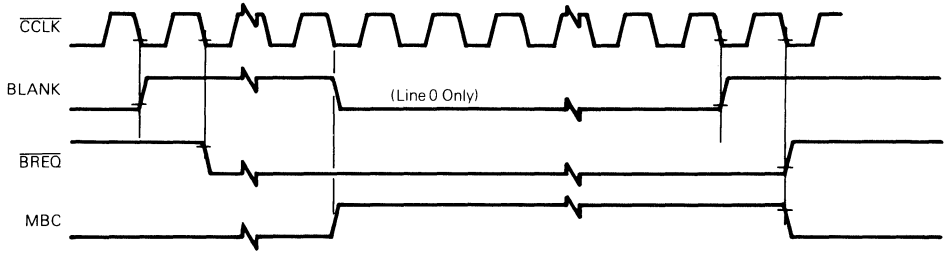


FIGURE 11 — ROW BUFFER MODE TIMING



NOTES:
1. If row table addressing is enabled, BREQ will be asserted at the middle of the last scan line of the prior row, and MBC will be asserted at the beginning of BLANK.
2. All voltage measurements are referenced to ground. All time measurements are at the 0.8 V to 2.0 V level for inputs and outputs. Input levels are 0.4 V to 2.4 V.

FIGURE 12 — ROW TABLE ADDRESS FORMAT

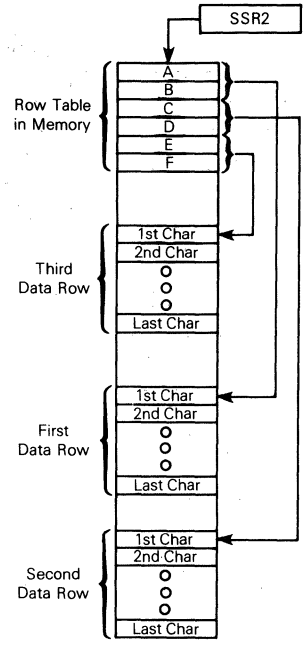


FIGURE 13 — ROW TABLE MODE CONFIGURATION (NON-ROW BUFFER MODES)

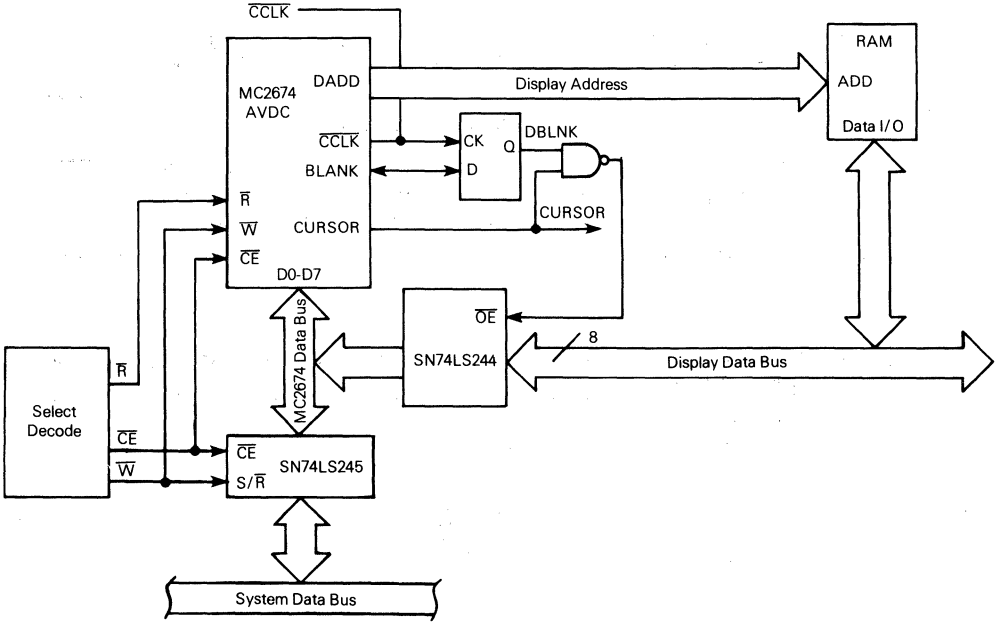
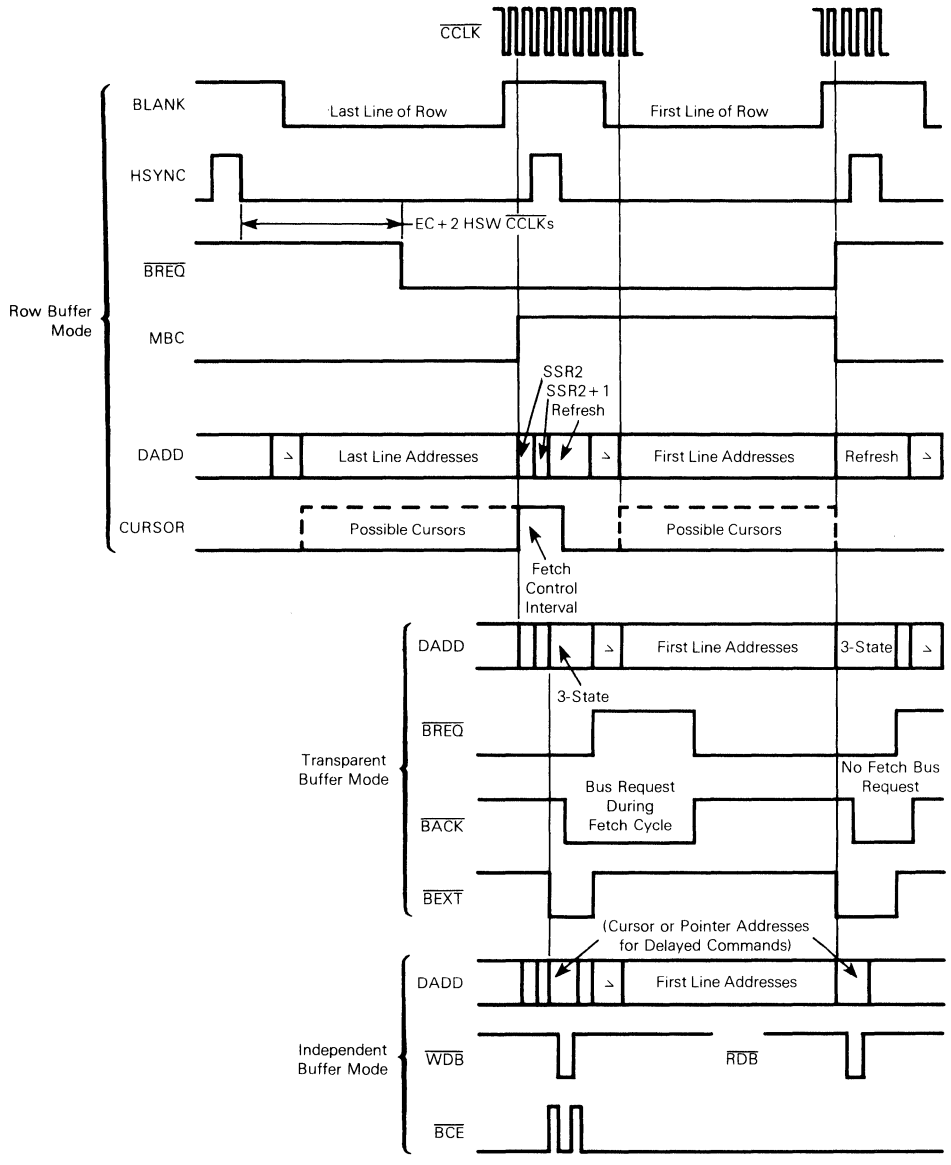


FIGURE 14 — ROW TABLE MODE TIMING



▵ = Multiplexed Control Signals
EC = Equalizing Constant
HSW = Horizontal SYNC Width

effects. The display control registers specify the memory address of the base character (upper left corner of screen), the cursor position, and the split screen addresses associated with the scrolling area or an alternate memory. These may require modification during operation.

After initial loading of the two register groups, the AVDC is ready to control the monitor screen. Prior to executing the AVDC commands which turn on the display cursor, the user should load the display memory with the first data to be displayed. During operation, the AVDC will sequentially address the display memory within the limits programmed into its registers. The memory outputs character codes to the system character and graphics generation logic, where they are converted to the serial video stream necessary to display the data on the CRT. The user effects changes to the CRT. The user effects changes to the display by modifying the contents of the display memory, the AVDC display control and command registers, and the initialization registers, if required. Interrupts and status conditions generated by the AVDC supply the "handshaking" information necessary for the CPU to effect real time display changes in the proper time frame if required.

INITIALIZATION REGISTERS

There are 15 initialization registers (IR0-IR14) which are accessed sequentially via a single address. The AVDC maintains an internal pointer to these registers which is incremented after each write at this address until the last register (IR14) is accessed. The pointer then continues to point to IR14 for further accesses. Upon a power-on or a master reset command, the internal pointer reset to point to the first register (IR0) of the initialization register group. The internal pointer can also be preset to any register of the group via the "load IR address pointer" command. These registers are write only and are used to specify parameters such as the system configuration, display format, cursor shape, and monitor timing. Register formats are shown in Figure 15.

3

FIGURE 15 — INITIALIZATION REGISTER FORMATS (Sheet 1 of 4)

IR0

7	6 5 4 3		2	1 0
Double Height/ Width	Scan Lines Per Character Row		Sync Select	Buffer-Mode Select
	Non-Interlaced	Interlaced		
	0000 = 1 Line 0001 = 2 Lines 0010 = 3 Lines • • 1110 = 15 Lines 1111 = 16 Lines	0000 = 2 Lines 0001 = 4 Lines 0010 = 6 Lines • • 1110 = 30 Lines 1111 = Undefined	0 = VSYNC 1 = CSYNC	00 = Independent 01 = Transparent 10 = Shared 11 = Row Buffer

IR1

7	6 5 4 3 2 1 0
Interlace Enable	Equalizing Constant
0 = Non-Interlace 1 = Interlace	0000000 = 1 $\overline{CCLK}$ 0000001 = 2 $\overline{CCLK}$ • • 1111110 = 127 $\overline{CCLK}$ 1111111 = 128 $\overline{CCLK}$ Calculated from: $EC = 0.5\ (H_{ACT} + H_{FP} + H_{SYNC} + H_{BP}) - 2(H_{SYNC})$

IR2

7	6 5 4 3	2 1 0
Row Table	Horizontal Sync Width	Horizontal Back Porch
0 = Off 1 = On	0000 = 2 $\overline{CCLK}$ 0001 = 4 $\overline{CCLK}$ • • 1110 = 30 $\overline{CCLK}$ 1111 = 32 $\overline{CCLK}$	000 = Not Allowed 001 = 3 $\overline{CCLK}$ • • 110 = 23 $\overline{CCLK}$ 111 = 27 $\overline{CCLK}$

FIGURE 15 — INITIALIZATION REGISTER FORMATS (Sheet 2 of 4)

IR3

7	6	5	4	3	2	1	0
Vertical Front Porch			Vertical Back Porch				
000 = 4 Scan Lines 001 = 8 Scan Lines • • 110 = 28 Scan Lines 111 = 32 Scan Lines			00000 = 4 Scan Lines 00001 = 6 Scan Lines • • 11110 = 64 Scan Lines 11111 = 66 Scan Lines				

IR4

7	6	5	4	3	2	1	0
Character Blink Rate	Active Character Rows Per Screen						
0 = 1/64 VSYNC 1 = 1/128 VSYNC	0000000 = 1 Row 0000001 = 2 Rows • • 1111110 = 127 Rows 1111111 = 128 Rows						

IR5

7	6	5	4	3	2	1	0
Active Characters Per Row							
00000010 = 3 Characters 00000011 = 4 Characters • • 11111110 = 255 Characters 11111111 = 256 Characters							

IR6

7	6	5	4	3	2	1	0
First Line of Cursor				Last Line of Cursor			
0000 = Scan Line 0 0001 = Scan Line 1 • • 1110 = Scan Line 14 1111 = Scan Line 15				0000 = Scan Line 0 0001 = Scan Line 1 • • 1110 = Scan Line 14 1111 = Scan Line 15			

IR7

7	6	5	4	3	2	1	0
Light Pen Line		Cursor Blink	Cursor Rate	Underline Position			
00 = Scan Line 3 01 = Scan Line 1 10 = Scan Line 5 11 = Scan Line 7		0 = Off 1 = On	0 = 1/32 1 = 1/64	0000 = Scan Line 0 0001 = Scan Line 1 • • 1110 = Scan Line 14 1111 = Scan Line 15			

3

FIGURE 15 — INITIALIZATION REGISTER FORMATS (Sheet 3 of 4)

IR8

7	6	5	4	3	2	1	0
Display Buffer First Address LSBs							
H'000' = 0 H'001' = 1 • • H'FFE' = 4,094 H'FFF' = 4,095						NOTE: MSBs are in IR9[3:0]	

IR9

7	6	5	4	3	2	1	0
Display Buffer Last Address				Display Buffer First Address MSBs			
0000 = 1,023 0001 = 2,047 • • 1110 = 15,359 1111 = 16,383				See IR8			

IR10

7	6	5	4	3	2	1	0
Display Pointer Address Lower							
See IR11							

IR11

7	6	5	4	3	2	1	0
LZ Down	LZ Up	Display Pointer Address Upper					
0 = Off 1 = On	0 = Off 1 = On	H'0000' = 0 H'0001' = 1 • • H'3FFF' = 16,383					

IR12

7	6	5	4	3	2	1	0
Scroll Start	Split Register 1						
0 = Off 1 = On	0000000 = Row 1 0000001 = Row 2 • • 1111111 = Row 128						

FIGURE 15 — INITIALIZATION REGISTER FORMATS (Sheet 4 of 4)

IR13

7	6 5 4 3 2 1 0
Scroll End	Split Register 2
0 = Off 1 = On	0000000 = Row 1 0000001 = Row 2 • • 1111111 = Row 128

IR14

7 6	5 4	3 2 1 0
Double 1	Double 2	Lines to Scroll
00 = Normal 01 = Double Width 10 = Double Width and Tops 11 = Double Width and Bottoms	00 = Normal 01 = Double Width 10 = Double Width and Tops 11 = Double Width and Bottoms	0000 = 1 0001 = 2 • • 1110 = 15 1111 = 16

DOUBLE HEIGHT/WIDTH ENABLE (IR0[7]) — When this bit is set, the value in IR14[7:6] is used to control the double height and width conditions of each character row. Assertion of this bit also allows IR14[7:6] to be programmed in two ways:

1. By the CP writing to IR14 directly.
2. When the contents of screen start register 1 (SSR1) upper are changed, either by the CPU writing to this register or by the automatic loading of SSR1 when operating in row table mode, the two most significant bits of SSR1 upper are copied into IR14[7:6]. Thus, the most significant bits of each row table entry can be used to control double height and double width attributes on a row-by-row basis.

IR14[5:4] are not active when this bit is set. When this bit is reset, the double height and width attributes operate as described in IR[14].

SCAN LINES PER CHARACTER ROW (IR0[6:3]) — Both interlaced and non-interlaced scanning are supported by the AVDC. For interlaced mode, two different formats can be implemented, depending on the interconnection between the AVDC and the character generator (see IR1[7]). This field defines the number of scan lines used to compose a character row for each technique. As scanning occurs, the scan line count is output on the LA0-LA3 and ODD pins.

VSYNC/CSYNC (IR0[2]) — This bit selects either vertical sync pulses or composite sync pulses on the VSYNC/CSYNC output (pin 18). The composite sync waveform conforms to EIA RS170 standards, with the vertical interval composed of six equalizing pulses, six vertical sync pulses, and six more equalizing pulses.

BUFFER MODE SELECT (IR0[1:0]) — Four buffer memory modes may be selectively enabled to accommodate the desired system configuration. See SYSTEM CONFIGURATIONS.

INTERLACE ENABLE (IR1[7]) — Specifies interlaced or non-interlaced timing operation. Two modes of interlaced operation are available, depending on whether L0-L3 or ODD, L0-L2 are used as the line address for the character generator. The resulting displays are shown in Figure 16.

For "interlaced sync" operation, the same information is displayed in both odd and even fields, resulting in enhanced readability. The AVDC outputs successive line numbers in ascending order on the LA0-LA3 lines, one per scan line for each field.

The "interlaced sync and video" format doubles the character density on the screen. The AVDC outputs successive line numbers in ascending order on the odd and LA0-LA2 lines, one per scan line for each field.

EQUALIZING CONSTANT (IR1[6:0]) — This field indirectly defines the horizontal front porch and is used internally to generate the equalizing pulses for the RS170 compatible CSYNC. The value for this field is the total number of character clocks ($\overline{CCLK}$s) during a horizontal line period divided by two, minus two times the number of character clocks in the horizontal sync pulse:

$$EC = \frac{H_{ACT} + H_{FP} + H_{SYNC} + H_{SP}}{2} - 2\,(H_{SYNC})$$

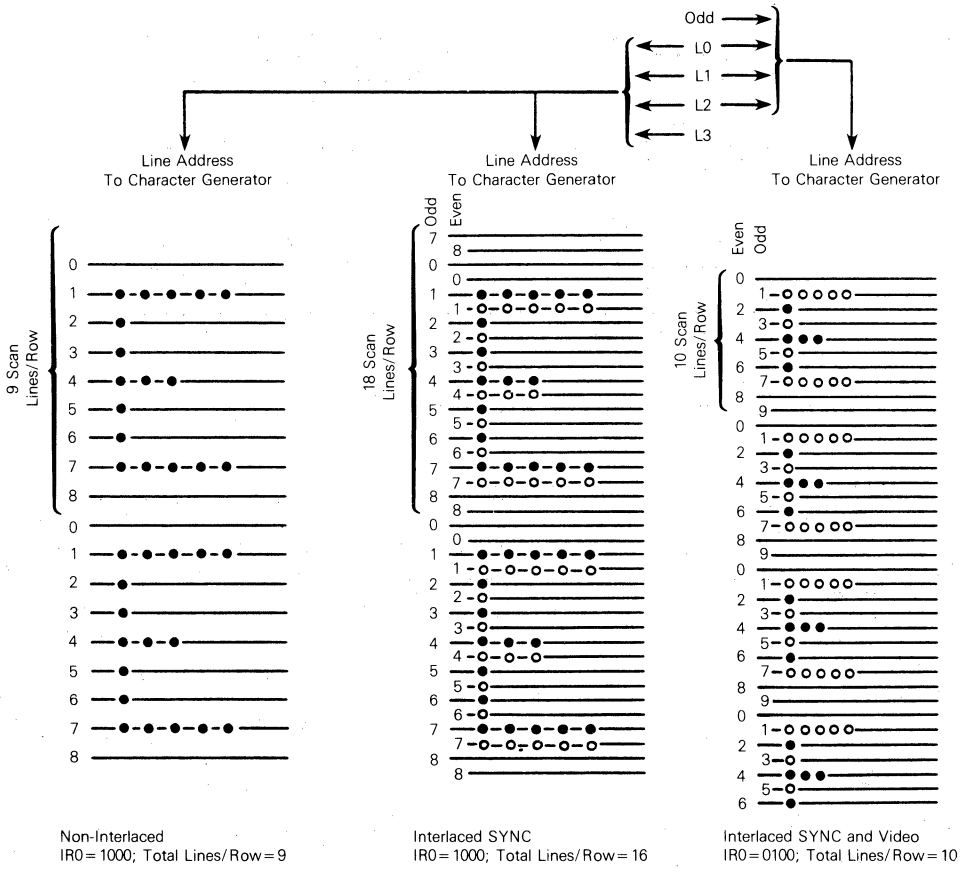
The definition of the individual parameters is illustrated in Figure 17.

Note that when using the attributes controller it will delay the blank pulse three $\overline{CCLK}$s relative to the HSYNC pulse.

ROW TABLE MODE ENABLE (IR2[7]) — Assertion/negation of this bit causes the AVDC to begin/terminate operating in row table mode starting at the next character row. See **ROW TABLE ADDRESS MODE.** By using the split interrupt capability of the AVDC, this mode can be enabled and disabled on a particular character row. This allows a combination of row table and sequential addressing to be utilized to provide maximum flexibility in generating the display.

HORIZONTAL SYNC PULSE WIDTH (IR2[6:3]) — This field specifies the width of the HSYNC pulse in $\overline{CCLK}$ periods.

FIGURE 16 — INTERLACED DISPLAY MODES



HORIZONTAL BACK PORCH (IR2[2:0]) — This field defines the number of $\overline{\text{CCLK}}$s between the trailing edge of HSYNC and the trailing edge of BLANK.

VERTICAL FRONT PORCH (IR3[7:3]) — This field specifies the number of scan line periods between the rising edges of BLANK and VSYNC during the vertical retrace interval. The vertical front porch is extended in increments of scan lines if the ACLL input is low at the end of the programmed value.

VERTICAL BACK PORCH (IR3[4:0]) — This field determines the number of scan line periods between the falling edges of the VSYNC and BLANK outputs.

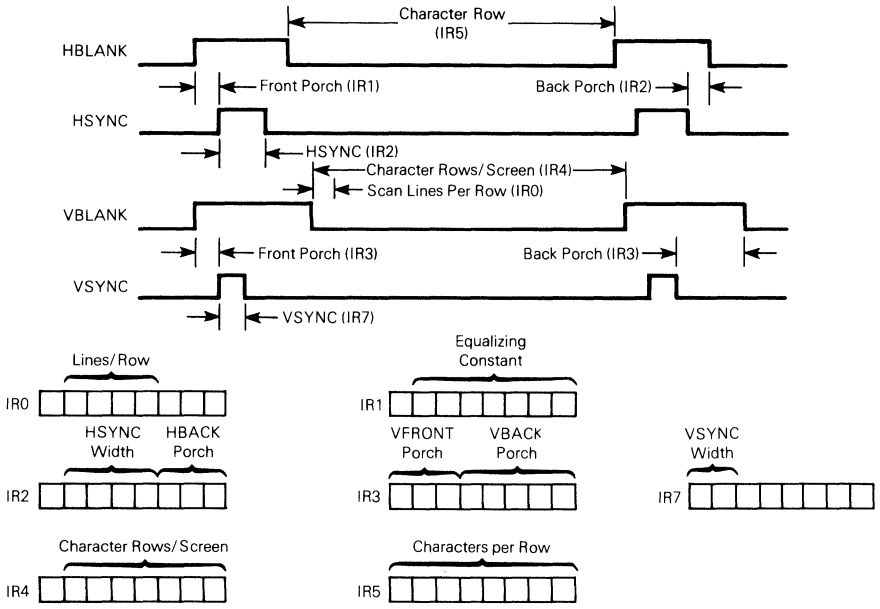
CHARACTER BLINK RATE (IR4[7]) — Specifies the frequency for the character blink attribute timing. The blink rate can be specified as 1/64 or 1/128 of the vertical field rate. The timing signal has a duty cycle of 50% and is multiplexed onto the DADD1/BLINK output at the falling edge of each BLANK.

CHARACTER ROWS PER SCREEN (IR4[6:0]) — This field defines the number of character rows to be displayed. The value multiplied by the scan lines per characte row, plus the vertical front porch, the vertical back porch values, and the vertical sync pulse width is the vertical scan period in scan lines.

ACTIVE CHARACTERS PER ROW (IR5[7:0]) — This field determines the number of characters to be displayed on each row of the CRT screen. The sum of this value, the horizontal front porch, the horizontal sync width, and the horizontl back porch is the horizontal scan period in CCLKs.

FIRST AND LAST SCAN LINE OF CURSOR (IR6[7:4], IR6[3:0]) — These two field specify the height and position of the cursor on the character block. The "first" line is the topmost line when scanning from the top to the bottom of the screen.

FIGURE 17 — HORIZONTAL AND VERTICAL TIMING



VERTICAL SYNC PULSE WIDTH (IR7[7:6]) — This field specifies the width of the VSYNC pulse in scan line periods.

CURSOR BLINK ENABLE (IR7[5]) — This bit controls whether or not the cursor output pin will be blinked at the selected rate (IR7[4]). The blink duty cycle for the cursor is 50%.

CURSOR BLINK RATE (IR7[4]) — The cursor blink rate can be specified at 1/32 or 1/64 of the vertical scan frequency. Blink is effective only if blink is enabled by IR7[5].

UNDERLINE POSITION (IR7[3:0]) — This field defines which scan line of the character row will be used for the underline attribute by the attributes controller. The timing signal is multiplexed onto the DADD10/UL output during the falling edge of BLANK.

DISPLAY BUFFER FIRST ADDRESS (IR9[3:0]), IR8[7:0] AND DISPLAY BUFFER LAST ADDRESS (IR9[7:4]) — These two fields define the area within the buffer memory where the display data will reside. When the data at the "display buffer last address" is displayed, the AVDC will wraparound and obtain the data to be displayed at the next screen position from the "display buffer first address". If "last address" is the end of a character row and a new screen start address has been loaded into the screen start register, or if "last address" is the last character position of the screen, the next data is obtained from the address contained in the screen start register.

Note that there is no restriction in displaying data from other areas of the addressable memory. Normally, the area between these two bounds is used for data which can be overwritten (e.g., as a result of scrolling), while data that is not to be overwritten would be contained outside these bounds and accessed by means of the automatic split screen or split screen interrupt features of the AVDC.

DISPLAY POINTER ADDRESS LOWER (IR10[7:0] AND DISPLAY POINTER ADDRESS UPPER (IR11[5:0]) — These two fields define a buffer memory address for AVDC controlled accesses in response to "read/write at pointer" commands. They also define the last buffer memory address to be written for the "write from cursor to pointer" command.

SCAN LINE ZERO DURING SCROLL DOWN (IRZ11[7]) — This field specifies normal scan line count or all scan line zero counts for the new character row that occurs at the top of the scrolling area during soft scroll down operation. If the character generator provides blanks during scan line zero, this will cause the new row to be automatically blanked on the display. This feature can be used, if necessary, to blank the new row until the CPU places "blank data" into the display buffer.

SCAN LINE ZERO DURING SCROLL UP (IR11[6]) — This field specifies normal scan line count or all scan line counts for the new character row that occurs at the bottom of the scrolling area during soft scroll up operation.

SCROLL START (IR12[7]) — This bit is asserted when soft scroll is to take place. The scrolling area begins at the row specified in split register 1 (IR12[6:0]). If set, the first

row to scroll scan line count will be reduced by the value in the lines to scroll register (IR14[3:0]). The scan line count of this row will start at the programmed offset value. When this bit is asserted, scroll end IR13[7] must be set before split register 2.

SPLIT REGISTER 1 (IR12[6:0]) — Split register 1 can be used to provide special screen effects such as soft (scan line by scan line) scrolling, double height/width rows, or to change the normal addressing sequence of the display memory. The contents of this field is compared, in real time, to the current row number. Upon a match, the AVDC sets the split screen 1 status bit, and issues an interrupt request if so programmed. The status change/interrupt request is made at the beginning of the scan line zero of the split screen character row. If enabled by the SPL1 bit of screen start register 2, an automatic split screen to the address specified in screen start register 2 will be made for the designated character row. During a scroll operation, this field defines the first character row of the scrolling area.

SCROLL END (IR13[7]) — This field specifies that the row programmed in split register 2 (IR13[6:0]) is to be the last scrolling row of the scrolling area. Note that this bit must be asserted for a valid row only when the scroll start bit IR12[7] is also asserted.

SPLIT REGISTER 2 (IR13[6:0]) — This field is similar to the split register 1 field except for the following:

1. Split screen 2 status bit is set.
2. During a scroll operation, this field defines the last character row of the scrolling area. This row will be followed by a partial row. The LTSR (IR14) value replaces the normal scan lines/row value for the partial row, thus keeping the total scan lines/screen the same.
3. If enabled by the SPL2 bit of screen start register 2, an automatic split to the address contained in screen start register 2 will occur in one of two ways:
   a) If not scrolling an automatic split will occur for the next character row.
   b) If scrolling, the automatic split will occur after the partial row being scrolled onto or off the screen.
4. The specified double width and height conditions (IR14) are also asserted in two possible ways:
   a) Automatic split will assert the programmed condition for the current row.
   b) During soft scroll operation the programmed conditions are asserted for the partial row scrolling onto or off the screen.

DOUBLE 1 (IR14[7:6]) — This field specifies the conditions (double width/height or normal) of the row designated in split register 1 (IR12[6:0]). When double height tops or bottoms has been specified, the AVDC will automatically toggle between tops and bottoms until another split 1 or 2 occurs which changes the double height/width condition. If a double height top row is specified, the scan line count will start at zero and increment the scan line every other scan line. If a double height bottom row is specified, the AVDC will start a one half the normal scan line total. If double width is specified, the AVDC will assert the DADD9/DW output at the falling edge of blank. This condition will also remain active until the next split 1 or 2. When IR0[7] = 1, the values written into bits 7 and 6 of screen start 1 upper will also be written into IR14[7:6] and the automatic toggling between tops and bottoms is disabled.

DOUBLE 2 (IR14[5:4]) — This field specifies the conditions (double width/height or normal) of the row designated in split register 2 (IR13[6:0]). Not used with IR0[7] = 1.

LINES TO SCROLL (IR14[3:0]) — This field defines the scan line increment to be used during a soft scroll operation. This value will only be used when scroll start (IR12[7]) and scroll end (IR13[7]) are enabled.

TIMING CONSIDERATIONS

Normally, the contents of the initialization registers are not changed during normal operation. However, this may be necessary to implement special display features such as multiple cursors and horizontal scrolling. Table 2 describes timing details for these registers which should be considered when implementing these features.

TABLE 2 — TIMING CONSIDERATIONS

Parameter	Timing Considerations
First Line of Cursor Last Line of Cursor Underline Line	These parameters must be established at a minimum of two character times prior to their occurrence.
Double Height Character Rows Double Width Character Rows Rows to Scroll	Set/reset prior to the row specified in split 1 or 2 registers.
Cursor Blink Cursor Blink Rate Character Blink Rate	New values become effective within one field after values are changed.
Split Register 1 Split Register 2	Change anytime prior to line zero of desired row.
Character Rows Per Screen	Change only during vertical blanking period.
Vertical Front Porch	Change prior to first line of VFP.
Vertical Back Porch	Change prior to four line after VSYNC.
Screen Start Register 1 Row Table Mode Enable	Change prior to the horizontal blanking interval of the last line of character row before row where new value is to be used.

DISPLAY CONTROL REGISTERS

There are seven registers in this group, each with an individual address. Their formats are illustrated in Figure 18. The command register is used to invoke one of 19 possible AVDC commands as described in COMMANDS. The remaining registers in the group store address values which specify the cursor location, the location of the first character to be be displayed on the screen, and any split screen address locations. The user initializes these registers after powering on the system and changes their values to control the data which is displayed.

SCREEN START REGISTERS 1 AND 2

The screen start 1 registers contain the address of the first character of the first row (upper left corner of the active display). At the beginning of the first scan line of the first row, this address is transferred to the row start register (RSR) and into the memory address counter (MAC). The counter is then advanced sequentially at the character clock rate for the number of times programmed into the active characters per row register (IR5), thus reaching the address of the last character of the row plus one. At the beginning of each subsequent scan line of the first row, the MAC is reloaded from the RSR and the above sequence is repeated. At the end of the last scan line of the first row, the contents of the MAC is loaded into the RSR to serve as the starting memory address for the second character row. This process is repeated for the programmed number of rows per screen. Thus, the data in the display memory is displayed sequentially starting from the address contained in the screen start register. After the ensuing vertical retrace interval, the entire process repeats again.

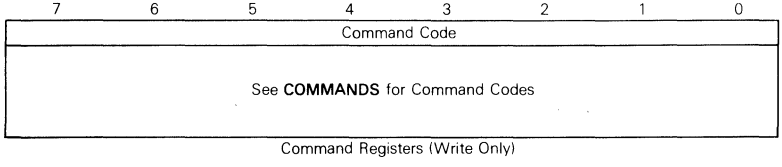
During vertical blanking, the address counter operation is modified by stopping the automatic load of the contents of the RSR into the counter, thereby allowing the address outputs to free-run. This allows dynamic memory refresh to occur during the vertical retrace interval. The refresh addressing starts at the last address displayed on the screen and increments by one for each character clock during the retrace interval. If the display buffer last address is encountered, refreshing continues from the display buffer first address.

The sequential operation described above will be modified upon the occurrence of any of three events. First, if during the incrementing of the memory address counter the "display buffer last address" (IR9[7:4]) is reached, the MAC will be loaded from the "display buffer first address" register (IR9[3:0] and IR8[7:0]) at the next character clock. Sequential operation will then resume starting from this address. This wraparound operation allows portions of the display buffer to be used for purposes other than storage of displayable data and is completely automatic without any CPU intervention (see Figure 19a).

The sequential row to row addressing can also be modified via split register 1 (IR12) and split register 2 (IR13), under CPU control, or by enabling the row table addressing mode. If bit 6 of screen start register 2 upper (SPL1) is set, the screen start register 2 contents will be loaded automatically into the RSR at the beginning of the first scan line of the row designated by split register 1 (IR12[6:0]). If bit 7 of screen start 2 upper (SPL2) is set, the screen start register 2 contents is automatically loaded into the RSR at the end of the last scan line of the row designated by split register 2 (IR13[6:0]). SPL1 and SPL2 are write only bits and will read as zero when reading screen start register 2.

If the contents of screen start register 1 (upper, lower, or both) are changed during any character row (e.g., row 'n'), the starting address of the next character row (row 'n + 1') will be the new value of the screen start register and addressing will continue sequentially from there. This allows features such as split screen operation, partial scroll, or status line display to be implemented. The split screen interrupt feature of the AVDC is useful in controlling the CPU initiated operations. Note that in order to obtain the correct screen display, screen start register 1 must be reloaded with the original (origin of display) value prior to the end of the vertical retrace. See Figure 19b.

FIGURE 18 — DISPLAY CONTROL REGISTER FORMATS (Sheet 1 of 2)



Command Registers (Write Only)

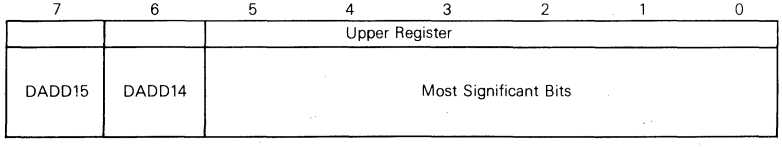


FIGURE 18 — DISPLAY CONTROL REGISTER FORMATS (Sheet 2 of 2)

7	6	5	4	3	2	1	0
Lower Register (Least Significant Bit)							
H'0000' = 0 H'0001' = 1 Through H'3FFE' = 16,382 H'3FFF' = 16,383			NOTE: Most significant bits are in upper register [5:0]				

NOTES:
1. Bits 7 and 6 of upper register are not used in the cursor address register.
2. Bits 7 and 6 of upper register are always zero when read by the CPU.
3. When IR0[7] = 1, the values written into bits 7 and 6 of screen start 1 upper will also be written into IR14[7:6] to control the double width and double height attributes of the display as follows:

7	6	Attribute
0	0	None
0	1	Double Width Only
1	0	Double Width and Double Height Tops
1	1	Double Width and Double Height Bottoms

Screen Start 1 Register (Read and Write) and Cursor Address Registers (Read and Write)

7	6	5	4	3	2	1	0
		Upper Register					
SPL2 0 = Off 1 = On	SPL1 0 = Off 1 = On	Most Significant Bits					

7	6	5	4	3	2	1	0
Lower Register (Least Significant Bit)							
H'0000' = 0 H'0001' = 1 Through H'3FFE' = 16,382 H'3FFF' = 16,383			NOTE: Most significant bits are in upper register [5:0]				

NOTE:
Bit 7 and bit 6 are always zero when read by the CPU.

Screen Start 2 Registers (Read and Write)

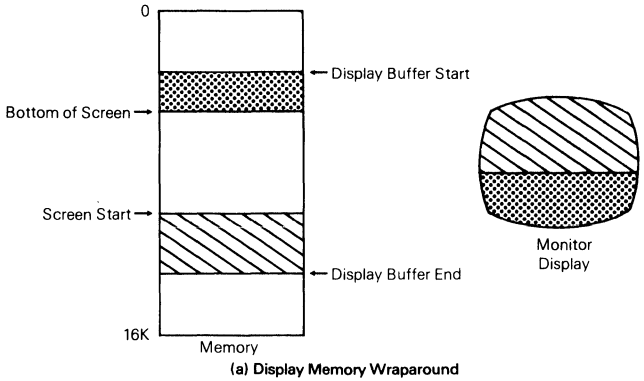
When row table addressing mode is enabled, the first address of the row table is designated in SSR2. The AVDC fetches the next row's starting address from the table during the blanking interval prior to the first scan line of each character row and loads it into SSR1 for use as the starting address of the next row. Since the contents of SSR2 changes as the table entries are fetched, it must be re-initialized to point to the first table entry during each vertical retrace interval.

The values of the two most significant bits of SSR1 upper are multiplexed onto the DADD1/DADD14 and DADD2/DADD15 outputs during the falling edge of BLANK. If IR0[7] = 0, these two bits act as memory page select bits which may be used to extend the display memory addressing range of the AVDC up to 64K. In that case, these two bits act as a two-bit counter which is incremented each time that "wraparound" occurs (see above). Note that the counter is incremented at the falling edge of BLANK and that for proper display operation the wraparound address should be programmed to occur at the last character position of a row. Also, the first address accessed in the new page will be the address contained in the display buffer first address register (IR9[3:0] and IR8[7:0]).

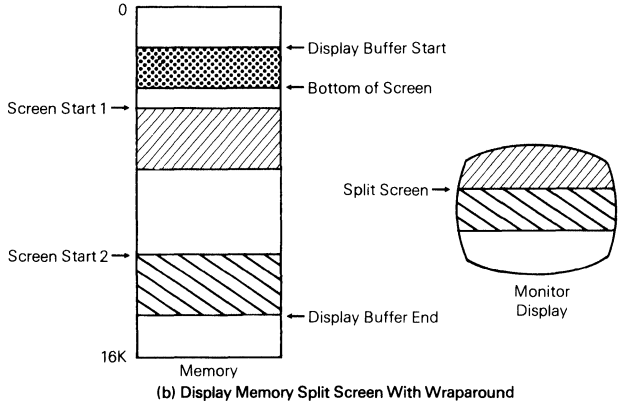
CURSOR ADDRESS REGISTERS

The contents of these registers define the buffer memory address of the cursor. The cursor output will be asserted when the memory address counter matches the value of the

FIGURE 19 — DISPLAY ADDRESSING OPERATION



(a) Display Memory Wraparound



(b) Display Memory Split Screen With Wraparound

cursor address registers for the scan lines specified in IR6. The cursor address registers can be read or written by the CPU or incremented via the "increment cursor address" command. In independent buffer mode, these registers define a buffer memory address for AVDC controlled access in response to "read/write at cursor with/without increment" commands, or the first address to be used in executing the "write from cursor to pointer" command.

INTERRUPT/STATUS REGISTERS

The interrupt and status registers provide information to the CPU to allow it to interact with the AVDC to effect desired changes that implement various display operations. The interrupt register provides information on five display operations. The interrupt register provides information on five possible interrupt conditions, as shown in Figure 20. These conditions can be selectively enabled or disabled (masked) from causing interrupts by certain AVDC commands. An interrupt condition which is enabled (masked bit equal to one) will cause the $\overline{INTR}$ output to be asserted and will cause the corresponding bit in the interrupt register to be set upon the occurrence of the interrupting condition. An interrupt condition which is disabled (mask bit equal to zero) has no effect on either the $\overline{INTR}$ output or the interrupt register.

The status register provides six bits of status information: the five possible interrupt conditions plus the RDFLG bit. For this register, however, the contents are not affected by the state of the mask bits.

Descriptions of each interrupt/status register bit follow. Unless otherwise indicated, a bit, once set, will remain set until reset by the CPU by issuing a "reset interrupt/status bits" command. The bits are also reset by a "master reset" command and upon power-up.

FIGURE 20 — INTERRUPT AND STATUS REGISTER FORMAT

7	6	5	4	3	2	1	0
		RDFLG	VBLANK	Line Zero	Split 1	Ready	Split 2
Not Used Always Read as 0		0 = Busy 1 = Ready	0 = No 1 = Yes	0 = No 1 = Yes	0 = No 1 = Yes	0 = Busy 1 = Ready	0 = No 1 = Yes

RDFLG (I/SR[5]) — This bit is present in the status register only. A zero indicates that the AVDC is currently executing the previously issued delayed command. A one indicates that the AVDC is ready to accept a new delayed command.

VBLANK (I/SR[4]) — Indicates the beginning of a vertical blanking interval. Set to one at the beginning of the first scan line of the vertical front porch.

LINE ZERO (I/SR[3]) — Set to one at the beginning of the first scan line (line 0) of each active character row.

SPLIT SCREEN 1 (I/SR[2]) — This bit is set when a match occurs between the current character row number and the value contained in split register 1, IR12[6:0]. The equality condition is only checked at the beginning of line zero of each character row.

READY (I/SR[1]) — The delayed commands affect the display and may require the AVDC to wait for a blanking interval before enacting the command. This bit is set to one when execution of a delayed command has been completed. No other delayed command should be invoked until the prior delayed command is completed.

SPLIT SCREEN 2 (I/SR[0]) — This bit is set when a match occurs between the current character row number and the value contained in split register 2 (IR13[6:0]).

COMMANDS

The AVDC commands are divided into two classes: the instantaneous commands which are executed immediately after they are invoked, and the delayed commands which may need to wait for a blanking interval prior to their execution. Command formats are shown in Table 3. The commands are asserted by performing a write operation to the command register with the appropriate bit pattern as the data byte.

TABLE 3 — AVDC COMMAND FORMATS

D7	D6	D5	D4	D3	D2	D1	D0	Hex	Command
Instantaneous Commands									
0	0	0	0	0	0	0	0		Master Reset
0	0	0	1	V	V	V	V		Load IR Pointer with Value V (V = 0 to 14)
0	0	1	d	d	d	1	0*		Disable Graphics
0	0	1	d	d	d	1	1*		Enable Graphics
0	0	1	d	1	N	d	0*		Display Off — Float DADD Bus if N = 1
0	0	1	d	1	N	d	1*		Display On — Next Field (N = 1) or Scan Line (N = 0)
0	0	1	1	d	d	d	0*		Cursor Off
0	0	1	1	d	d	d	1*		Cursor On
0	1	0	N	N	N	N	N		Reset Interrupt/Status: bit Reset where N = 1
1	0	0	N	N	N	N	N		Disable Interrupt: Disable where N = 1
0	1	1	N	N	N	N	N		Enable Interrupt: Enables Interrupts where N = 1
			V B	L Z	S P 1	R D Y	S P 2		Interrupt Bit Assignments
Delayed Commands									
1	0	1	0	0	1	0	0	A4	Read at Pointer Address
1	0	1	0	0	0	1	0	A2	Write at Pointer Address
1	0	1	0	1	0	0	1	A9	Increment Cursor Address
1	0	1	0	1	1	0	0	AC	Read at Cursor Address
1	0	1	0	1	0	1	0	AA	Write at Cursor Address
1	0	1	0	1	1	0	1	AD	Read at Cursor Address and Increment Address
1	0	1	0	1	0	1	1	AB	Write at Cursor Address and Increment Address
1	0	1	1	1	0	1	1	BB	Write from Cursor Address to Pointer Address
1	0	1	1	1	1	0	1	BD	Read from Cursor Address to Pointer Address

NOTES:
*Any combination of these three commands is valid.
d = Don't care.

INSTANTANEOUS COMMANDS

The instantaneous commands are executed immediately after the trailing edge of the write pulse during which the command is issued. These commands do not affect the state of the RDFLG or READY interrupt/status bits and can be invoked at any time.

MASTER RESET

This command initializes the AVDC and can be invoked at any time to return the AVDC to its initial state. Upon power-up, two successive master reset commands must be applied to release the AVDC's internal power-on circuits. In transparent and shared buffer modes, the CTRL1 input must be high when the command is issued. The command causes the following:

1. VSYNC and HSYNC are driven low for the duration of the command and BLANK goes high. After command completion, HSYNC and VSYNC will begin operation and BLANK will remain high until a "display on" command is received.
2. The interrupt and status bits and masks are set to zero, except for the RDFLG flag which is set to a one.
3. The row buffer mode, cursor-off, display-off, and line graphics disable states are set.
4. The initialization register pointer is set to address IR0.
5. IR2[7] is reset.

LOAD IR ADDRESS

This command is used to preset the initialization register pointer with the value "V" defined by D3-D0. Allowable values are 0 to 14.

ENABLE GRAPHICS

After invoking this command, the AVDC will increment the MAC to the next consecutive memory address for each scan line even if more than one scan line per row is programmed. This mode can be used for bit-mapped graphics where each location in the display buffer within the defined area contains the bit pattern to be displayed. This command is row buffered and should be asserted during the character row prior to the row where this feature is required. This allows the user to enter and exit graphics mode on character row boundaries.

To perform split screen operations while in graphics mode use SSR2 only.

DADD0/LG is asserted during the trailing edge of BLANK for each scan line while this mode is active.

DISABLE GRAPHICS

Normal addressing resumes at the next row boundary.

DISPLAY OFF

Asserts the BLANK output. The DADD0 through DADD13 display address bus outputs can be optionally placed in the three-state condition by setting bit 2 to a one when invoking the command.

DISPLAY ON

Restores normal blanking operation either at the beginning of the next field (bit 2 = 1) or at the beginning of the next scan line (bit 2 = 0). Also returns the DADD0-DADD13 drivers to their active state.

CURSOR OFF

Disables cursor operation. Cursor output is placed in the low state.

CURSOR ON

Enables normal cursor operation.

RESET INTERRUPT/STATUS BITS

This command resets the designated bits in the interrupt and status registers. The bit positions correspond to the bit positions in the registers:

Bit 0 – Split 2
Bit 1 – Ready
Bit 2 – Split 1
Bit 3 – Line Zero
Bit 4 – Vertical Blank

DISABLE INTERRUPTS

Sets the interrupt mask to zeros for the designated conditions, thus disabling these conditions from being set in the interrupt register and asserting the INTR output. Bit position correspondence is as above.

ENABLE INTERRUPTS

This command writes the associated interrupt mask bit to a one. This enables the corresponding conditions to be set in the interrupt register and asserts the INTR output. Bit position correspondence is as above.

DELAYED COMMANDS

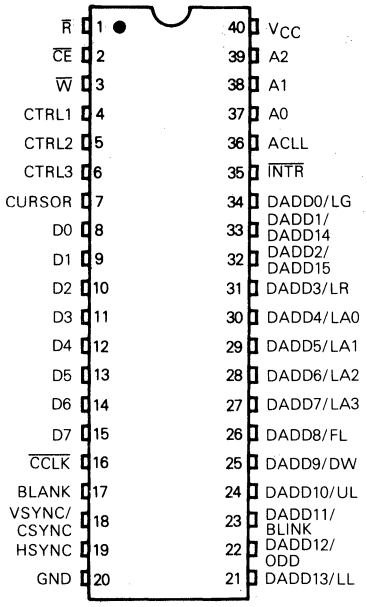
This group of commands is utilized for the independent buffer mode of operation, although the "increment cursor" command can also be used in other modes. With the exception of the "write from cursor to pointer" and "increment cursor" commands, all the commands of this type will be executed immediately or will be delayed depending on when the command is invoked. If invoked during the active screen time, the command is executed at the next horizontal blanking interval. If invoked during a vertical retrace interval or a "display off" state, the command is executed immediately.

The "increment cursor" command is executed immediately after it is issued and requires approximately three $\overline{CCLK}$ periods for completion. The "write from cursor to pointer" command executes during blanking intervals. The AVDC will execute as many writes as possible during each blanking interval. If the command is not completed during the current blanking interval, the command will be held in suspension during the next active portion of the screen and continues during the next blanking interval until the command is completed.

ORDERING INFORMATION (V_{CC} = 5 V ± 5%, T_A = 0°C to 70°C)

Package Type	Frequency	Order Number
Plastic	2.7 MHz	MC2674B3P
P Suffix	4.0 MHz	MC2674B4P

PIN ASSIGNMENT



3

MC6800

8-Bit Microprocessing Unit (MPU)

The MC6800 is a monolithic 8-bit microprocessor forming the central control function for Motorola's M6800 Family. Compatible with TTL, the MC6800, as with all M6800 system parts, requires only one +5.0-volt power supply and no external TTL devices for bus interface.

The MC6800 is capable of addressing 64K bytes of memory with its 16-bit address lines. The 8-bit data bus is bidirectional as well as three-state, making direct memory addressing and multiprocessing applications realizable.

- 8-Bit Parallel Processing
- Bidirectional Data Bus
- 16-Bit Address Bus — 64K Bytes of Addressing
- 72 Instructions — Variable Length
- Seven Addressing Modes — Direct, Relative, Immediate, Indexed, Extended, Implied, and Accumulator
- Variable Length Stack
- Vectored Restart
- Maskable Interrupt Vector
- Separate Nonmaskable Interrupt — Internal Registers Saved in Stack
- Six Internal Registers — Two Accumulators, Index Register, Program Counter, Stack Pointer and Condition Code Register
- Direct Memory Addressing (DMA) and Multiple Processor Capability
- Simplified Clocking Characteristics
- Clock Rates as High as 2.0 MHz
- Simple Bus Interface without TTL
- Halt and Single Instruction Execution Capability

This document contains information on a new product. Specifications and information herein are subject to change without notice.

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range MC6800, MC68A00, MC68B00, MC6800C, MC68A00C	T_A	T_L to T_H −0 to 70 −40 to +85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C

THERMAL RESISTANCE

Rating	Symbol	Value	Unit
Plastic Package Cerdip Package	θ_{JA}	100 60	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A

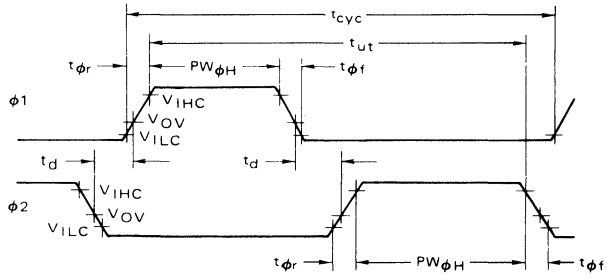
DC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 Vdc, ±5%, V_{SS} = 0, $T_A = T_L$ to T_H unless otherwise noted)

Characteristic		Symbol	Min	Typ	Max	Unit
Input High Voltage	Logic	V_{IH}	$V_{SS}+2.0$	–	V_{CC}	V
	$\phi1, \phi2$	V_{IHC}	$V_{CC}-0.6$	–	$V_{CC}+0.3$	
Input Low Voltage	Logic	V_{IL}	$V_{SS}-0.3$	–	$V_{SS}+0.8$	V
	$\phi1, \phi2$	V_{ILC}	$V_{SS}-0.3$	–	$V_{SS}+0.4$	
Input Leakage Current						
(V_{in} = 0 to 5.25 V, V_{CC} = Max)	Logic	I_{in}	–	1.0	2.5	μA
(V_{in} = 0 to 5.25 V, V_{CC} = 0 V to 5.25 V)	$\phi1, \phi2$		–	–	100	
Hi-Z Input Leakage Current	D0-D7	I_{IZ}	–	2.0	10	μA
(V_{in} = 0.4 to 2.4 V, V_{CC} = Max)	A0-A15, R/$\overline{W}$		–	–	100	
Output High Voltage						
(I_{Load} = −205 μA, V_{CC} = Min)	D0-D7	V_{OH}	$V_{SS}+2.4$	–	–	V
(I_{Load} = −145 μA, V_{CC} = Min)	A0-A15, R/$\overline{W}$, VMA		$V_{SS}+2.4$	–	–	
(I_{Load} = −100 μA, V_{CC} = Min)	BA		$V_{SS}+2.4$	–	–	
Output Low Voltage (I_{Load} = 1.6 mA, V_{CC} = Min)		V_{OL}	–	–	$V_{SS}+0.4$	V
Internal Power Dissipation (Measured at $T_A = T_L$)		P_{INT}	–	0.5	1.0	W
Capacitance						
(V_{in} = 0, T_A = 25°C, f = 1.0 MHz)	$\phi1$	C_{in}	–	25	35	pF
	$\phi2$		–	45	70	
	D0-D7		–	10	12.5	
	Logic Inputs		–	6.5	10	
	A0-A15, R/$\overline{W}$, VMA	C_{out}	–	–	12	pF

CLOCK TIMING (V_{CC} = 5.0 V, ±5%, V_{SS} = 0, $T_A = T_L$ to T_H unless otherwise noted)

Characteristic		Symbol	Min	Typ	Max	Unit
Frequency of Operation	MC6800	f	0.1	–	1.0	MHz
	MC68A00		0.1	–	1.5	
	MC68B00		0.1	–	2.0	
Cycle Time (Figure 1)	MC6800	t_{cyc}	1.000	–	10	µs
	MC68A00		0.666	–	10	
	MC68B00		0.500	–	10	
Clock Pulse Width (Measured at V_{CC} – 0.6 V)	$\phi1, \phi2$ – MC6800	$PW_{\phi H}$	400	–	9500	ns
	$\phi1, \phi2$ – MC68A00		230	–	9500	
	$\phi1, \phi2$ – MC68B00		180	–	9500	
Total $\phi1$ and $\phi2$ Up Time	MC6800	t_{ut}	900	–	–	ns
	MC68A00		600	–	–	
	MC68B00		440	–	–	
Rise and Fall Time (Measured between V_{SS} + 0.4 and V_{CC} – 0.6)		t_r, t_f	–	–	100	ns
Delay Time or Clock Separation (Figure 1) (Measured at $V_{OV} = V_{SS} + 0.6$ V @ $t_r = t_f \leq 100$ ns)		t_d	0	–	9100	ns
(Measured at $V_{OV} = V_{SS} + 1.0$ V @ $t_r = t_f \leq 35$ ns)			0	–	9100	

FIGURE 1 — CLOCK TIMING WAVEFORM



NOTES:
1. Voltage levels shown are $V_L \leq 0.4$, $V_H \geq 2.4$ V, unless otherwise specified.
2. Measurement points shown are 0.8 V and 2.0 V, unless otherwise noted.

READ/WRITE TIMING (Reference Figures 2 through 6, 8, 9, 11, 12 and 13)

Characteristic	Symbol	MC6800			MC68A00			MC68B00			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Address Delay											
C = 90 pF	t_{AD}	–	–	270	–	–	180	–	–	150	ns
C = 30 pF		–	–	250	–	–	165	–	–	135	
Peripheral Read Access Time $t_{acc} = t_{ut} - (t_{AD} + t_{DSR})$	t_{acc}	530	–	—	360	–	—	250	–	—	ns
Data Setup Time (Read)	t_{DSR}	100	–	–	60	–	–	40	–	–	ns
Input Data Hold Time	t_H	10	–	–	10	–	–	10	–	–	ns
Output Data Hold Time	t_H	10	25	–	10	25	–	10	25	–	ns
Address Hold Time (Address, R/$\overline{W}$, VMA)	t_{AH}	30	50	–	30	50	–	30	50	–	ns
Enable High Time for DBE Input	t_{EH}	450	–	–	280	–	–	220	–	–	ns
Data Delay Time (Write)	t_{DDW}	–	–	225	–	–	200	–	–	160	ns
Processor Controls											
Processor Control Setup Time	t_{PCS}	200	–	–	140	–	–	110	–	–	ns
Processor Control Rise and Fall Time	t_{PCr}, t_{PCf}	–	–	100	–	–	100	–	–	100	
Bus Available Delay	t_{BA}	–	–	250	–	–	165	–	–	135	
Hi-Z Enable	t_{TSE}	0	–	40	0	–	40	0	–	40	
Hi-Z Delay	t_{TSD}	–	–	270	–	–	270	–	–	220	
Data Bus Enable Down Time During $\phi1$ Up Time	t_{DBE}	150	–	–	120	–	–	75	–	–	
Data Bus Enable Rise and Fall Times	t_{DBEr}, t_{DBEf}	–	–	25	–	–	25	–	–	25	

FIGURE 2 — READ DATA FROM MEMORY OR PERIPHERALS

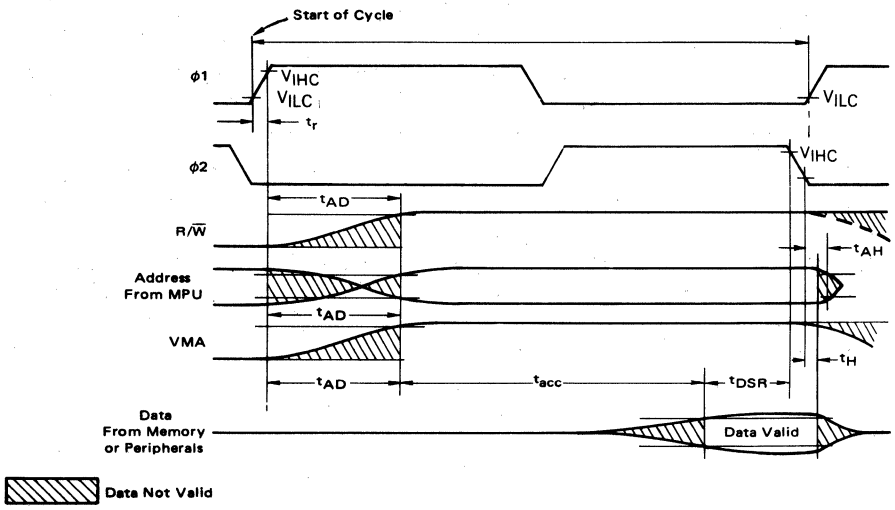
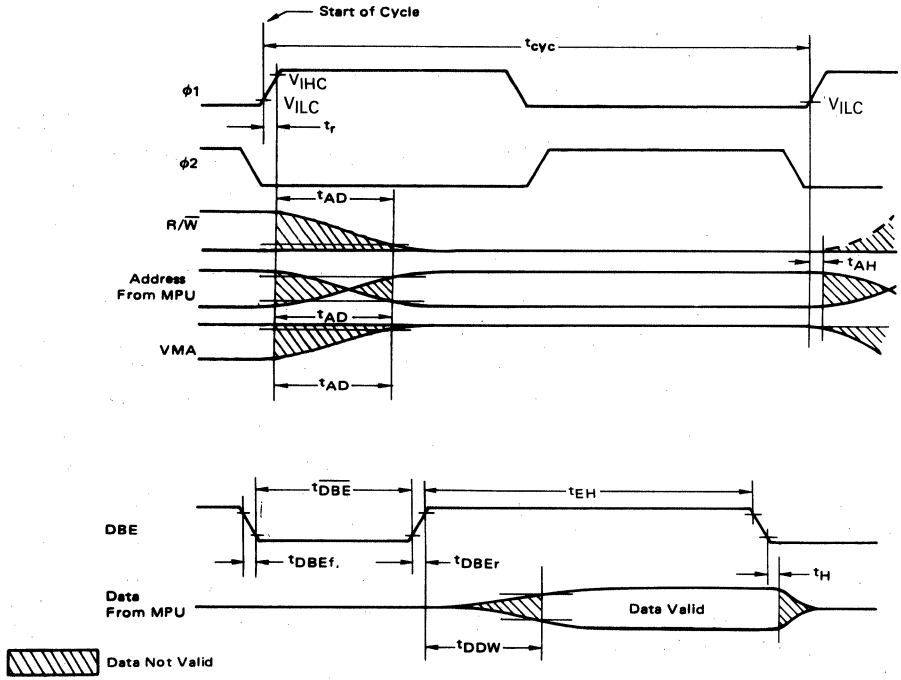


FIGURE 3 — WRITE IN MEMORY OR PERIPHERALS



NOTES:

1. Voltage levels shown are $V_L \leq 0.4$, $V_H \geq 2.4$ V, unless otherwise specified.
2. Measurement points shown are 0.8 V and 2.0 V, unless otherwise noted.

FIGURE 4 — TYPICAL DATA BUS OUTPUT DELAY versus CAPACITIVE LOADING (T_{DDW})

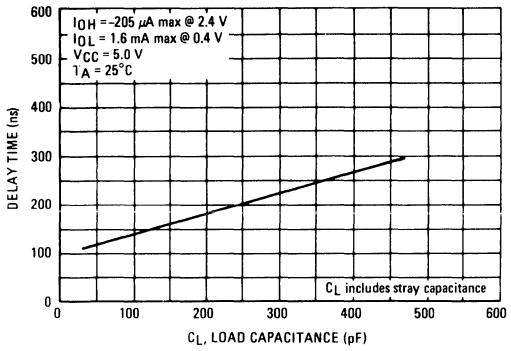
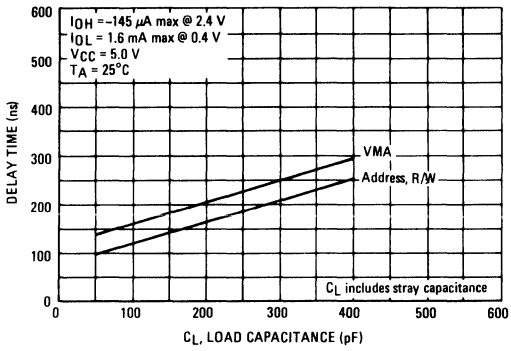
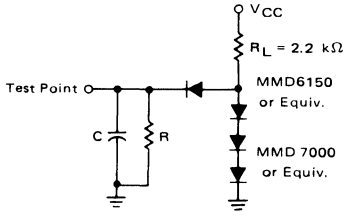


FIGURE 5 — TYPICAL READ/WRITE, VMA, AND ADDRESS OUTPUT DELAY versus CAPACITIVE LOADING (T_{AD})



3

FIGURE 6 — BUS TIMING TEST LOADS



C = 130 pF for D0-D7, E
= 90 pF for A0-A15, R/$\overline{W}$, and VMA
(Except t_{AD2})
= 30 pF for A0-A15, R/$\overline{W}$, and VMA
(t_{AD2} only)
= 30 pF for BA

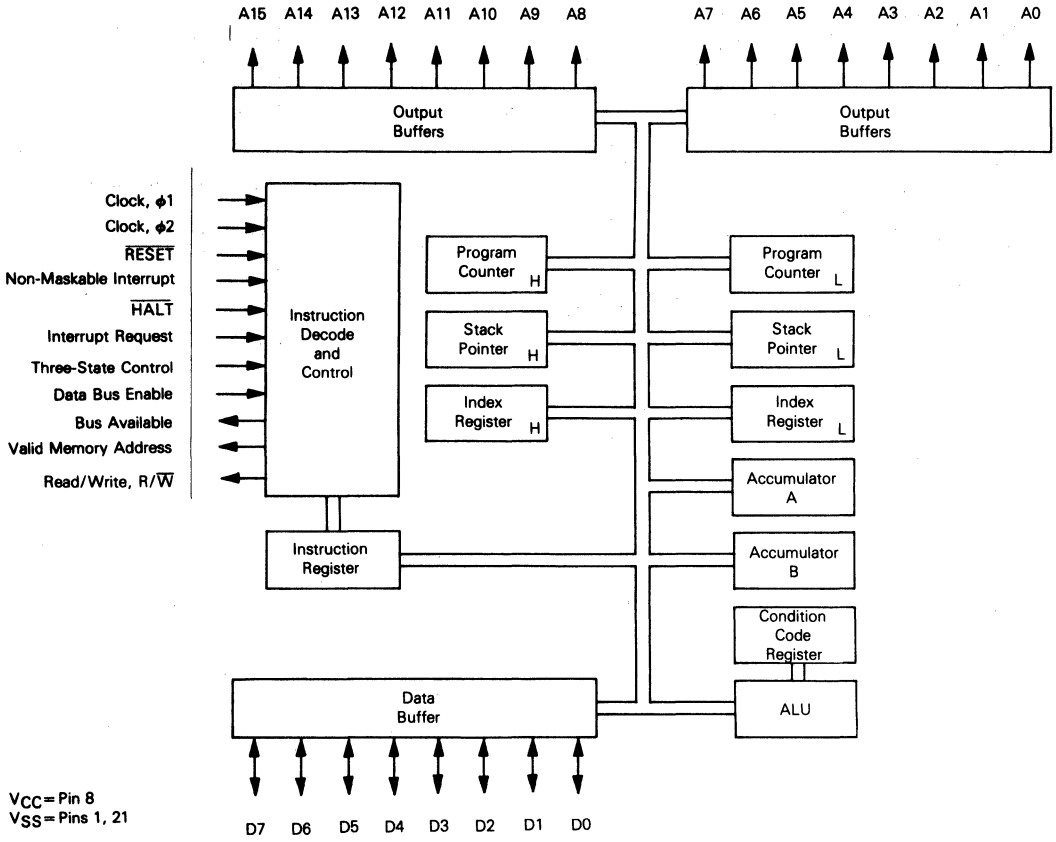
R = 11.7 kΩ for D0-D7
= 16.5 kΩ for A0-A15, R/$\overline{W}$, and VMA
= 24 kΩ for BA

TEST CONDITIONS

The dynamic test load for the Data Bus is 130 pF and one standard TTL load as shown. The Address, R/$\overline{W}$, and VMA outputs are tested under two conditions to allow optimum operation in both buffered and unbuffered systems. The resistor (R) is chosen to insure specified load currents during V_{OH} measurement.

Notice that the Data Bus lines, the Address lines, the Interrupt Request line, and the DBE line are all specified and tested to guarantee 0.4 V of dynamic noise immunity at both "1" and "0" logic levels.

FIGURE 7 — EXPANDED BLOCK DIAGRAM



MPU SIGNAL DESCRIPTION

Proper operation of the MPU requires that certain control and timing signals be provided to accomplish specific functions and that other signal lines be monitored to determine the state of the processor.

Clocks Phase One and Phase Two (ϕ1, ϕ2) — Two pins are used for a two-phase non-overlapping clock that runs at the V_{CC} voltage level.

Figure 1 shows the microprocessor clocks. The high level is specified at V_{IHC} and the low level is specified at V_{ILC}. The allowable clock frequency is specified by f (frequency). The minimum ϕ1 and ϕ2 high level pulse widths are specified by $PW_{\phi H}$ (pulse width high time). To guarantee the required access time for the peripherals, the clock up time, t_{ut}, is specified. Clock separation, t_d, is measured at a maximum voltage of V_{OV} (overlap voltage). This allows for a multitude of clock variations at the system frequency rate.

Address Bus (A0-A15) — Sixteen pins are used for the address bus. The outputs are three-state bus drivers capable of driving one standard TTL load and 90 pF. When the output is turned off, it is essentially an open circuit. This permits the MPU to be used in DMA applications. Putting TSC in its high state forces the Address bus to go into the three-state mode.

Data Bus (D0-D7) — Eight pins are used for the data bus. It is bidirectional, transferring data to and from the memory and peripheral devices. It also has three-state output buffers capable of driving one standard TTL load and 130 pF. Data Bus is placed in the three-state mode when DBE is low.

Data Bus Enable (DBE) — This level sensitive input is the three-state control signal for the MPU data bus and will enable the bus drivers when in the high state. This input is TTL compatible; however in normal operation, it would be driven by the phase two clock. During an MPU read cycle, the data bus drivers will be disabled internally. When it is desired that another device control the data bus, such as in Direct Memory Access (DMA) applications, DBE should be held low.

If additional data setup or hold time is required on an MPU write, the DBE down time can be decreased, as shown in Figure 3 (DBE$\neq\phi$2). The minimum down time for DBE is t_{DBE} as shown. By skewing DBE with respect to E, data setup or hold time can be increased.

Bus Available (BA) — The Bus Available signal will normally be in the low state; when activated, it will go to the high state indicating that the microprocessor has stopped and that the address bus is available. This will occur if the $\overline{HALT}$ line is in the low state or the processor is in the WAIT state as a result of the execution of a WAIT instruction. At such time, all three-state output drivers will go to their off state and other outputs to their normally inactive level. The processor is removed from the WAIT state by the occurrence of a maskable (mask bit I=0) or nonmaskable interrupt. This output is capable of driving one standard TTL load and 30 pF. If TSC is in the high state, Bus Available will be low.

Read/Write (R/$\overline{W}$) — This TTL compatible output signals the peripherals and memory devices wether the MPU is in a Read (high) or Write (low) state. The normal standby state of this signal is Read (high). Three-State Control going high will turn Read/Write to the off (high impedance) state. Also, when the processor is halted, it will be in the off state. This output is capable of driving one standard TTL load and 90 pF.

$\overline{RESET}$ — The $\overline{RESET}$ input is used to reset and start the MPU from a power down condition resulting from a power failure or initial start-up of the processor. This level sensitive input can also be used to reinitialize the machine at any time after start-up.

If a high level is detected in this input, this will signal the MPU to begin the reset sequence. During the reset sequence, the contents of the last two locations (FFFE, FFFF) in memory will be loaded into the Program Counter to point to the beginning of the reset routine. During the reset routine, the interrupt mask bit is set and must be cleared under program control before the MPU can be interrupted by $\overline{IRQ}$. While $\overline{RESET}$ is low (assuming a minimum of 8 clock cycles have occurred) the MPU output signals will be in the following states: VMA=low, BA=low, Data Bus=high impedance, R/$\overline{W}$=high (read state), and the Address Bus will contain the reset address FFFE. Figure 8 illustrates a power up sequence using the $\overline{RESET}$ control line. After the power supply reaches 4.75 V, a minimum of eight clock cycles are required for the processor to stabilize in preparation for restarting. During these eight cycles, VMA will be in an indeterminate state so any devices that are enabled by VMA which could accept a false write during this time (such as battery-backed RAM) must be disabled until VMA is forced low after eight cycles. $\overline{RESET}$ can go high asynchronously with the system clock any time after the eighth cycle.

$\overline{RESET}$ timing is shown in Figure 8. The maximum rise and fall transition times are specified by t_{PCr} and t_{PCf}. If $\overline{RESET}$ is high at t_{PCS} (processor control setup time), as shown in Figure 8, in any given cycle then the restart sequence will begin on the next cycle as shown. The $\overline{RESET}$ control line may also be used to reinitialize the MPU system at any time during its operation. This is accomplished by pulsing RESET low for the duration of a minimum of three complete ϕ2 cycles. The $\overline{RESET}$ pulse can be completely asynchronous with the MPU system clock and will be recognized during ϕ2 if setup time t_{PCS} is met.

Interrupt Request ($\overline{IRQ}$) — This level sensitive input requests that an interrupt sequence be generated within the machine. The processor will wait until it completes the current instruction that is being executed before it recognizes the request. At that time, if the interrupt mask bit in the Condition Code Register is not set, the machine will begin an interrupt sequence. The Index Register, Program Counter, Accumulators, and Condition Code Register are stored away on the stack. Next, the MPU will respond to the interrupt request by setting the interrupt mask bit high so that no further interrupts may occur. At the end of the cycle, a 16-bit address will be loaded that points to a vectoring address which is located in memory locations FFF8 and FFF9. An address loaded at these locations causes the MPU to branch to an interrupt routine in memory. Interrupt timing is shown in Figure 9.

FIGURE 8 — $\overline{\text{RESET}}$ TIMING

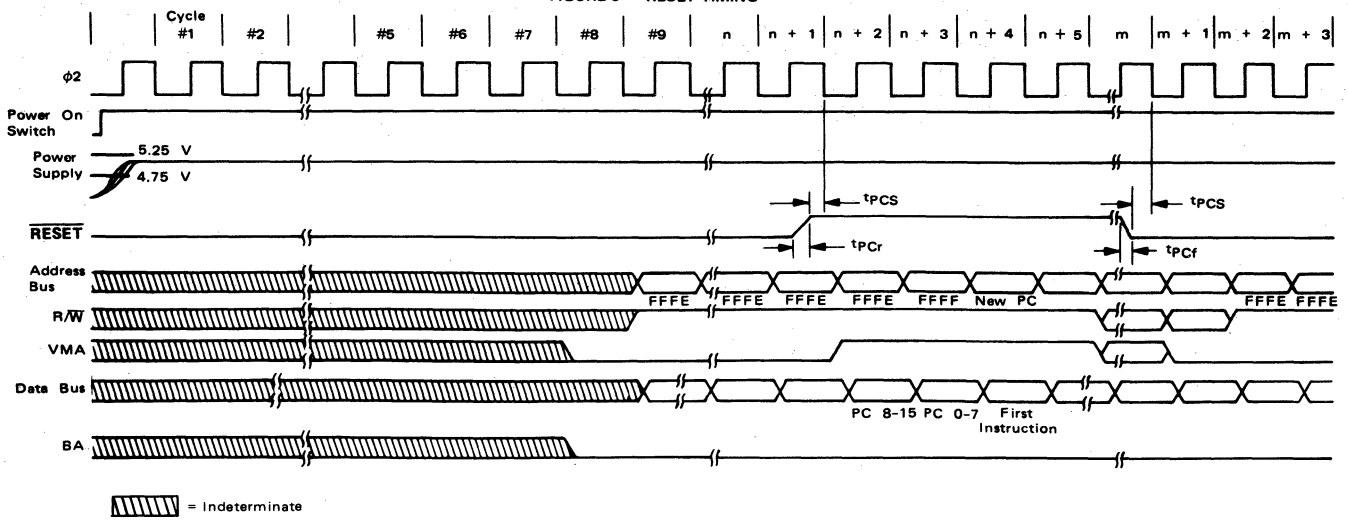
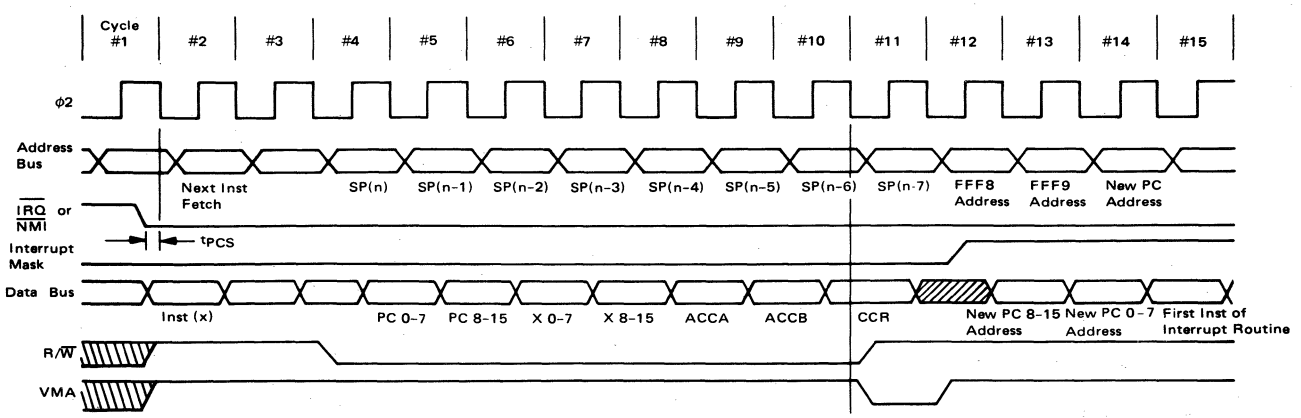


FIGURE 9 — INTERRUPT TIMING



The $\overline{HALT}$ line must be in the high state for interrupts to be serviced. Interrupts will be latched internally while $\overline{HALT}$ is low.

The $\overline{IRQ}$ has a high-impedance pullup device internal to the chip; however, a 3 kΩ external resistor to V_{CC} should be used for wire-OR and optimum control of interrupts.

Non-Maskable Interrupt ($\overline{NMI}$) and Wait for Interrupt (WAI) — The MC6800 is capable of handling two types of interrupts: maskable ($\overline{IRQ}$) as described earlier, and non-maskable ($\overline{NMI}$) which is an edge sensitive input. $\overline{IRQ}$ is maskable by the interrupt mask in the condition code register while $\overline{NMI}$ is not maskable. The handling of these interrupts by the MPU is the same except that each has its own vector address. The behavior of the MPU when interrupted is shown in Figure 9 which details the MPU response to an interrupt while the MPU is executing the control program. The interrupt shown could be either $\overline{IRQ}$ or $\overline{NMI}$ and can be asynchronous with respect to $\phi 2$. The interrupt is shown going low at time t_{PCS} in cycle #1 which precedes the first cycle of an instruction (OP code fetch). This instruction is not executed but instead the Program Counter (PC), Index Register (IX), Accumulators (ACCX), and the Condition Code Register (CCR) are pushed onto the stack.

The Interrupt Mask bit is set to prevent further interrupts. The address of the interrupt service routine is then fetched from FFFC, FFFD for an $\overline{NMI}$ interrupt and from FFF8, FFF9 for an $\overline{IRQ}$ interrupt. Upon completion of the interrupt service routine, the execution of RTI will pull the PC, IX, ACCX, and CCR off the stack; the Interrupt Mask bit is restored to its condition prior to Interrupts (see Figure 10).

Figure 11 is a similar interrupt sequence, except in this case, a WAIT instruction has been executed in preparation for the interrupt. This technique speeds up the MPU's response to the interrupt because the stacking of the PC, IX, ACCX, and the CCR is already done. While the MPU is waiting for the interrupt, Bus Available will go high indicating the following states of the control lines: VMA is low, and the Address Bus, R/$\overline{W}$ and Data Bus are all in the high impedance state. After the interrupt occurs, it is serviced as previously described.

A 3-10 kΩ external resistor to V_{CC} should be used for wire-OR and optimum control of interrupts.

MEMORY MAP FOR INTERRUPT VECTORS

Vector		Description
MS	LS	
FFFE	FFFF	Reset
FFFC	FFFD	Non-Maskable Interrupt
FFFA	FFFB	Software Interrupt
FFF8	FFF9	Interrupt Request

Refer to Figure 10 for program flow for Interrupts.

Three-State Control (TSC) — When the level sensitive Three-State Control (TSC) line is a logic "1", the Address Bus and the R/$\overline{W}$ line are placed in a high-impedance state. VMA and BA are forced low when TSC="1" to prevent false reads or writes on any device enabled by VMA. It is necessary to delay program execution while TSC is held high. This is done by insuring that no transitions of $\phi 1$ (or $\phi 2$) occur during this period. (Logic levels of the clocks are irrelevant so long as they do not change). Since the MPU is a dynamic device, the $\phi 1$ clock can be stopped for a maximum time $PW_{\phi H}$ without destroying data within the MPU. TSC then can be used in a short Direct Memory Access (DMA) application.

Figure 12 shows the effect of TSC on the MPU. TSC must have its transitions at t_{TSE} (three-state enable) while holding $\phi 1$ high and $\phi 2$ low as shown. The Address Bus and R/$\overline{W}$ line will reach the high-impedance state at t_{TSD} (three-state delay), with VMA being forced low. In this example, the Data Bus is also in the high-impedance state while $\phi 2$ is being held low since DBE=$\phi 2$. At this point in time, a DMA transfer could occur on cycles #3 and #4. When TSC is returned low, the MPU Address and R/$\overline{W}$ lines return to the bus. Because it is too late in cycle #5 to access memory, this cycle is dead and used for synchronization. Program execution resumes in cycle #6.

Valid Memory Address (VMA) — This output indicates to peripheral devices that there is a valid address on the address bus. In normal operation, this signal should be utilized for enabling peripheral interfaces such as the PIA and ACIA. This signal is not three-state. One standard TTL load and 90 pF may be directly driven by this active high signal.

$\overline{HALT}$ — When this level sensitive input is in the low state, all activity in the machine will be halted. This input is level sensitive.

The $\overline{HALT}$ line provides an input to the MPU to allow control of program execution by an outside source. If $\overline{HALT}$ is high, the MPU will execute the instructions; if it is low, the MPU will go to a halted or idle mode. A response signal, Bus Available (BA) provides an indication of the current MPU status. When BA is low, the MPU is in the process of executing the control program; if BA is high, the MPU has halted and all internal activity has stopped.

When BA is high, the Address Bus, Data Bus, and R/$\overline{W}$ line will be in a high-impedance state, effectively removing the MPU from the system bus. VMA is forced low so that the floating system bus will not activate any device on the bus that is enabled by VMA.

While the MPU is halted, all program activity is stopped, and if either an $\overline{NMI}$ or $\overline{IRQ}$ interrupt occurs, it will be latched into the MPU and acted on as soon as the MPU is taken out of the halted mode. If a $\overline{RESET}$ command occurs while the MPU is halted, the following states occur: VMA=low, BA=low, Data Bus=high impedance, R/$\overline{W}$=high (read state), and the Address Bus will contain address FFFE as long as $\overline{RESET}$ is low. As soon as the $\overline{RESET}$ line goes high, the MPU will go to locations FFFE and FFFF for the address of the reset routine.

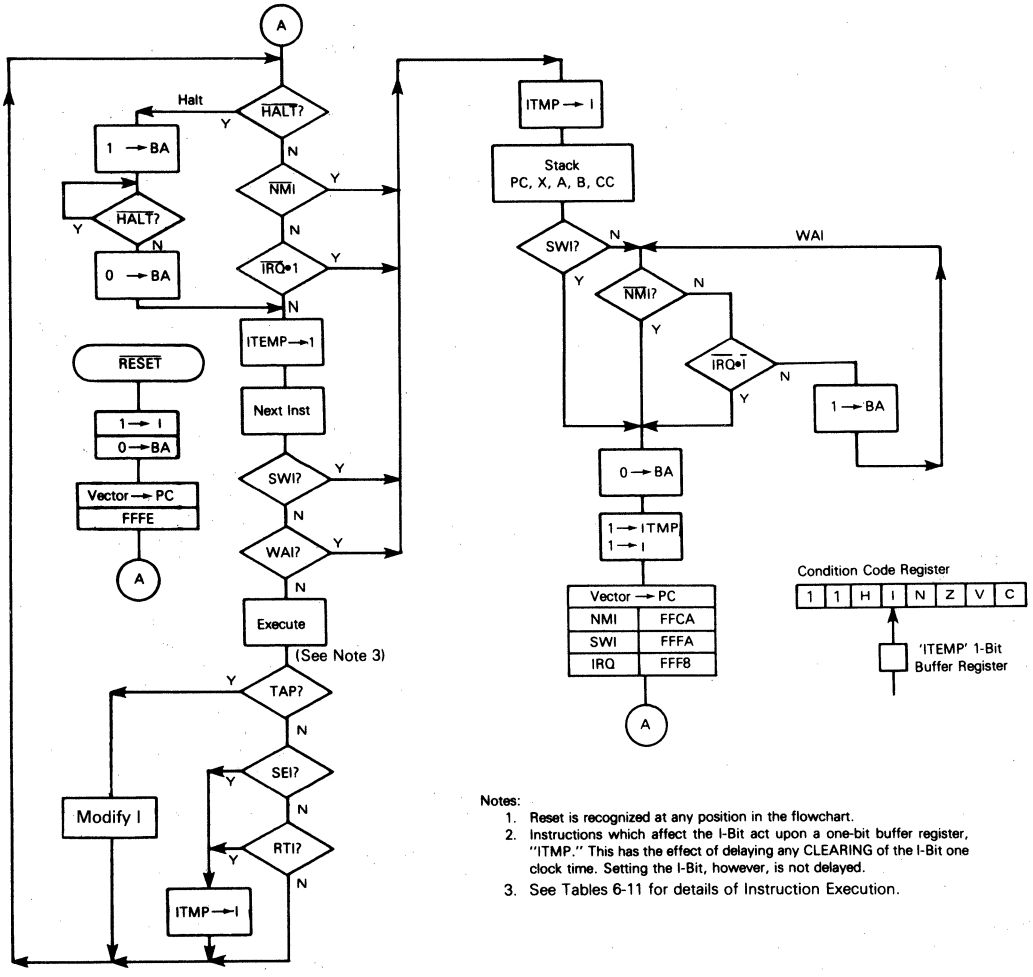
Figure 13 shows the timing relationships involved when halting the MPU. The instruction illustrated is a one byte, 2 cycle instruction such as CLRA. When $\overline{HALT}$ goes low, the MPU will halt after completing execution of the current instruction. The transition of $\overline{HALT}$ must occur t_{PCS} before the trailing edge of $\phi 1$ of the last cycle of an instruction (point A of Figure 13). $\overline{HALT}$ must not go low any time later than the minmum t_{PCS} specified.

The fetch of the OP code by the MPU is the first cycle of the instruction. If $\overline{HALT}$ had not been low at Point A but went low during $\phi 2$ of that cycle, the MPU would have halted after completion of the following instruction. BA will go high by time t_{BA} (bus available delay time) after the last instruction cycle. At this point in time, VMA is low and R/$\overline{W}$, Address Bus, and the Data Bus are in the high-impedance state.

3

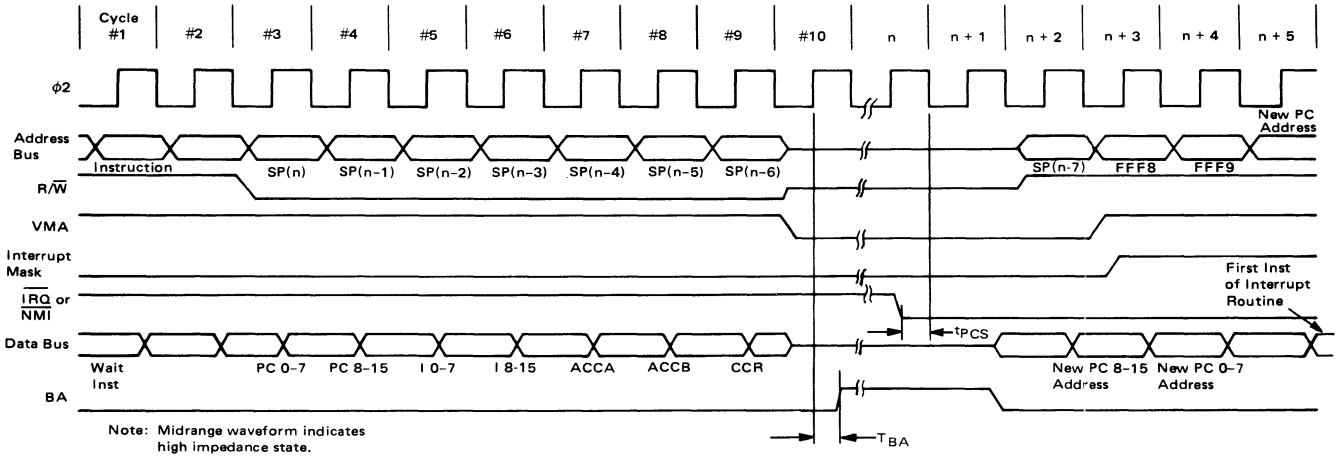
To debug programs it is advantageous to step through programs instruction by instruction. To do this, HALT must be brought high for one MPU cycle and then returned low as shown at point B of Figure 13. Again, the transitions of $\overline{HALT}$ must occur t_{PCS} before the trailing edge of $\phi 1$. BA will go low at t_{BA} after the leading edge of the next $\phi 1$, indicating that the Address Bus, Data Bus, VMA and R/$\overline{W}$ lines are back on the bus. A single byte, 2 cycle instruction such as LSR is used for this example also. During the first cycle, the instruction Y is fetched from address M + 1. BA returns high at t_{BA} on the last cycle of the instruction indicating the MPU is off the bus. If instruction Y had been three cycles, the width of the BA low time would have been increased by one cycle.

FIGURE 10 — MPU FLOWCHART



Notes:
1. Reset is recognized at any position in the flowchart.
2. Instructions which affect the I-Bit act upon a one-bit buffer register, "ITMP." This has the effect of delaying any CLEARING of the I-Bit one clock time. Setting the I-Bit, however, is not delayed.
3. See Tables 6-11 for details of Instruction Execution.

FIGURE 11 — WAIT INSTRUCTION TIMING



Note: Midrange waveform indicates high impedance state.

FIGURE 12 — THREE-STATE CONTROL TIMING

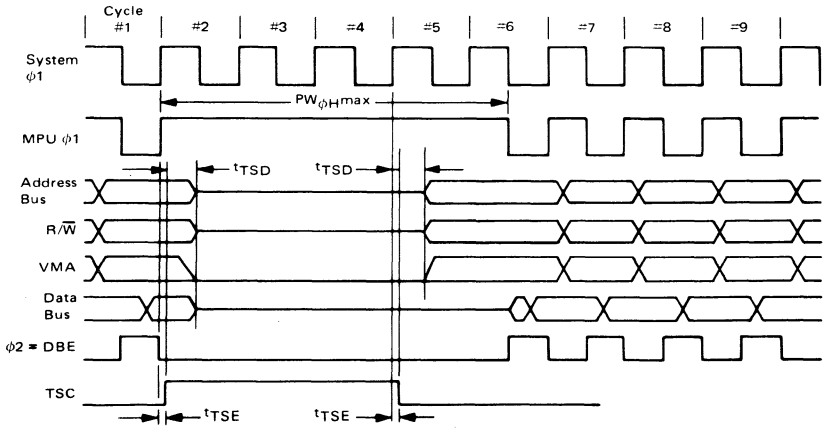
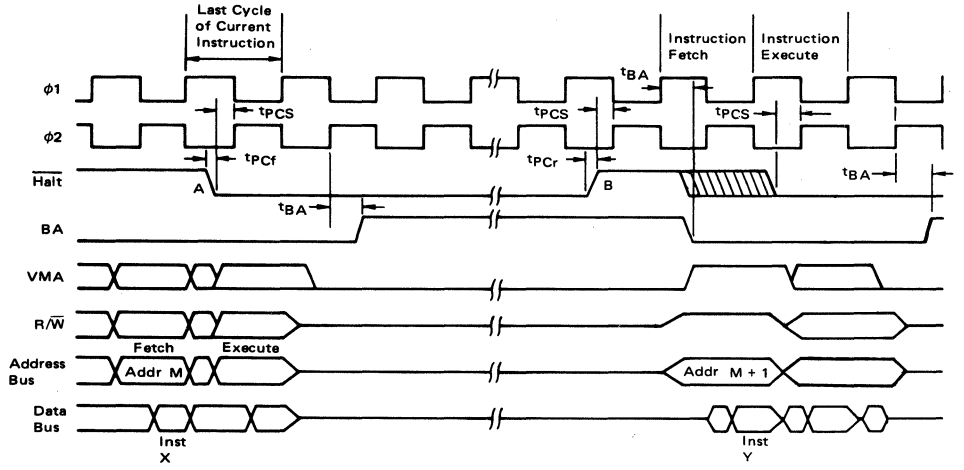


FIGURE 13 – $\overline{\text{HALT}}$ AND SINGLE INSTRUCTION EXECUTION FOR SYSTEM DEBUG



Note: Midrange waveform indicates high impedance state.

MPU REGISTERS

The MPU has three 16-bit registers and three 8-bit registers available for use by the programmer (Figure 14).

Program Counter – The program counter is a two byte (16 bits) register that points to the current program address.

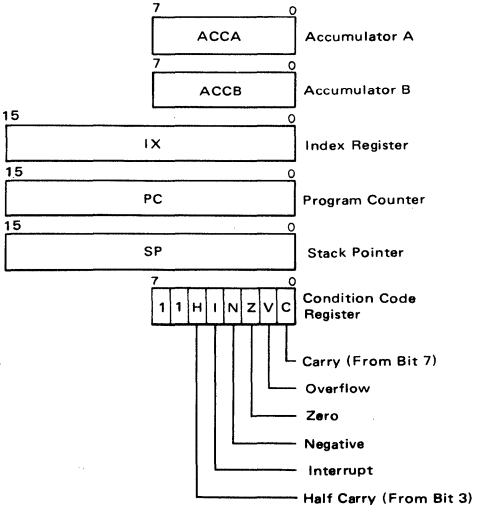
Stack Pointer – The stack ponter is a two byte register that contains the address of the next available location in an external push-down/pop-up stack. This stack is normally a random access Read/Write memory that may have any location (address) that is convenient. In those applications that require storage of information in the stack when power is lost, the stack must be nonvolatile.

Index Register – The index register is a two byte register that is used to store data or a sixteen bit memory address for the Indexed mode of memory addressing.

Accumulators – The MPU contains two 8-bit accumulators that are used to hold operands and results from an arithmetic logic unit (ALU).

Condition Code Register – The condition code register indicates the results of an Arithmetic Logic Unit operation: Negative (N), Zero (Z), Overflow (V), Carry from bit 7 (C), and half carry from bit 3 (H). These bits of the Condition Code Register are used as testable conditions for the conditional branch instructions. Bit 4 is the interrupt mask bit (I). The unused bits of the Condition Code Register (b6 and b7) are ones.

FIGURE 14 – PROGRAMMING MODEL OF THE MICROPROCESSING UNIT



MPU INSTRUCTION SET

The MC6800 instructions are described in detail in the M6800 Programming Manual. This Section will provide a brief introduction and discuss their use in developing MC6800 control programs. The MC6800 has a set of 72 different executable source instructions. Included are binary and decimal arithmetic, logical, shift, rotate, load, store, conditional or unconditional branch, interrupt and stack manipulation instructions.

Each of the 72 executable instructions of the source language assembles into 1 to 3 bytes of machine code. The number of bytes depends on the particular instruction and on the addressing mode. (The addressing modes which are available for use with the various executive instructions are discussed later.)

The coding of the first (or only) byte corresponding to an executable instruction is sufficient to identify the instruction and the addressing mode. The hexadecimal equivalents of the binary codes, which result from the translation of the 72 instructions in all valid modes of addressing, are shown in Table 1. There are 197 valid machine codes, 59 of the 256 possible codes being unassigned.

When an instruction translates into two or three bytes of code, the second byte, or the second and third bytes contain(s) an operand, an address, or information from which an address is obtained during execution.

Microprocessor instructions are often divided into three general classifications: (1) memory reference, so called because they operate on specific memory locations; (2) operating instructions that function without needing a memory reference; (3) I/O instructions for transferring data between the microprocessor and peripheral devices.

In many instances, the MC6800 performs the same operation on both its internal accumulators and the external memory locations. In addition, the MC6800 interface adapters (PIA and ACIA) allow the MPU to treat peripheral devices exactly like other memory locations, hence, no I/O instructions as such are required. Because of these features, other classifications are more suitable for introducing the MC6800's instruction set: (1) Accumulator and memory operations; (2) Program control operations; (3) Condition Code Register operations.

3

TABLE 1 — HEXADECIMAL VALUES OF MACHINE CODES

00	*			40	NEG	A		80	SUB	A	IMM	C0	SUB	B	IMM
01	NOP			41	*			81	CMP	A	IMM	C1	CMP	B	IMM
02	*			42	*			82	SBC	A	IMM	C2	SBC	B	IMM
03	*			43	COM	A		83	*			C3	*		
04	*			44	LSR	A		84	AND	A	IMM	C4	AND	B	IMM
05	*			45	*			85	BIT	A	IMM	C5	BIT	B	IMM
06	TAP			46	ROR	A		86	LDA	A	IMM	C6	LDA	B	IMM
07	TPA			47	ASR	A		87	*			C7	*		
08	INX			48	ASL	A		88	EOR	A	IMM	C8	EOR	B	IMM
09	DEX			49	ROL	A		89	ADC	A	IMM	C9	ADC	B	IMM
0A	CLV			4A	DEC	A		8A	ORA	A	IMM	CA	ORA	B	IMM
0B	SEV			4B	*			8B	ADD	A	IMM	CB	ADD	B	IMM
0C	CLC			4C	INC	A		8C	CPX	A	IMM	CC	*		
0D	SEC			4D	TST	A		8D	BSR		REL	CD	*		
0E	CLI			4E	*			8E	LDS		IMM	CE	LDX		IMM
0F	SEI			4F	CLR	A		8F	*			CF	*		
10	SBA			50	NEG	B		90	SUB	A	DIR	D0	SUB	B	DIR
11	CBA			51	*			91	CMP	A	DIR	D1	CMP	B	DIR
12	*			52	*			92	SBC	A	DIR	D2	SBC	B	DIR
13	*			53	COM	B		93	*			D3	*		
14	*			54	LSR	B		94	AND	A	DIR	D4	AND	B	DIR
15	*			55	*			95	BIT	A	DIR	D5	BIT	B	DIR
16	TAB			56	ROR	B		96	LDA	A	DIR	D6	LDA	B	DIR
17	TBA			57	ASR	B		97	STA	A	DIR	D7	STA	B	DIR
18	*			58	ASL	B		98	EOR	A	DIR	D8	EOR	B	DIR
19	DAA			59	ROL	B		99	ADC	A	DIR	D9	ADC	B	DIR
1A	*			5A	DEC	B		9A	ORA	A	DIR	DA	ORA	B	DIR
1B	ABA			5B	*			9B	ADD	A	DIR	DB	ADD	B	DIR
1C	*			5C	INC	B		9C	CPX		DIR	DC	*		
1D	*			5D	TST	B		9D	*			DD	*		
1E	*			5E	*			9E	LDS		DIR	DE	LDX		DIR
1F	*			5F	CLR	B		9F	STS		DIR	DF	STX		DIR
20	BRA		REL	60	NEG		IND	A0	SUB	A	IND	E0	SUB	B	IND
21	*			61	*			A1	CMP	A	IND	E1	CMP	B	IND
22	BHI		REL	62	*			A2	SBC	A	IND	E2	SBC	B	IND
23	BLS		REL	63	COM		IND	A3	*			E3	*		
24	BCC		REL	64	LSR		IND	A4	AND	A	IND	E4	AND	B	IND
25	BCS		REL	65	*			A5	BIT	A	IND	E5	BIT	B	IND
26	BNE		REL	66	ROR		IND	A6	LDA	A	IND	E6	LDA	B	IND
27	BEQ		REL	67	ASR		IND	A7	STA	A	IND	E7	STA	B	IND
28	BVC		REL	68	ASL		IND	A8	EOR	A	IND	E8	EOR	B	IND
29	BVS		REL	69	ROL		IND	A9	ADC	A	IND	E9	ADC	B	IND
2A	BPL		REL	6A	DEC		IND	AA	ORA	A	IND	EA	ORA	B	IND
2B	BMI		REL	6B	*			AB	ADD	A	IND	EB	ADD	B	IND
2C	BGE		REL	6C	INC		IND	AC	CPX		IND	EC	*		
2D	BLT		REL	6D	TST		IND	AD	JSR		IND	ED	*		
2E	BGT		REL	6E	JMP		IND	AE	LDS		IND	EE	LDX		IND
2F	BLE		REL	6F	CLR		IND	AF	STS		IND	EF	STX		IND
30	TSX			70	NEG		EXT	B0	SUB	A	EXT	F0	SUB	B	EXT
31	INS			71	*			B1	CMP	A	EXT	F1	CMP	B	EXT
32	PUL	A		72	*			B2	SBC	A	EXT	F2	SBC	B	EXT
33	PUL	B		73	COM		EXT	B3	*			F3	*		
34	DES			74	LSR		EXT	B4	AND	A	EXT	F4	AND	B	EXT
35	TXS			75	*			B5	BIT	A	EXT	F5	BIT	B	EXT
36	PSH	A		76	ROR		EXT	B6	LDA	A	EXT	F6	LDA	B	EXT
37	PSH	B		77	ASR		EXT	B7	STA	A	EXT	F7	STA	B	EXT
38	*			78	ASL		EXT	B8	EOR	A	EXT	F8	EOR	B	EXT
39	RTS			79	ROL		EXT	B9	ADC	A	EXT	F9	ADC	B	EXT
3A	*			7A	DEC		EXT	BA	ORA	A	EXT	FA	ORA	B	EXT
3B	RTI			7B	*			BB	ADD	A	EXT	FB	ADD	B	EXT
3C	*			7C	INC		EXT	BC	CPX		EXT	FC	*		
3D	*			7D	TST		EXT	BD	JSR		EXT	FD	*		
3E	WAI			7E	JMP		EXT	BE	LDS		EXT	FE	LDX		EXT
3F	SWI			7F	CLR		EXT	BF	STS		EXT	FF	STX		EXT

Notes: 1. Addressing Modes:

A	= Accumulator A
B	= Accumulator B
REL	= Relative
IND	= Indexed
IMM	= Immediate
DIR	= Direct

2. Unassigned code indicated by "*".

TABLE 2 — ACCUMULATOR AND MEMORY OPERATIONS

OPERATIONS	MNEMONIC	IMMED OP	IMMED ~	IMMED #	DIRECT OP	DIRECT ~	DIRECT #	INDEX OP	INDEX ~	INDEX #	EXTND OP	EXTND ~	EXTND #	IMPLIED OP	IMPLIED ~	IMPLIED #	BOOLEAN/ARITHMETIC OPERATION (All register labels refer to contents)	5 H	4 I	3 N	2 Z	1 V	0 C
Add	ADDA	8B	2	2	9B	3	2	AB	5	2	BB	4	3				A + M → A	↕	•	↕	↕	↕	↕
	ADDB	CB	2	2	DB	3	2	EB	5	2	FB	4	3				B + M → B	↕	•	↕	↕	↕	↕
Add Acmltrs	ABA													1B	2	1	A + B → A	↕	•	↕	↕	↕	↕
Add with Carry	ADCA	89	2	2	99	3	2	A9	5	2	B9	4	3				A + M + C → A	↕	•	↕	↕	↕	↕
	ADCB	C9	2	2	D9	3	2	E9	5	2	F9	4	3				B + M + C → B	↕	•	↕	↕	↕	↕
And	ANDA	84	2	2	94	3	2	A4	5	2	B4	4	3				A · M → A	•	•	↕	↕	R	•
	ANDB	C4	2	2	D4	3	2	E4	5	2	F4	4	3				B · M → B	•	•	↕	↕	R	•
Bit Test	BITA	85	2	2	95	3	2	A5	5	2	B5	4	3				A · M	•	•	↕	↕	R	•
	BITB	C5	2	2	D5	3	2	E5	5	2	F5	4	3				B · M	•	•	↕	↕	R	•
Clear	CLR							6F	7	2	7F	6	3				00 → M	•	•	R	S	R	R
	CLRA													4F	2	1	00 → A	•	•	R	S	R	R
	CLRB													5F	2	1	00 → B	•	•	R	S	R	R
Compare	CMPA	81	2	2	91	3	2	A1	5	2	B1	4	3				A − M	•	•	↕	↕	↕	↕
	CMPB	C1	2	2	D1	3	2	E1	5	2	F1	4	3				B − M	•	•	↕	↕	↕	↕
Compare Acmltrs	CBA													11	2	1	A − B	•	•	↕	↕	↕	↕
Complement, 1's	COM							63	7	2	73	6	3				M̄ → M	•	•	↕	↕	R	S
	COMA													43	2	1	Ā → A	•	•	↕	↕	R	S
	COMB													53	2	1	B̄ → B	•	•	↕	↕	R	S
Complement, 2's (Negate)	NEG							60	7	2	70	6	3				00 − M → M	•	•	↕	↕	①	②
	NEGA													40	2	1	00 − A → A	•	•	↕	↕	①	②
	NEGB													50	2	1	00 − B → B	•	•	↕	↕	①	②
Decimal Adjust, A	DAA													19	2	1	Converts Binary Add. of BCD Characters into BCD Format	•	•	↕	↕	↕	③
Decrement	DEC							6A	7	2	7A	6	3				M − 1 → M	•	•	↕	↕	4	•
	DECA													4A	2	1	A − 1 → A	•	•	↕	↕	4	•
	DECB													5A	2	1	B − 1 → B	•	•	↕	↕	4	•
Exclusive OR	EORA	88	2	2	98	3	2	A8	5	2	B8	4	3				A⊕M → A	•	•	↕	↕	R	•
	EORB	C8	2	2	D8	3	2	E8	5	2	F8	4	3				B⊕M → B	•	•	↕	↕	R	•
Increment	INC							6C	7	2	7C	6	3				M + 1 → M	•	•	↕	↕	⑤	•
	INCA													4C	2	1	A + 1 → A	•	•	↕	↕	⑤	•
	INCB													5C	2	1	B + 1 → B	•	•	↕	↕	⑤	•
Load Acmltr	LDAA	86	2	2	96	3	2	A6	5	2	B6	4	3				M → A	•	•	↕	↕	R	•
	LDAB	C6	2	2	D6	3	2	E6	5	2	F6	4	3				M → B	•	•	↕	↕	R	•
Or, Inclusive	ORAA	8A	2	2	9A	3	2	AA	5	2	BA	4	3				A + M → A	•	•	↕	↕	R	•
	ORAB	CA	2	2	DA	3	2	EA	5	2	FA	4	3				B + M → B	•	•	↕	↕	R	•
Push Data	PSHA													36	4	1	A → M_{SP}, SP − 1 → SP	•	•	•	•	•	•
	PSHB													37	4	1	B → M_{SP}, SP − 1 → SP	•	•	•	•	•	•
Pull Data	PULA													32	4	1	SP + 1 → SP, M_{SP} → A	•	•	•	•	•	•
	PULB													33	4	1	SP + 1 → SP, M_{SP} → B	•	•	•	•	•	•
Rotate Left	ROL							69	7	2	79	6	3				M	•	•	↕	↕	⑥	↕
	ROLA													49	2	1	A	•	•	↕	↕	⑥	↕
	ROLB													59	2	1	B — C b7 b0	•	•	↕	↕	⑥	↕
Rotate Right	ROR							66	7	2	76	6	3				M	•	•	↕	↕	⑥	↕
	RORA													46	2	1	A	•	•	↕	↕	⑥	↕
	RORB													56	2	1	B — C b7 b0	•	•	↕	↕	⑥	↕
Shift Left, Arithmetic	ASL							68	7	2	78	6	3				M	•	•	↕	↕	⑥	↕
	ASLA													48	2	1	A — 0	•	•	↕	↕	⑥	↕
	ASLB													58	2	1	B — C b7 b0	•	•	↕	↕	⑥	↕
Shift Right, Arithmetic	ASR							67	7	2	77	6	3				M	•	•	↕	↕	⑥	↕
	ASRA													47	2	1	A	•	•	↕	↕	⑥	↕
	ASRB													57	2	1	B — b7 b0 C	•	•	↕	↕	⑥	↕
Shift Right, Logic	LSR							64	7	2	74	6	3				M	•	•	R	↕	⑥	↕
	LSRA													44	2	1	A — 0	•	•	R	↕	⑥	↕
	LSRB													54	2	1	B — b7 b0 C	•	•	R	↕	⑥	↕
Store Acmltr.	STAA				97	4	2	A7	6	2	B7	5	3				A → M	•	•	↕	↕	R	•
	STAB				D7	4	2	E7	6	2	F7	5	3				B → M	•	•	↕	↕	R	•
Subtract	SUBA	80	2	2	90	3	2	A0	5	2	B0	4	3				A − M → A	•	•	↕	↕	↕	↕
	SUBB	C0	2	2	D0	3	2	E0	5	2	F0	4	3				B − M → B	•	•	↕	↕	↕	↕
Subtract Acmltrs.	SBA													10	2	1	A − B → A	•	•	↕	↕	↕	↕
Subtr. with Carry	SBCA	82	2	2	92	3	2	A2	5	2	B2	4	3				A − M − C → A	•	•	↕	↕	↕	↕
	SBCB	C2	2	2	D2	3	2	E2	5	2	F2	4	3				B − M − C → B	•	•	↕	↕	↕	↕
Transfer Acmltrs	TAB													16	2	1	A → B	•	•	↕	↕	R	•
	TBA													17	2	1	B → A	•	•	↕	↕	R	•
Test, Zero or Minus	TST							6D	7	2	7D	6	3				M − 00	•	•	↕	↕	R	R
	TSTA													4D	2	1	A − 00	•	•	↕	↕	R	R
	TSTB													5D	2	1	B − 00	•	•	↕	↕	R	R
																		H	I	N	Z	V	C

LEGEND:

- OP Operation Code (Hexadecimal);
- ~ Number of MPU Cycles;
- # Number of Program Bytes;
- + Arithmetic Plus;
- − Arithmetic Minus;
- · Boolean AND;
- M_{SP} Contents of memory location pointed to be Stack Pointer;
- + Boolean Inclusive OR;
- ⊕ Boolean Exclusive OR;
- M Complement of M;
- → Transfer Into;
- 0 Bit = Zero;
- 00 Byte = Zero;

CONDITION CODE SYMBOLS:

- H Half-carry from bit 3;
- I Interrupt mask
- N Negative (sign bit)
- Z Zero (byte)
- V Overflow, 2's complement
- C Carry from bit 7
- R Reset Always
- S Set Always
- ↕ Test and set if true, cleared otherwise
- • Not Affected

CONDITION CODE REGISTER NOTES:
(Bit set if test is true and cleared otherwise)

1. (Bit V) Test: Result = 10000000?
2. (Bit C) Test: Result = 00000000?
3. (Bit C) Test: Decimal value of most significant BCD Character greater than nine? (Not cleared if previously set.)
4. (Bit V) Test: Operand = 10000000 prior to execution?
5. (Bit V) Test: Operand = 01111111 prior to execution?
6. (Bit V) Test: Set equal to result of N⊕C after shift has occurred.

Note — Accumulator addressing mode instructions are included in the column for IMPLIED addressing

PROGRAM CONTROL OPERATIONS

Program Control operation can be subdivided into two categories: (1) Index Register/Stack Pointer instructions; (2) Jump and Branch operations.

Index Register/Stack Pointer Operations

The instructions for direct operation on the MPU's Index Register and Stack Pointer are summarized in Table 3. Decrement (DEX, DES), increment (INX, INS), load (LDX, LDS), and store (STX, STS) instructions are provided for both. The Compare instruction, CPX, can be used to compare the Index Register to a 16-bit value and update the Condition Code Register accordingly.

The TSX instruction causes the Index Register to be loaded with the address of the last data byte put onto the "stack." The TXS instruction loads the Stack Pointer with a value equal to one less than the current contents of the Index Register. This causes the next byte to be pulled from the "stack" to come from the location indicated by the Index Register. The utility of these two instructions can be clarified by describing the "stack" concept relative to the M6800 system.

The "stack" can be thought of as a sequential list of data stored in the MPU's read/write memory. The Stack Pointer contains a 16-bit memory address that is used to access the list from one end on a last-in-first-out (LIFO) basis in contrast to the random access mode used by the MPU's other addressing modes.

The MC6800 instruction set and interrupt structure allow extensive use of the stack concept for efficient handling of data movement, subroutines and interrupts. The instructions can be used to establish one or more "stacks" anywhere in read/write memory. Stack length is limited only by the amount of memory that is made available.

Operation of the Stack Pointer with the Push and Pull instructions is illustrated in Figures 15 and 16. The Push instruction (PSHA) causes the contents of the indicated accumulator (A in this example) to be stored in memory at the location indicated by the Stack Pointer. The Stack Pointer is automatically decremented by one following the storage operation and is "pointing" to the next empty stack location. The Pull instruction (PULA or PULB) causes the last byte stacked to be loaded into the appropriate accumulator. The Stack Pointer is automatically incremented by one just prior to the data transfer so that it will point to the last byte stacked rather than the next empty location. Note that the PULL instruction does not "remove" the data from memory; in the example, 1A is still in location (m + 1) following execution of PULA. A subsequent PUSH instruction would overwrite that location with the new "pushed" data.

Execution of the Branch to Subroutine (BSR) and Jump to Subroutine (JSR) instructions cause a return address to be saved on the stack as shown in Figures 18 through 20. The stack is decremented after each byte of the return address is pushed onto the stack. For both of these instructions, the return address is the memory location following the bytes of code that correspond to the BSR and JSR instruction. The code required for BSR or JSR may be either two or three bytes, depending on whether the JSR is in the indexed (two bytes) or the extended (three bytes) addressing mode. Before it is stacked, the Program Counter is automatically incremented the correct number of times to be pointing at the location of the next instruction. The Return from Subroutine Instruction, RTS, causes the return address to be retrieved and loaded into the Program Counter as shown in Figure 21.

There are several operations that cause the status of the MPU to be saved on the stack. The Software Interrupt (SWI) and Wait for Interrupt (WAI) instructions as well as the maskable ($\overline{IRQ}$) and non-maskable ($\overline{NMI}$) hardware interrupts all cause the MPU's internal registers (except for the Stack Pointer itself) to be stacked as shown in Figure 23. MPU status is restored by the Return from Interrupt, RTI, as shown in Figure 22.

Jump and Branch Operation

The Jump and Branch instructions are summarized in Table 4. These instructions are used to control the transfer or operation from one point to another in the control program.

The No Operation instruction, NOP, while included here, is a jump operation in a very limited sense. Its only effect is to increment the Program Counter by one. It is useful during program development as a "stand-in" for some other instruction that is to be determined during debug. It is also used for equalizing the execution time through alternate paths in a control program.

TABLE 3 — INDEX REGISTER AND STACK POINTER INSTRUCTIONS

		IMMED			DIRECT			INDEX			EXTND			IMPLIED				COND. CODE REG.					
																		5	4	3	2	1	0
POINTER OPERATIONS	MNEMONIC	OP	~	#	OP	~	#	OP	~	#	OP	~	#	OP	~	#	BOOLEAN/ARITHMETIC OPERATION	H	I	N	Z	V	C
Compare Index Reg	CPX	8C	3	3	9C	4	2	AC	6	2	BC	5	3				$X_H - M, X_L - (M+1)$	•	•	①	↕	②	•
Decrement Index Reg	DEX													09	4	1	$X - 1 \rightarrow X$	•	•	•	↕	•	•
Decrement Stack Pntr	DES													34	4	1	$SP - 1 \rightarrow SP$	•	•	•	•	•	•
Increment Index Reg	INX													08	4	1	$X + 1 \rightarrow X$	•	•	•	↕	•	•
Increment Stack Pntr	INS													31	4	1	$SP + 1 \rightarrow SP$	•	•	•	•	•	•
Load Index Reg	LDX	CE	3	3	DE	4	2	EE	6	2	FE	5	3				$M \rightarrow X_H, (M+1) \rightarrow X_L$	•	•	③	↕	R	•
Load Stack Pntr	LDS	8E	3	3	9E	4	2	AE	6	2	BE	5	3				$M \rightarrow SP_H, (M+1) \rightarrow SP_L$	•	•	③	↕	R	•
Store Index Reg	STX				DF	5	2	EF	7	2	FF	6	3				$X_H \rightarrow M, X_L \rightarrow (M+1)$	•	•	③	↕	R	•
Store Stack Pntr	STS				9F	5	2	AF	7	2	BF	6	3				$SP_H \rightarrow M, SP_L \rightarrow (M+1)$	•	•	③	↕	R	•
Indx Reg → Stack Pntr	TXS													35	4	1	$X - 1 \rightarrow SP$	•	•	•	•	•	•
Stack Pntr → Indx Reg	TSX													30	4	1	$SP + 1 \rightarrow X$	•	•	•	•	•	•

① (Bit N) Test: Sign bit of most significant (MS) byte of result = 1?
② (Bit V) Test: 2's complement overflow from subtraction of ms bytes?
③ (Bit N) Test: Result less than zero? (Bit 15 = 1)

FIGURE 15 — STACK OPERATION, PUSH INSTRUCTION

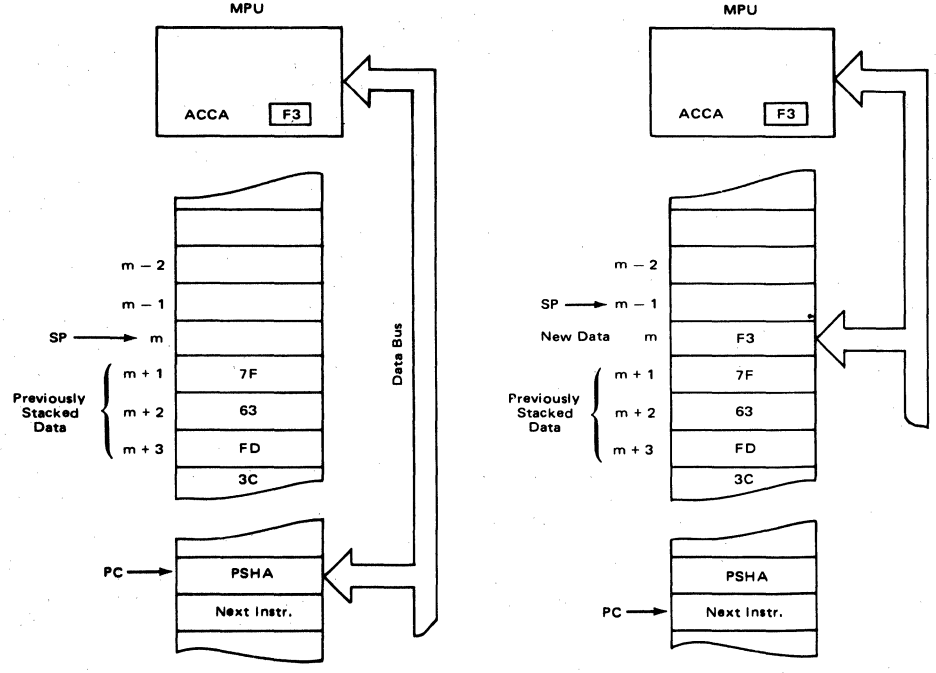


FIGURE 16 — STACK OPERATION, PULL INSTRUCTION

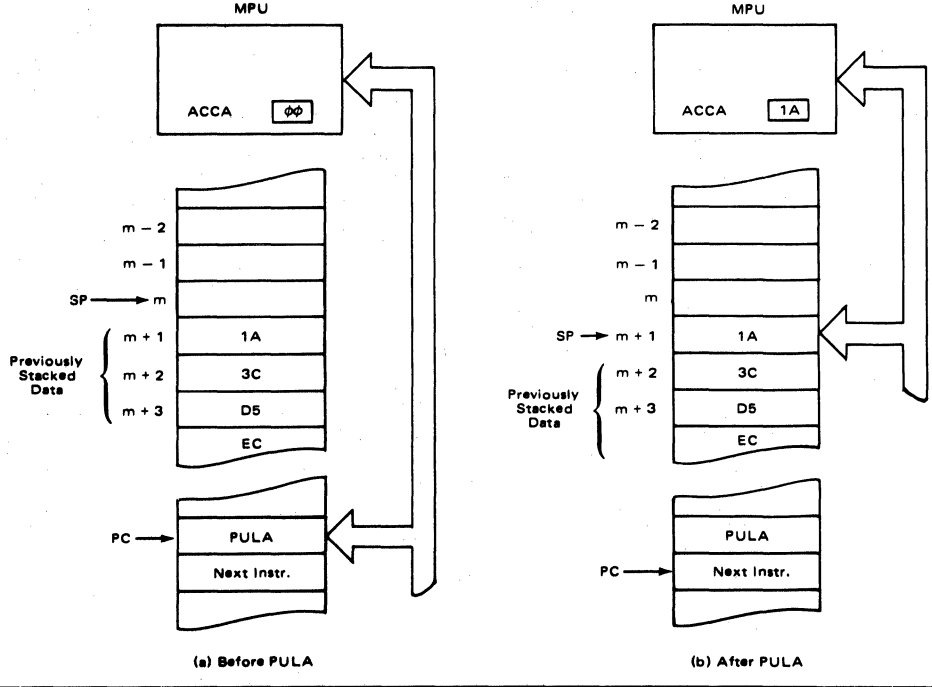


TABLE 4 — JUMP AND BRANCH INSTRUCTIONS

OPERATIONS	MNEMONIC	RELATIVE OP	RELATIVE ~	RELATIVE #	INDEX OP	INDEX ~	INDEX #	EXTND OP	EXTND ~	EXTND #	IMPLIED OP	IMPLIED ~	IMPLIED #	BRANCH TEST	COND. CODE REG. 5 H	4 I	3 N	2 Z	1 V	0 C
Branch Always	BRA	20	4	2										None	•	•	•	•	•	•
Branch If Carry Clear	BCC	24	4	2										C = 0	•	•	•	•	•	•
Branch If Carry Set	BCS	25	4	2										C = 1	•	•	•	•	•	•
Branch If = Zero	BEQ	27	4	2										Z = 1	•	•	•	•	•	•
Branch If ≥ Zero	BGE	2C	4	2										N ⊕ V = 0	•	•	•	•	•	•
Branch If > Zero	BGT	2E	4	2										Z + (N ⊕ V) = 0	•	•	•	•	•	•
Branch If Higher	BHI	22	4	2										C + Z = 0	•	•	•	•	•	•
Branch If ≤ Zero	BLE	2F	4	2										Z + (N ⊕ V) = 1	•	•	•	•	•	•
Branch If Lower Or Same	BLS	23	4	2										C + Z = 1	•	•	•	•	•	•
Branch If < Zero	BLT	2D	4	2										N ⊕ V = 1	•	•	•	•	•	•
Branch If Minus	BMI	2B	4	2										N = 1	•	•	•	•	•	•
Branch If Not Equal Zero	BNE	26	4	2										Z = 0	•	•	•	•	•	•
Branch If Overflow Clear	BVC	28	4	2										V = 0	•	•	•	•	•	•
Branch If Overflow Set	BVS	29	4	2										V = 1	•	•	•	•	•	•
Branch If Plus	BPL	2A	4	2										N = 0	•	•	•	•	•	•
Branch To Subroutine	BSR	8D	8	2											•	•	•	•	•	•
Jump	JMP				6E	4	2	7E	3	3				See Special Operations	•	•	•	•	•	•
Jump To Subroutine	JSR				AD	8	2	BD	9	3					•	•	•	•	•	•
No Operation	NOP										01	2	1	Advances Prog. Cntr. Only	•	•	•	•	•	•
Return From Interrupt	RTI										3B	10	1		———			①	———	
Return From Subroutine	RTS										39	5	1		•	•	•	•	•	•
Software Interrupt	SWI										3F	12	1	See Special Operations	•	•	•	•	•	•
Wait for Interrupt*	WAI										3E	9	1		•	②	•	•	•	•

*WAI puts Address Bus, R/W, and Data Bus in the three-state mode while VMA is held low.

① (All) Load Condition Code Register from Stack. (See Special Operations)
② (Bit 1) Set when interrupt occurs. If previously set, a Non-Maskable Interrupt is required to exit the wait state.

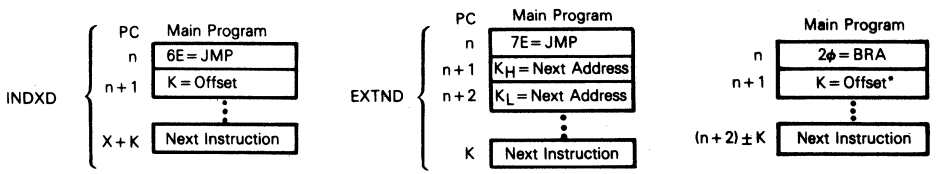
3

Execution of the Jump Instruction, JMP, and Branch Always, BRA, affects program flow as shown in Figure 17. When the MPU encounters the Jump (Indexed) instruction, it adds the offset to the value in the Index Register and uses the result as the address of the next instruction to be executed. In the extended addressing mode, the address of the next instruction to be executed is fetched from the two locations immediately following the JMP instruction. The Branch Always (BRA) instruction is similar to the JMP (extended) instruction except that the relative addressing mode applies and the branch is limited to the range within −125 or +127 bytes of the branch instruction itself. The opcode for the BRA instruction requires one less byte than JMP (extended) but takes one more cycle to execute.

The effect on program flow for the Jump to Subroutine (JSR) and Branch to Subroutine (BSR) is shown in Figures 18 through 20. Note that the Program Counter is properly incremented to be pointing at the correct return address before it is stacked. Operation of the Branch to Subroutine and Jump to Subroutine (extended) instruction is similar except for the range. The BSR instruction requires less opcode than JSR (2 bytes versus 3 bytes) and also executes one cycle faster than JSR. The Return from Subroutine, RTS, is used as the end of a subroutine to return to the main program as indicated in Figure 21.

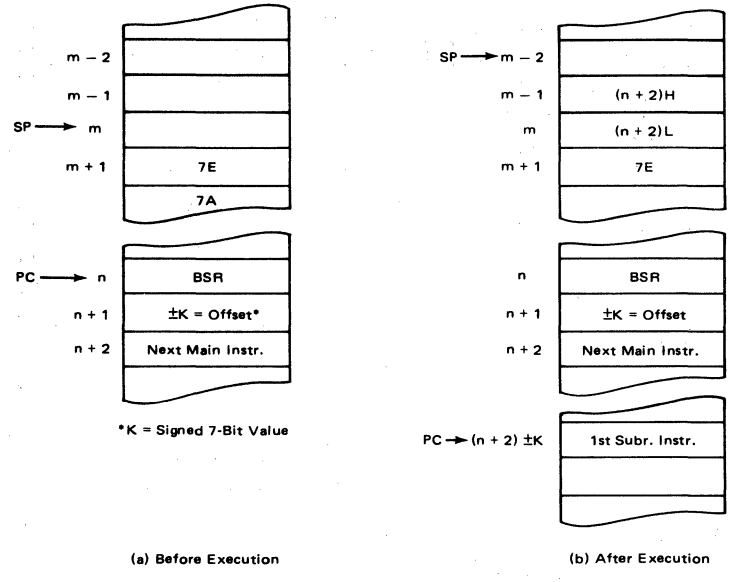
The effect of executing the Software Interrupt, SWI, and the Wait for Interrupt, WAI, and their relationship to the hardware interrupts is shown in Figure 22. SWI causes the MPU contents to be stacked and then fetches the starting address of the interrupt routine from the memory locations that respond to the addresses FFFA and FFFB. Note that as in the case of the subroutine instructions, the Program Counter is incremented to point at the correct return address before being stacked. The Return from Interrupt instruction, RTI, (Figure 22) is used at the end of an interrupt routine to restore control to the main program. The SWI instruction is useful for inserting break points in the control program, that is, it can be used to stop operation and put the MPU registers in memory where they can be examined. The WAI instruction is used to decrease the time required to service a hardware interrupt; it stacks the MPU contents and then waits for the interrupt to occur, effectively removing the stacking time from a hardware interrupt sequence.

FIGURE 17 — PROGRAM FLOW FOR JUMP AND BRANCH INSTRUCTIONS



(a) Jump (b) Branch

FIGURE 18 — PROGRAM FLOW FOR BSR



3

FIGURE 19 — PROGRAM FLOW FOR JSR (EXTENDED)

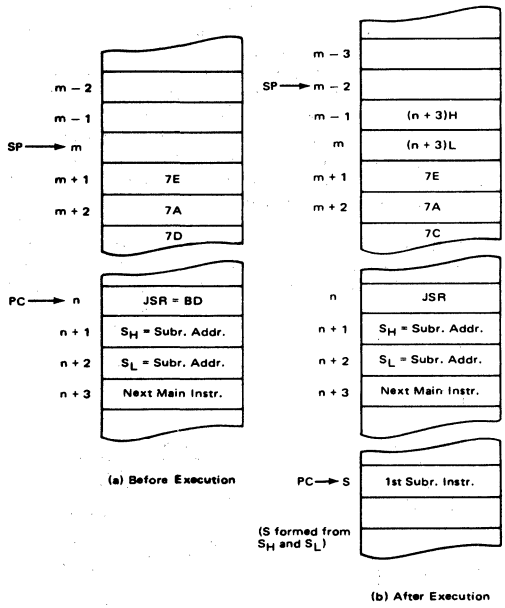


FIGURE 20 — PROGRAM FLOW FOR JSR (INDEXED)

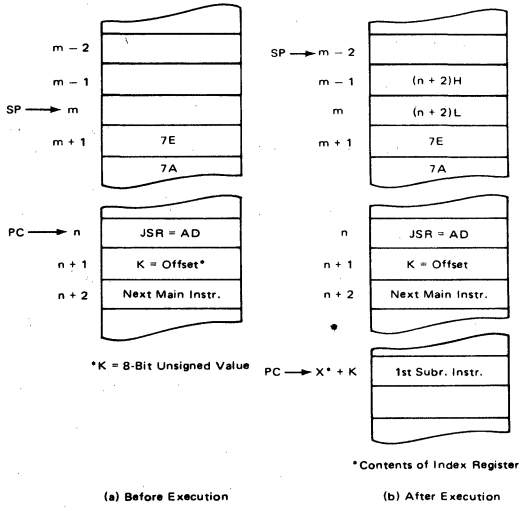


FIGURE 21 — PROGRAM FLOW FOR RTS

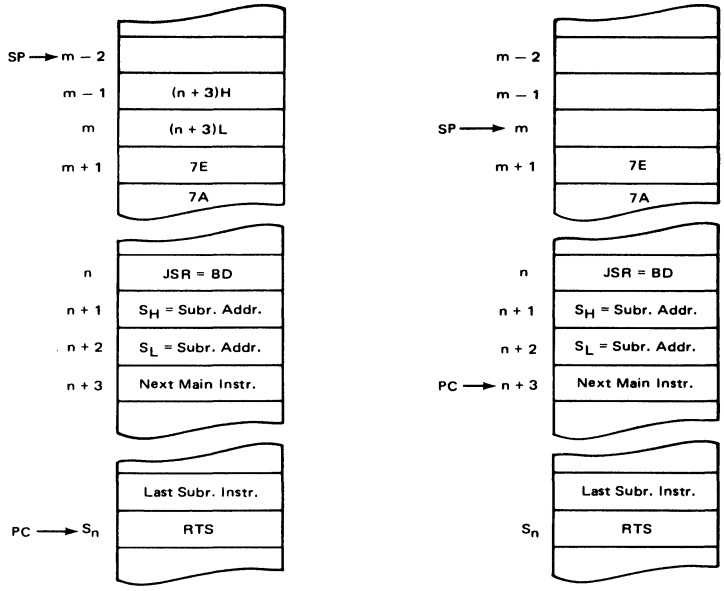


FIGURE 22 — PROGRAM FLOW FOR RTI

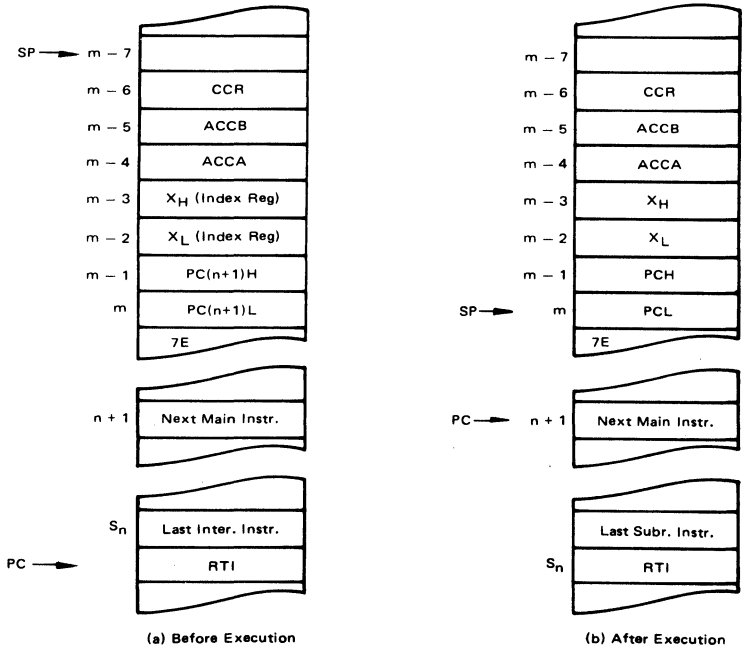
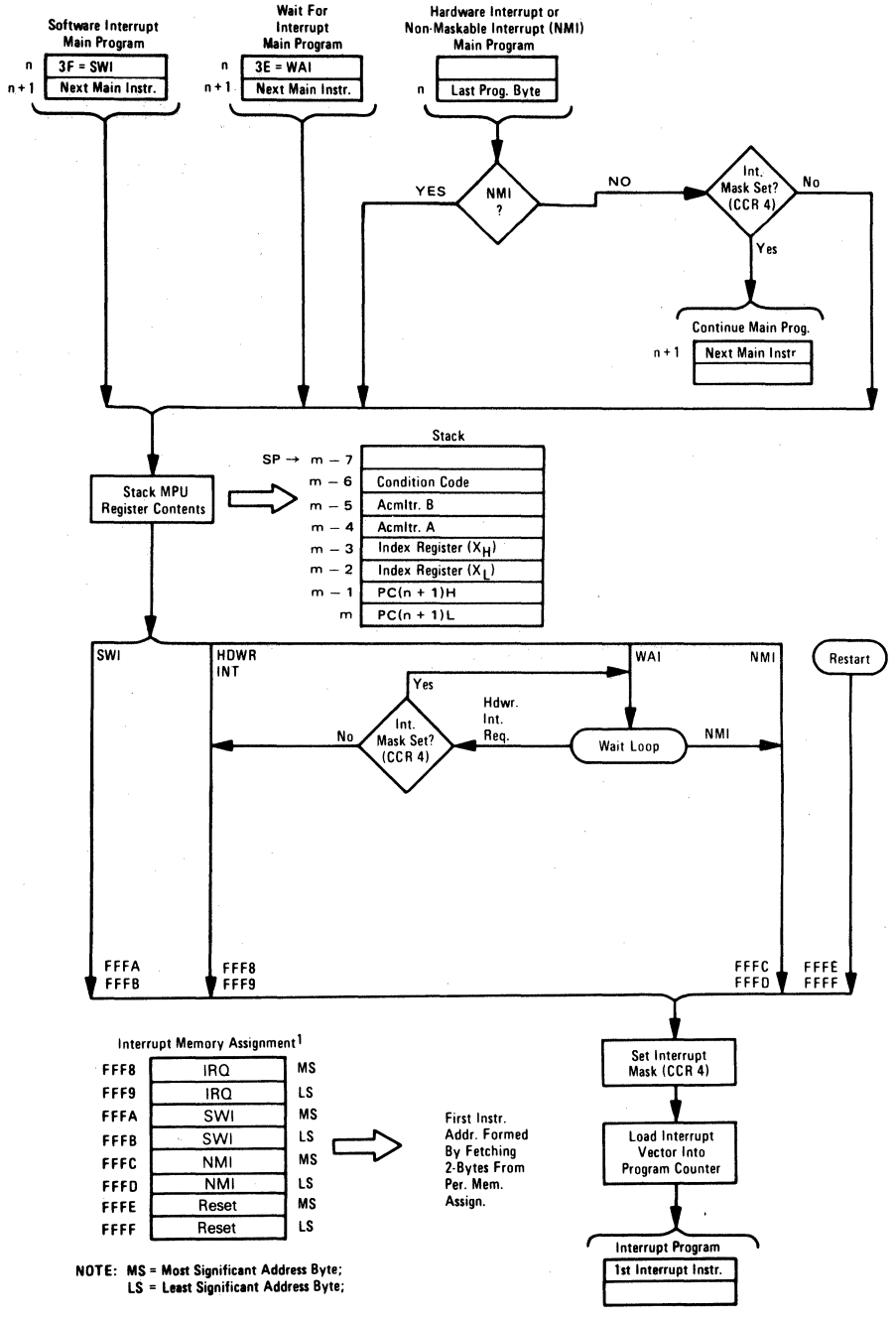


FIGURE 23 — PROGRAM FLOW FOR INTERRUPTS



NOTE: MS = Most Significant Address Byte;
LS = Least Significant Address Byte;

FIGURE 24 — CONDITIONAL BRANCH INSTRUCTIONS

BMI :	N = 1 ;	BEQ :	Z = 1 ;
BPL :	N = ϕ ;	BNE :	Z = ϕ ;
BVC :	V = ϕ ;	BCC :	C = ϕ ;
BVS :	V = 1 ;	BCS :	C = 1 ;
BHI :	C + Z = ϕ ;	BLT :	N⊕V = 1 ;
BLS :	C + Z = 1 ;	BGE :	N⊕V = ϕ ;
BLE :	Z + (N⊕V) = 1 ;		
BGT :	Z + (N⊕V) = ϕ ;		

The conditional branch instructions, Figure 24, consists of seven pairs of complementary instructions. They are used to test the results of the preceding operation and either continue with the next instruction in sequence (test fails) or cause a branch to another point in the program (test succeeds).

Four of the pairs are used for simple tests of status bits N, Z, V, and C:

1. Branch on Minus (BMI) and Branch On Plus (BPL) tests the sign bit, N, to determine if the previous result was negative or positive, respectively.

2. Branch On Equal (BEQ) and Branch On Not Equal (BNE) are used to test the zero status bit, Z, to determine whether or not the result of the previous operation was equal to zero. These two instructions are useful following a Compare (CMP) instruction to test for equality between an accumulator and the operand. They are also used following the Bit Test (BIT) to determine whether or not the same bit positions are set in an accumulator and the operand.

3. Branch On Overflow Clear (BVC) and Branch On Overflow Set (BVS) tests the state of the V bit to determine if the previous operation caused an arithmetic overflow.

4. Branch On Carry Clear (BCC) and Branch On Carry Set (BCS) tests the state of the C bit to determine if the previous operation caused a carry to occur. BCC and BCS are useful for testing relative magnitude when the values being tested are regarded as unsigned binary numbers, that is, the values are in the range 00 (lowest) to FF (highest). BCC following a comparison (CMP) will cause a branch if the (unsigned) value in the accumulator is higher than or the same as the value of the operand. Conversely, BCS will cause a branch if the accumulator value is lower than the operand.

The fifth complementary pair, Branch On Higher (BHI) and Branch On Lower or Same (BLS) are, in a sense, complements to BCC and BCS. BHI tests for both C and $Z = 0$; if used following a CMP, it will cause a branch if the value in the accumulator is higher than the operand. Conversely, BLS will cause a branch if the unsigned binary value in the accumulator is lower than or the same as the operand.

The remaining two pairs are useful in testing results of operations in which the values are regarded as signed two's complement numbers. This differs from the unsigned binary case in the following sense: in unsigned, the orientation is higher or lower; in signed two's complement, the comparison is between larger or smaller where the range of values is between -128 and $+127$.

Branch On Less Than Zero (BLT) and Branch On Greater Than Or Equal Zero (BGE) test the status bits for $N \oplus V = 1$ and $N \oplus V = 0$, respectively. BLT will always cause a branch following an operation in which two negative numbers were added. In addition, it will cause a branch following a CMP in which the value in the accumulator was negative and the operand was positive. BLT will never cause a branch following a CMP in which the accumulator value was positive and the operand negative. BGE, the complement to BLT, will cause a branch following operations in which two positive values were added or in which the result was zero.

The last pair, Branch On Less Than Or Equal Zero (BLE) and Branch On Greater Than Zero (BGT) test the status bits for $Z \oplus (N + V) = 1$ and $Z \oplus (N + V) = 0$, respectively. The action of BLE is identical to that for BLT except that a branch will also occur if the result of the previous result was zero. Conversely, BGT is similar to BGE except that no branch will occur following a zero result.

CONDITION CODE REGISTER OPERATIONS

The Condition Code Register (CCR) is a 6-bit register within the MPU that is useful in controlling program flow during system operation. The bits are defined in Figure 25.

The instructions shown in Table 5 are available to the user for direct manipulation of the CCR.

A CLI-WAI instruction sequence operated properly, with early MC6800 processors, only if the preceding instruction was odd (Least Significant Bit = 1). Similarly it was advisable to precede any SEI instruction with an odd opcode — such as NOP. These precautions are not necessary for MC6800 processors indicating manufacture in November 1977 or later.

Systems which require an interrupt window to be opened under program control should use a CLI-NOP-SEI sequence rather than CLI-SEI.

FIGURE 25 — CONDITION CODE REGISTER BIT DEFINITION

b_5	b_4	b_3	b_2	b_1	b_0
H	I	N	Z	V	C

H = Half-carry; set whenever a carry from b_3 to b_4 of the result is generated by ADD, ABA, ADC; cleared if no b_3 to b_4 carry; not affected by other instructions.

I = Interrupt Mask; set by hardware or software interrupt or SEI instruction; cleared by CLI instruction. (Normally not used in arithmetic operations.) Restored to a zero as a result of an RT1 instruction if I_m stored on the stacked is low.

N = Negative; set if high order bit (b_7) of result is set; cleared otherwise.

Z = Zero; set if result = 0; cleared otherwise.

V = Overlow; set if there was arithmetic overflow as a result of the operation; cleared otherwise.

C = Carry; set if there was a carry from the most significant bit (b_7) of the result; cleared otherwise.

TABLE 5 — CONDITION CODE REGISTER INSTRUCTIONS

		IMPLIED				COND. CODE REG.					
						5	4	3	2	1	0
OPERATIONS	MNEMONIC	OP	~	#	BOOLEAN OPERATION	H	I	N	Z	V	C
Clear Carry	CLC	0C	2	1	0→C	•	•	•	•	•	R
Clear Interrupt Mask	CLI	0E	2	1	0→I	•	R	•	•	•	•
Clear Overflow	CLV	0A	2	1	0→V	•	•	•	•	R	•
Set Carry	SEC	0D	2	1	1→C	•	•	•	•	•	S
Set Interrupt Mask	SEI	0F	2	1	1→I	•	S	•	•	•	•
Set Overflow	SEV	0B	2	1	1→V	•	•	•	•	S	•
Acmltr A→CCR	TAP	06	2	1	A→CCR	①					
CCR→Acmltr A	TPA	07	2	1	CCR→A	•	•	•	•	•	•

R = Reset
S = Set
• = Not affected
① (ALL) Set according to the contents of Accumulator A.

ADDRESSING MODES

The MPU operates on 8-bit binary numbers presented to it via the data bus. A given number (byte) may represent either data or an instruction to be executed, depending on where it is encountered in the control program. The M6800 has 72 unique instructions; however, it recognizes and takes action on 197 of the 256 possibilities that can occur using an 8-bit word length. This larger number of instructions results from the fact that many of the executive instructions have more than one addressing mode.

These addressing modes refer to the manner in which the program causes the MPU to obtain its instructions and data. The programmer must have a method for addressing the MPU's internal registers and all of the external memory locations.

Selection of the desired addressing mode is made by the user as the source statements are written. Translation into appropriate opcode then depends on the method used. If manual translation is used, the addressing mode is inherent in the opcode. For example, the immediate, direct, indexed, and extended modes may all be used with the ADD instruction. The proper mode is determined by selecting (hexadecimal notation) 8B, 9B, AB, or BB, respectively.

The source statement format includes adequate information for the selection if an assembler program is used to generate the opcode. For instance, the immediate mode is selected by the assembler whenever it encounters the "#" symbol in the operand field. Similarly, an "X" in the operand field causes the indexed mode to be selected. Only the relative mode applies to the branch instructions; therefore, the mnemonic instruction itself is enough for the assemble to determine addressing mode.

For the instructions that use both Direct and Extended modes, the Assembler selects the Direct mode if the operand value is in the range 0-255 and Extended otherwise. There are a number of instructions for which the Extended mode is valid but the Direct is not. For these instructions, the Assembler automatically selects the Extended mode even if the operand is in the 0-255 range. The addressing modes are summarized in Figure 26.

Inherent (Includes "Accumulator Addressing" Mode)

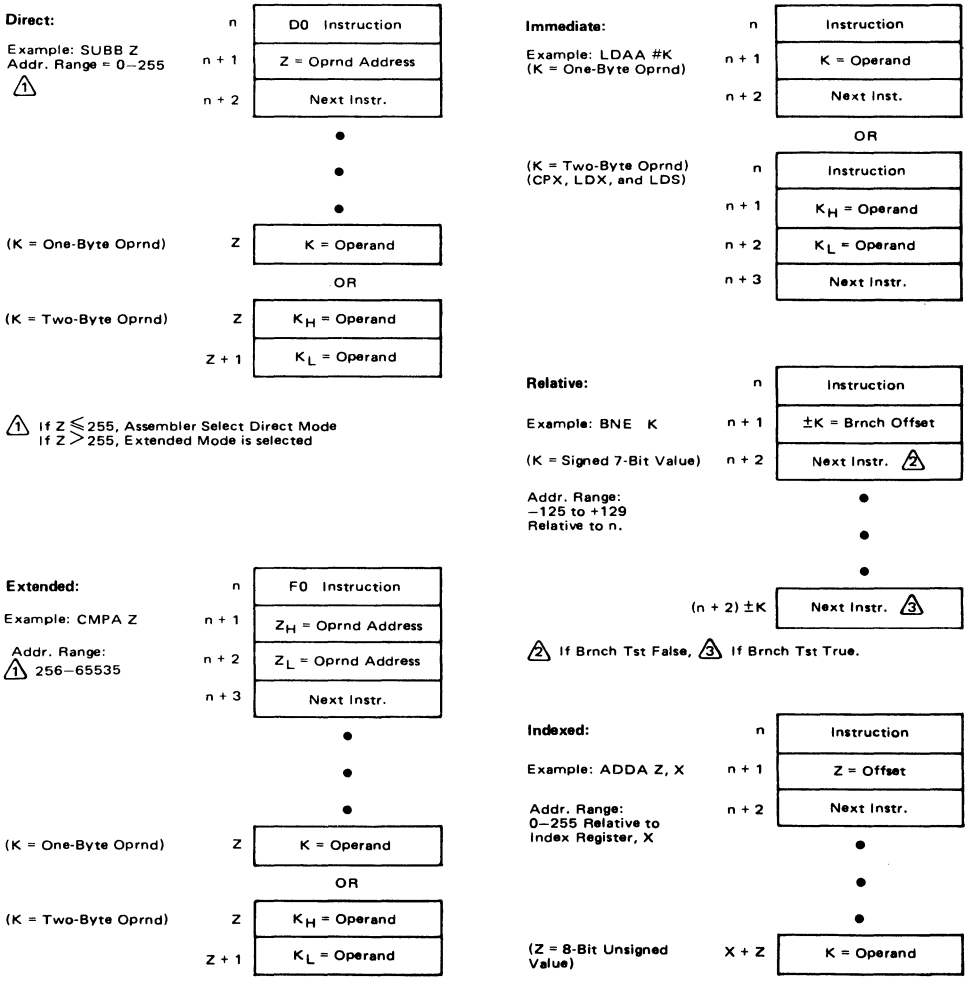
The successive fields in a statement are normally separated by one or more spaces. An exception to this rule occurs for instructions that use dual addressing in the operand field and for instructions that must distinguish between the two accumulators. In these cases, A and B are "operands" but the space between them and the operator may be omitted. This is commonly done, resulting in apparent four character mnemonics for those instructions.

The addition instruction, ADD, provides an example of dual addressing in the operand field:

Operator	Operand	Comment
ADDA	MEM12	ADD CONTENTS OF MEM12 TO ACCA
or		
ADDB	MEM12	ADD CONTENTS OF MEM12 TO ACCB

The example used earlier for the test instruction, TST, also applies to the accumulators and uses the "accumulator addressing mode" to designate which of the two accumulators is being tested:

FIGURE 26 — ADDRESSING MODE SUMMARY



Operator	Comment
TSTB	TEST CONTENTS OF ACCB

or

TSTA	TEST CONTENTS OF ACCA

A number of the instructions either alone or together with an accumulator operand contain all of the address information that is required, that is, "inherent" in the instruction itself. For instance, the instruction ABA causes the MPU to add the contents of accmulators A and B together and place the result in accumulator A. The instruction INCB, another example of "accumulator addressing," causes the contents of accumulator B to be increased by one. Similarly, INX, increment the Index Register, causes the contents of the Index Register to be increased by one.

Program flow for instructions of this type is illustrated in Figures 27 and 28. In these figures, the general case is shown on the left and a specific example is shown on the right. Numerical examples are in decimal notation. Instructions of this type require only one byte of opcode. Cycle-by-cycle operation of the inherent mode is shown in Table 6.

3

Immediate Addressing Mode — In the Immediate addressing mode, the operand is the value that is to be operated on. For instance, the instruction

Operator	Operand	Comment
LDAA	#25	LOAD 25 INTO ACCA

causes the MPU to "immediately load accumulator A with the value 25"; no further address reference is required. The Immediate mode is selected by preceding the operand value with the "#" symbol. Program flow for this addressing mode is illustrated in Figure 29.

The operand format allows either properly defined symbols or numerical values. Except for the instructions CPX, LDX, and LDS, the operand may be any value in the range 0 to 255. Since Compare Index Register (CPX), Load Index Register (LDX), and Load Stack Pointer (LDS), require 16-bit values, the immediate mode for these three instructions require two-byte operands. In the Immediate addressing mode, the "address" of the operand is effectively the memory location immediately following the instruction itself. Table 7 shows the cycle-by-cycle operation for the immediate addressing mode.

Direct and Extended Addressing Modes — In the Direct and Extended modes of addressing, the operand field of the source statement is the *address* of the value that is to be operated on. The Direct and Extended modes differ only in the range of memory locations to which they can direct the MPU. Direct addressing generates a single 8-bit operand and, hence, can address only memory locations 0 through 255; a two byte operand is generated for Extended addressing, enabling the MPU to reach the remaining memory locations, 256 through 65535. An example of Direct addressing and its effect on program flow is illustrated in Figure 30.

The MPU, after encountering the opcode for the instruction LDAA (Direct) at memory location 5004 (Program Counter = 5004), looks in the next location, 5005, for the address of the operand. It then sets the program counter equal to the value found there (100 in the example) and fetches the operand, in this case a value to be loaded into accumulator A, from that location. For instructions requiring a two-byte operand such as LDX (Load the Index Register), the operand bytes would be retrieved from locations 100 and 101. Table 8 shows the cycle-by-cycle operation for the direct mode of addressing.

Extended addressing, Figure 31, is similar except that a two-byte address is obtained from locations 5007 and 5008 after the LDAB (Extended) opcode shows up in location 5006. Extended addressing can be thought of as the "standard" addressing mode, that is, it is a method of reaching any place in memory. Direct addressing, since only one address byte is required, provides a faster method of processing data and generates fewer bytes of control code. In most applications, the direct addressing range, memory locations 0-255, are reserved for RAM. They are used for data buffering and temporary storage of system variables, the area in which faster addressing is of most value. Cycle-by-cycle operation is shown in Table 9 for Extended Addressing.

FIGURE 27 — INHERENT ADDRESSING

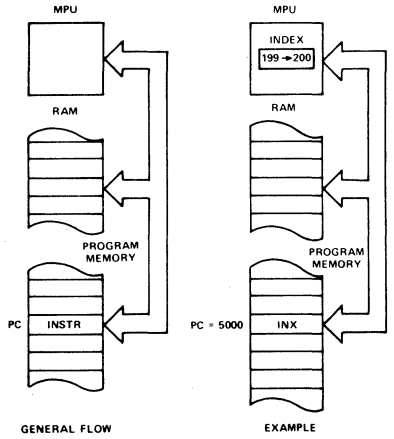
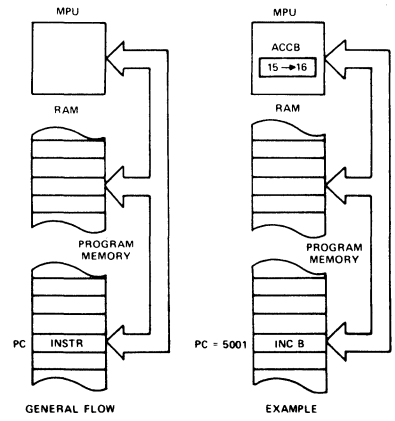


FIGURE 28 — ACCUMULATOR ADDRESSING



Relative Address Mode — In both the Direct and Extended modes, the address obtained by the MPU is an absolute numerical address. The Relative addressing mode, implemented for the MPU's branch instructions, specifies a memory location relative to the Program Counter's current location. Branch instructions generate two bytes of machine code, one for the instruction opcode and one for the "relative" address (see Figure 32). Since it is desirable to be able to branch in either direction, the 8-bit address byte is interpreted as a signed 7-bit value; the 8th bit of the operand is treated as a sign bit, "0" = plus and "1" = minus. The remaining seven bits represent the numerical value. This results in a relative addressing range of ± 127 with respect to the location of the branch instruction itself. However, the branch range is computed with respect to the next instruction that would be executed if the branch conditions are not satisfied. Since two bytes are generated, the next instruction is located at PC + 2. If D is defined as the address of the branch destination, the range is then:

$$(PC + 2) - 127 \leq D \leq (PC + 2) + 127$$

or

$$PC - 125 \leq D \leq PC + 129$$

that is, the destination of the branch instruction must be within −125 to +129 memory locations of the branch instruction itself. For transferring control beyond this range, the unconditional jump (JMP), jump to subroutine (JSR), and return from subroutine (RTS) are used.

In Figure 32, when the MPU encounters the opcode for BEQ (Branch if result of last instruction was zero), it tests the Zero bit in the Condition Code Register. If that bit is "0," indicating a non-zero result, the MPU continues execution with the next instruction (in location 5010 in Figure 32). If the previous result was zero, the branch condition is satisfied and the MPU adds the offset, 15 in this case, to PC + 2 and branches to location 5025 for the next instruction.

The branch instructions allow the programmer to efficiently direct the MPU to one point or another in the control program depending on the outcome of test results. Since the control program is normally in read-only memory and cannot be changed, the relative address used in execution of branch instructions is a constant numerical value. Cycle-by-cycle operation is shown in Table 10 for relative addressing.

Indexed Addressing Mode — With Indexed addressing, the numerical address is variable and depends on the current contents of the Index Register. A source statement such as

Operator	Operand	Comment
STAA	X	PUT A IN INDEXED LOCATION

causes the MPU to store the contents of accumulator A in

3

TABLE 6 — INHERENT MODE CYCLE-BY-CYCLE OPERATION

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
ABA DAA SEC ASL DEC SEI ASR INC SEV CBA LSR TAB CLC NEG TAP CLI NOP TBA CLR ROL TPA CLV ROR TST COM SBA	2	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
DES DEX INS INX	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
		3	0	Previous Register Contents	1	Irrelevant Data (Note 1)
		4	0	New Register Contents	1	Irrelevant Data (Note 1)
PSH	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
		3	1	Stack Pointer	0	Accumulator Data
		4	0	Stack Pointer − 1	1	Accumulator Data
PUL	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
		3	0	Stack Pointer	1	Irrelevant Data (Note 1)
		4	1	Stack Pointer + 1	1	Operand Data from Stack
TSX	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
		3	0	Stack Pointer	1	Irrelevant Data (Note 1)
		4	0	New Index Register	1	Irrelevant Data (Note 1)
TXS	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
		3	0	Index Register	1	Irrelevant Data
		4	0	New Stack Pointer	1	Irrelevant Data
RTS	5	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Irrelevant Data (Note 2)
		3	0	Stack Pointer	1	Irrelevant Data (Note 1)
		4	1	Stack Pointer + 1	1	Address of Next Instruction (High Order Byte)
		5	1	Stack Pointer + 2	1	Address of Next Instruction (Low Order Byte)

TABLE 6 — INHERENT MODE CYCLE-BY-CYCLE OPERATION (CONTINUED)

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
WAI	9	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
		3	1	Stack Pointer	0	Return Address (Low Order Byte)
		4	1	Stack Pointer – 1	0	Return Address (High Order Byte)
		5	1	Stack Pointer – 2	0	Index Register (Low Order Byte)
		6	1	Stack Pointer – 3	0	Index Register (High Order Byte)
		7	1	Stack Pointer – 4	0	Contents of Accumulator A
		8	1	Stack Pointer – 5	0	Contents of Accumulator B
		9	1	Stack Pointer – 6 (Note 3)	1	Contents of Cond. Code Register
RTI	10	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Irrelevant Data (Note 2)
		3	0	Stack Pointer	1	Irrelevant Data (Note 1)
		4	1	Stack Pointer + 1	1	Contents of Cond. Code Register from Stack
		5	1	Stack Pointer + 2	1	Contents of Accumulator B from Stack
		6	1	Stack Pointer + 3	1	Contents of Accumulator A from Stack
		7	1	Stack Pointer + 4	1	Index Register from Stack (High Order Byte)
		8	1	Stack Pointer + 5	1	Index Register from Stack (Low Order Byte)
		9	1	Stack Pointer + 6	1	Next Instruction Address from Stack (High Order Byte)
		10	1	Stack Pointer + 7	1	Next Instruction Address from Stack (Low Order Byte)
SWI	12	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Irrelevant Data (Note 1)
		3	1	Stack Pointer	0	Return Address (Low Order Byte)
		4	1	Stack Pointer – 1	0	Return Address (High Order Byte)
		5	1	Stack Pointer – 2	0	Index Register (Low Order Byte)
		6	1	Stack Pointer – 3	0	Index Register (High Order Byte)
		7	1	Stack Pointer – 4	0	Contents of Accumulator A
		8	1	Stack Pointer – 5	0	Contents of Accumulator B
		9	1	Stack Pointer – 6	0	Contents of Cond. Code Register
		10	0	Stack Pointer – 7	1	Irrelevant Data (Note 1)
		11	1	Vector Address FFFA (Hex)	1	Address of Subroutine (High Order Byte)
		12	1	Vector Address FFFB (Hex)	1	Address of Subroutine (Low Order Byte)

Note 1. If device which is addressed during this cycle uses VMA, then the Data Bus will go to the high impedance three-state condition. Depending on bus capacitance, data from the previous cycle may be retained on the Data Bus.
Note 2. Data is ignored by the MPU.
Note 3. While the MPU is waiting for the interrupt, Bus Available will go high indicating the following states of the control lines: VMA is low; Address Bus, R/$\overline{W}$, and Data Bus are all in the high impedance state.

the memory location specified by the contents of the Index Register (recall that the label "X" is reserved to designate the Index Register). Since there are instructions for manipulating X during program execution (LDX, INX, DEC, etc.), the Indexed addressing mode provides a dynamic "on the fly" way to modify program activity.

The operand field can also contain a numerical value that will be automatically added to X during execution. This format is illustrated in Figure 33.

When the MPU encounters the LDAB (Indexed) opcode in location 5006, it looks in the next memory location for the value to be added to X (5 in the example) and calculates the required address by adding 5 to the present Index Register value of 400. In the operand format, the offset may be represented by a label or a numerical value in the range 0-255 as in the example. In the earlier example, STAA X, the operand is equivalent to 0, X, that is, the 0 may be omitted when the desired address is equal to X. Table 11 shows the cycle-by-cycle operation for the Indexed Mode of Addressing.

FIGURE 29 — IMMEDIATE ADDRESSING MODE

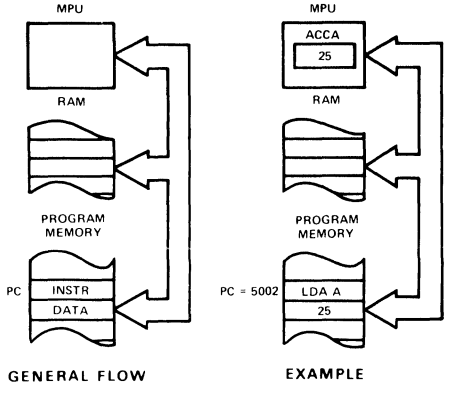


FIGURE 30 — DIRECT ADDRESSING MODE

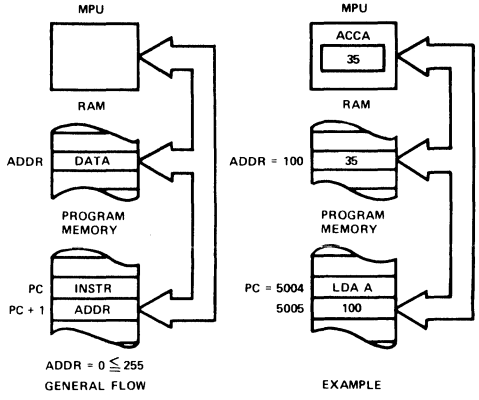


TABLE 7 — IMMEDIATE MODE CYCLE-BY-CYCLE OPERATION

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	2	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Operand Data
CPX LDS LDX	3	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Operand Data (High Order Byte)
		3	1	Op Code Address + 2	1	Operand Data (Low Order Byte)

TABLE 8 — DIRECT MODE CYCLE-BY-CYCLE OPERATION

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	3	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand
		3	1	Address of Operand	1	Operand Data
CPX LDS LDX	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand
		3	1	Address of Operand	1	Operand Data (High Order Byte)
		4	1	Operand Address + 1	1	Operand Data (Low Order Byte)
STA	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Destination Address
		3	0	Destination Address	1	Irrelevant Data (Note 1)
		4	1	Destination Address	0	Data from Accumulator
STS STX	5	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand
		3	0	Address of Operand	1	Irrelevant Data (Note 1)
		4	1	Address of Operand	0	Register Data (High Order Byte)
		5	1	Address of Operand + 1	0	Register Data (Low Order Byte)

Note 1. If device which is address during this cycle uses VMA, then the Data Bus will go to the high impedance three-state condition. Depending on bus capacitance, data from the previous cycle may be retained on the Data Bus.

FIGURE 31 — EXTENDED ADDRESSING MODE

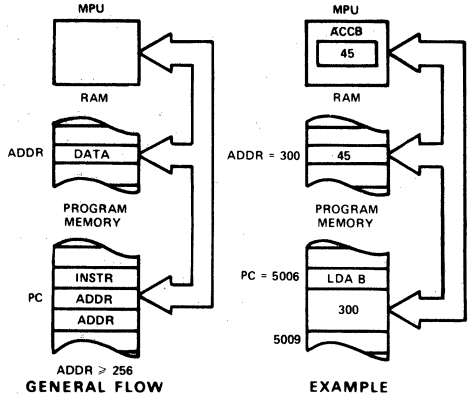


TABLE 9 — EXTENDED MODE CYCLE-BY-CYCLE

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
STS STX	6	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	0	Address of Operand	1	Irrelevant Data (Note 1)
		5	1	Address of Operand	0	Operand Data (High Order Byte)
		6	1	Address of Operand + 1	0	Operand Data (Low Order Byte)
JSR	9	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Subroutine (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Subroutine (Low Order Byte)
		4	1	Subroutine Starting Address	1	Op Code of Next Instruction
		5	1	Stack Pointer	0	Return Address (Low Order Byte)
		6	1	Stack Pointer − 1	0	Return Address (High Order Byte)
		7	0	Stack Pointer − 2	1	Irrelevant Data (Note 1)
		8	0	Op Code Address + 2	1	Irrelevant Data (Note 1)
		9	1	Op Code Address + 2	1	Address of Subroutine (Low Order Byte)
JMP	3	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Jump Address (High Order Byte)
		3	1	Op Code Address + 2	1	Jump Address (Low Order Byte)
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	1	Address of Operand	1	Operand Data
CPX LDS LDX	5	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	1	Address of Operand	1	Operand Data (High Order Byte)
		5	1	Address of Operand + 1	1	Operand Data (Low Order Byte)
STA A STA B	5	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Destination Address (High Order Byte)
		3	1	Op Code Address + 2	1	Destination Address (Low Order Byte)
		4	0	Operand Destination Address	1	Irrelevant Data (Note 1)
		5	1	Operand Destination Address	0	Data from Accumulator
ASL LSR ASR NEG CLR ROL COM ROR DEC TST INC	6	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	1	Address of Operand	1	Current Operand Data
		5	0	Address of Operand	1	Irrelevant Data (Note 1)
		6	1/0 (Note 2)	Address of Operand	0	New Operand Data (Note 2)

Note 1. If device which is addressed during this cycle uses VMA, then the Data Bus will go to the high impedance three-state condition. Depending on bus capacitance, data from the previous cycle may be retained on the Data Bus.

Note 2. For TST, VMA = 0 and Operand data does not change.

FIGURE 32 — RELATIVE ADDRESSING MODE

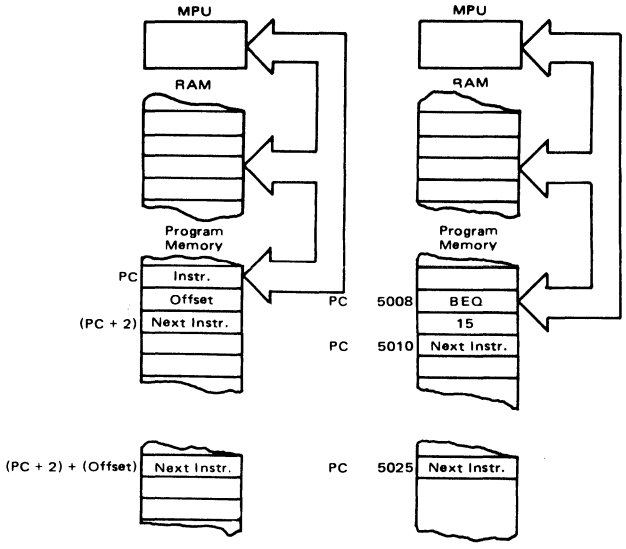


FIGURE 33 — INDEXED ADDRESSING MODE

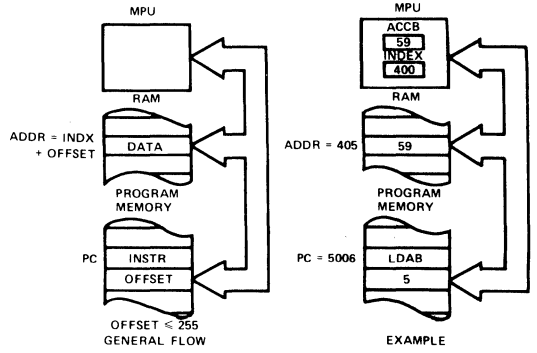


TABLE 10 — RELATIVE MODE CYCLE-BY-CYCLE OPERATION

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
BCC BHI BNE BCS BLE BPL BEQ BLS BRA BGE BLT BVC BGT BMI BVS	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Branch Offset
		3	0	Op Code Address + 2	1	Irrelevant Data (Note 1)
		4	0	Branch Address	1	Irrelevant Data (Note 1)
BSR	8	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Branch Offset
		3	0	Return Address of Main Program	1	Irrelevant Data (Note 1)
		4	1	Stack Pointer	0	Return Address (Low Order Byte)
		5	1	Stack Pointer — 1	0	Return Address (High Order Byte)
		6	0	Stack Pointer — 2	1	Irrelevant Data (Note 1)
		7	0	Return Address of Main Program	1	Irrelevant Data (Note 1)
		8	0	Subroutine Address	1	Irrelevant Data (Note 1)

Note 1. If device which is addressed during this cycle uses VMA, then the Data Bus will go to the high impedance three-state condition. Depending on bus capacitance, data from the previous cycle may be retained on the Data Bus.

TABLE 11 — INDEXED MODE CYCLE-BY-CYCLE

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
INDEXED						
JMP	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	5	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
		5	1	Index Register Plus Offset	1	Operand Data
CPX LDS LDX	6	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
		5	1	Index Register Plus Offset	1	Operand Data (High Order Byte)
		6	1	Index Register Plus Offset + 1	1	Operand Data (Low Order Byte)
STA	6	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
		5	0	Index Register Plus Offset	1	Irrelevant Data (Note 1)
		6	1	Index Register Plus Offset	0	Operand Data
ASL LSR ASR NEG CLR ROL COM ROR DEC TST INC	7	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
		5	1	Index Register Plus Offset	1	Current Operand Data
		6	0	Index Register Plus Offset	1	Irrelevant Data (Note 1)
		7	1/0 (Note 2)	Index Register Plus Offset	0	New Operand Data (Note 2)
STS STX	7	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
		5	0	Index Register Plus Offset	1	Irrelevant Data (Note 1)
		6	1	Index Register Plus Offset	0	Operand Data (High Order Byte)
		7	1	Index Register Plus Offset + 1	0	Operand Data (Low Order Byte)
JSR	8	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	1	Stack Pointer	0	Return Address (Low Order Byte)
		5	1	Stack Pointer − 1	0	Return Address (High Order Byte)
		6	0	Stack Pointer − 2	1	Irrelevant Data (Note 1)
		7	0	Index Register	1	Irrelevant Data (Note 1)
		8	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)

Note 1. If device which is addressed during this cycle uses VMA, then the Data Bus will go to the high impedance three-state condition. Depending on bus capacitance, data from the previous cycle may be retained on the Data Bus.

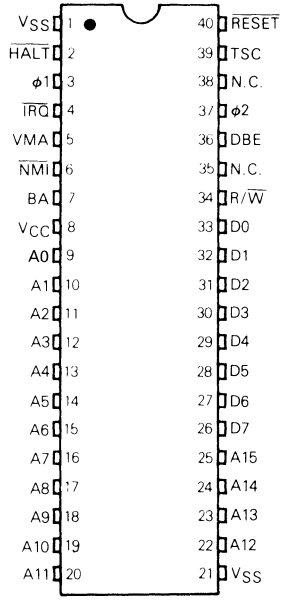
Note 2. For TST, VMA = 0 and Operand data does not change.

ORDERING INFORMATION

Package Type	Frequency (MHz)	Temperature	Order Number
Cerdip	1.0	0°C to 70°C	MC6800S
S Suffix	1.0	−40°C to 85°C	MC6800CS
	1.5	0°C to 70°C	MC68A00S
	1.5	−40°C to 85°C	MC68A00CS
	2.0	0°C to 70°C	MC68B00S
Plastic	1.0	0°C to 70°C	MC6800P
P Suffix	1.0	−40°C to 85°C	MC6800CP
	1.5	0°C to 70°C	MC68A00P
	1.5	−40°C to 85°C	MC68A00CP
	2.0	0°C to 70°C	MC68B00P

3

PIN ASSIGNMENT



MC6801
MC6803

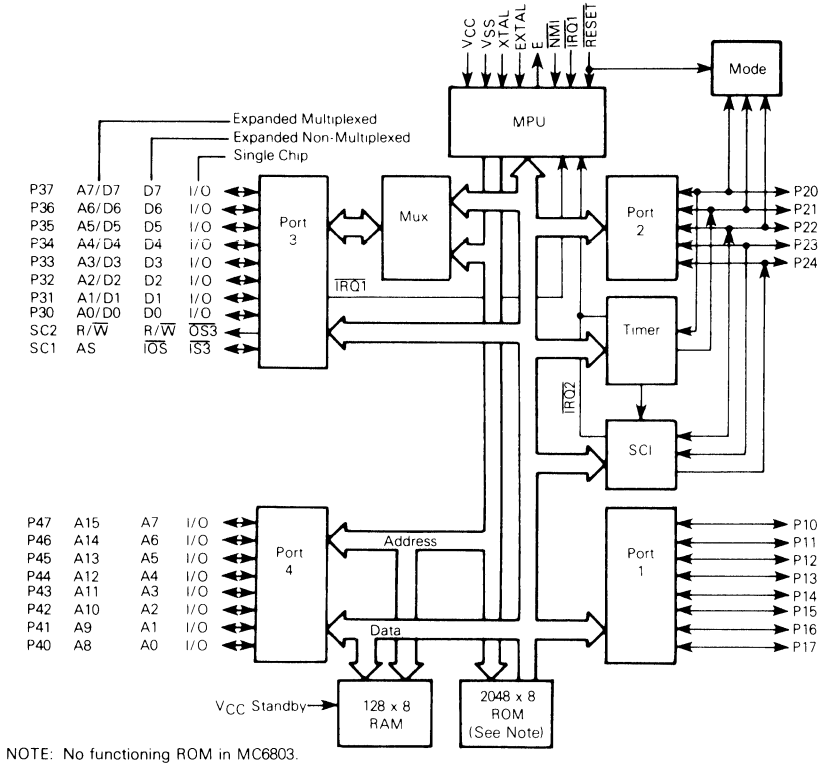
Microcontroller/Microprocessor (MCU/MPU)

The MC6801 is an 8-bit single-chip microcontroller unit (MCU) which significantly enhances the capabilities of the M6800 Family of parts. It includes an upgraded M6800 microprocessor unit (MPU) with upward-source and object-code compatibility. Execution times of key instructions have been improved and several new instructions have been added including an unsigned multiply. The MCU can function as a monolithic microcontroller or can be expanded to a 64K byte address space. It is TTL compatible and requires one +5-volt power supply. On-chip resources include 2048 bytes of ROM, 128 bytes of RAM, a serial communications interface (SCI), parallel I/O, and a three-function programmable timer. The MC6803 can be considered as an MC6801 operating in modes 2 or 3. An EPROM version of the MC6801, the MC68701 microcontroller, is available for systems development. The MC68701 is pin and code compatible with the MC6801/MC6803 and can be used to emulate the MC6801/MC6803. The MC68701 is described in a separate Advanced Information publication.

- Enhanced MC6800 Instruction Set
- 8×8 Multiply Instruction
- Serial Communications Interface (SCI)
- Upward Source and Object Code Compatibility with the M6800
- 16-Bit Three-Function Programmable Timer
- Single-Chip or Expanded Operation to 64K Byte Address Space
- Bus Compatibility with the M6800 Family
- 2048 Bytes of ROM (MC6801 Only)
- 128 Bytes of RAM
- 64 Bytes of RAM Retainable During Powerdown
- 29 Parallel I/O and Two Handshake Control Lines
- Internal Clock Generator with Divide-by-Four Output
- −40 to 85°C Temperature Range

This document contains information on a new product. Specifications and information herein are subject to change without notice.

FIGURE 1 — M6801 MICROCOMPUTER FAMILY BLOCK DIAGRAM



NOTE: No functioning ROM in MC6803.

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range MC6801, MC6803 MC6801C, MC6803C	T_A	T_L to T_H 0 to 70 −40 to +85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high-impedance circuit. For proper operation it is recommended the V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{CC}$. Input protection is enhanced by connecting unused inputs to either V_{DD} or V_{SS}.

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic Cerdip	θ_{JA}	 50 50	°C/W

3

CONTROL TIMING (V_{CC} = 5.0 V ±5%, V_{SS} = 0)

Characteristic	Symbol	MC6801		MC6801-1		MC68B01		Unit
		Min	Max	Min	Max	Min	Max	
Frequency of Operation	f_o	0.5	1.0	0.5	1.25	0.5	2.0	MHz
Crystal Frequency	f_{XTAL}	2.0	4.0	2.0	5.0	2.0	8.0	MHz
External Oscillator Frequency	$4f_o$	2.0	4.0	2.0	5.0	2.0	8.0	MHz
Crystal Oscillator Start Up Time	t_{rc}	—	100	—	100	—	100	ms
Processor Control Setup Time	t_{PCS}	200	—	170	—	110	—	ns

DC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 Vdc ±5%, V_{SS} = 0, $T_A = T_L$ to T_H, unless otherwise noted)

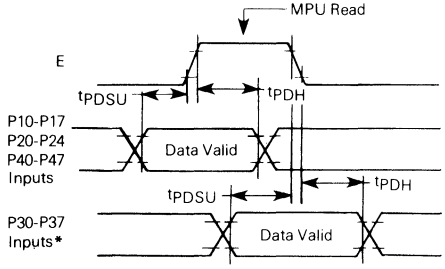
Characteristic		Symbol	MC6801 MC6803		MC6801C MC6803C		Unit
			Min	Max	Min	Max	
Input High Voltage	$\overline{RESET}$ Other Inputs	V_{IH}	$V_{SS}+4.0$ $V_{SS}+2.0$	V_{CC} V_{CC}	$V_{SS}+4.0$ $V_{SS}+2.2$	V_{CC} V_{CC}	V
Input Low Voltage	All Inputs	V_{IL}	$V_{SS}-0.3$	$V_{SS}+0.8$	$V_{SS}-0.3$	$V_{SS}+0.8$	V
Input Load Current (V_{in} = 0 to 2.4 V)	Port 4 SC1	I_{in}	— —	0.5 0.8	— —	0.8 1.0	mA
Input Leakage Current (V_{in} = 0 to 5.25 V)	$\overline{NMI}$, $\overline{IRQ1}$, $\overline{RESET}$	I_{in}	—	2.5	—	5.0	μA
Hi-Z (Off State) Input Current (V_{in} = 0.5 to 2.4 V)	Ports 1, 2, and 3	I_{TSI}	—	10	—	20	μA
Output High Voltage (I_{Load} = −65 μA, V_{CC} = Min)* (I_{Load} = −100 μA, V_{CC} = Min)	Port 4, SC1, SC2 Other Outputs	V_{OH}	$V_{SS}+2.4$ $V_{SS}+2.4$	— —	$V_{SS}+2.4$ $V_{SS}+2.4$	— —	V
Output Low Voltage (I_{Load} = 2.0 mA, V_{CC} = Min)	All outputs	V_{OL}	—	$V_{SS}+0.5$	—	$V_{SS}+0.6$	V
Darlington Drive Current (V_O = 1.5 V)	Port 1	I_{OH}	1.0	4.0	1.0	5.0	mA
Internal Power Dissipation (Measured at $T_A = T_L$ in Steady-State Operation)		P_{INT}	—	1200	—	1500	mW
Input Capacitance (V_{in} = 0, T_A = 25°C, f_o = 1.0 MHz)	Port 3, Port 4, SC1 Other Inputs	C_{in}	— —	12.5 10	— —	12.5 10	pF
V_{CC} Standby	Powerdown Powerup	V_{SBB} V_{SB}	4.0 4.75	5.25 5.25	4.0 4.75	5.25 5.25	V
Standby Current	Powerdown	I_{SBB}	—	6.0	—	8.0	mA

*Negotiable to −100 μA (for further information contact the factory)

PERIPHERAL PORT TIMING (Refer to Figures 2–5)

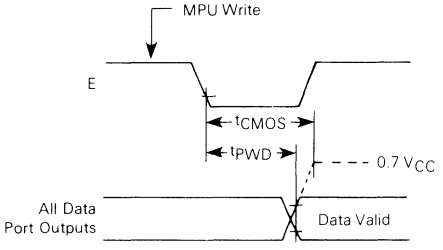
Characteristic	Symbol	MC6801 MC6803		MC6801-1 MC6803-1		MC68B01 MC68B03		Unit
		Min	Max	Min	Max	Min	Max	
Peripheral Data Setup Time	t_{PDSU}	200	—	200	—	100	—	ns
Peripheral Data Hold Time	t_{PDH}	200	—	200	—	100	—	ns
Delay Time, Enable Positive Transition to $\overline{OS3}$ Negative Transition	t_{OSD1}	—	350	—	350	—	250	ns
Delay Time, Enable Positive Transition to $\overline{OS3}$ Positive Transition	t_{OSD2}	—	350	—	350	—	250	ns
Delay Time, Enable Negative Transition to Peripheral Data Valid	t_{PWD}	—	350	—	350	—	250	ns
Delay Time, Enable Negative Transition to Peripheral CMOS Data Valid	t_{CMOS}	—	2.0	—	2.0	—	2.0	μs
Input Strobe Pulse Width	t_{PWIS}	200	—	200	—	100	—	ns
Input Data Hold Time	t_{IH}	50	—	50	—	30	—	ns
Input Data Setup Time	t_{IS}	20	—	20	—	20	—	ns

FIGURE 2 — DATA SETUP AND HOLD TIMES (MPU READ)



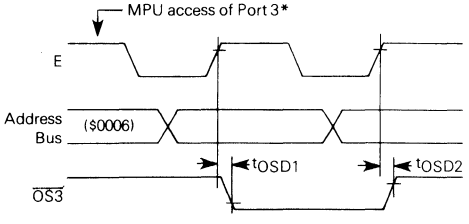
* Port 3 non-latched operation (LATCH ENABLE = 0)

FIGURE 3 — DATA SETUP AND HOLD TIMES (MPU WRITE)



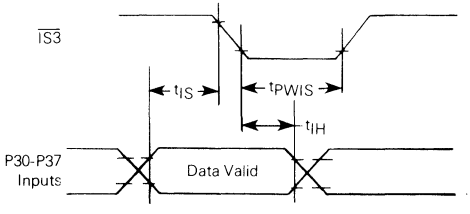
NOTES:
1. 10 k pullup resistor required for port 2 to reach 0.7 V_{CC}.
2. Not applicable to P21.
3. Port 4 cannot be pulled above V_{CC}.

FIGURE 4 — PORT 3 OUTPUT STROBE TIMING (MC6801 SINGLE-CHIP MODE)



* Access matches output strobe select (OSS = 0, a read; OSS = 1, a write)

FIGURE 5 — PORT 3 LATCH TIMING (MC6801 SINGLE-CHIP MODE)



NOTE: Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted.

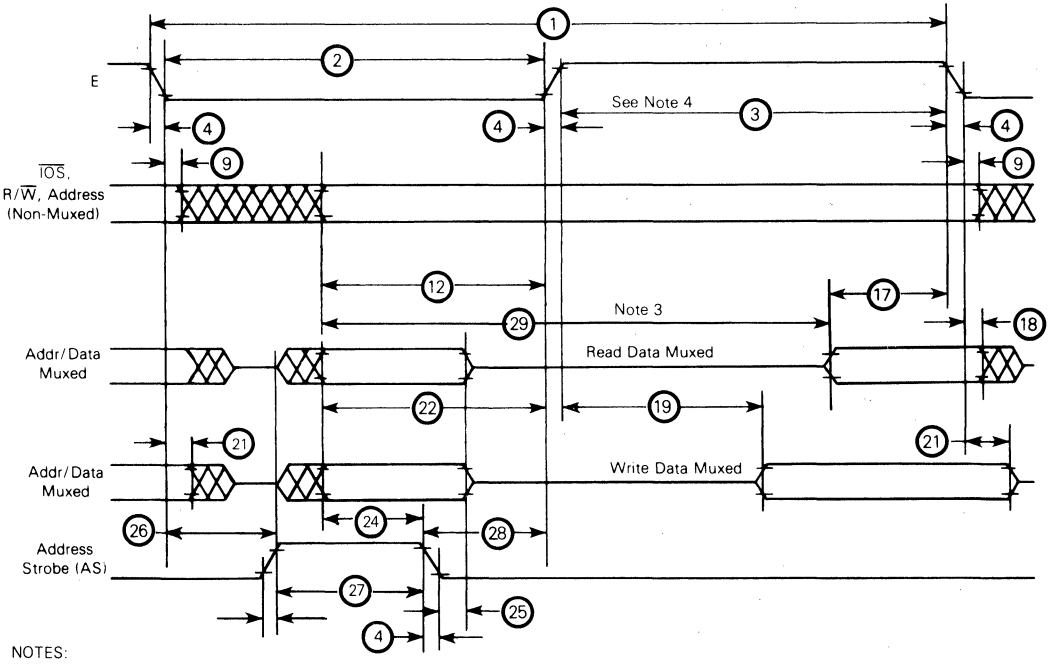
BUS TIMING (See Notes 1 and 2)

Ident. Number	Characteristics	Symbol	MC6801 MC6803		MC6801-1 MC6803-1		MC68B01 MC68B03		Unit
			Min	Max	Min	Max	Min	Max	
1	Cycle Time	t_{cyc}	1.0	2.0	0.8	2.0	0.5	2.0	μs
2	Pulse Width, E Low	PW_{EL}	430	1000	360	1000	210	1000	ns
3	Pulse Width, E High	PW_{EH}	450	1000	360	1000	220	1000	ns
4	Clock Rise and Fall Time	t_r, t_f	—	25	—	25	—	20	ns
9	Address Hold Time	t_{AH}	20	—	20	—	10	—	ns
12	Non-Muxed Address Valid Time to E*	t_{AV}	200	—	150	—	70	—	ns
17	Read Data Setup Time	t_{DSR}	80	—	70	—	40	—	ns
18	Read Data Hold Time	t_{DHR}	10	—	10	—	10	—	ns
19	Write Data Delay Time	t_{DDW}	—	225	—	200	—	120	ns
21	Write Data Hold Time	t_{DHW}	20	—	20	—	10	—	ns
22	Muxed Address Valid Time to E Rise*	t_{AVM}	200	—	150	—	80	—	ns
24	Muxed Address Valid Time to AS Fall*	t_{ASL}	60	—	50	—	20	—	ns
25	Muxed Address Hold Time	t_{AHL}	20	—	20	—	10	—	ns
26	Delay time, E to AS Rise*	t_{ASD}	90**	—	70**	—	45**	—	ns
27	Pulse Width, AS High*	PW_{ASH}	220	—	170	—	110	—	ns
28	Delay Time, AS to E Rise*	t_{ASED}	90	—	70	—	45	—	ns
29	Usable Access Time*	t_{ACC}	595	—	465	—	270	—	ns

*At specified cycle time.

**t_{ASD} parameters listed assume external TTL clock drive with 50% ±5% duty cycle. Devices driven by an external TTL clock with 50% ±1% duty cycle or which use a crystal have the following t_{ASD} specifications: 100 nanoseconds minimum (1.0 MHz devices), 80 nanoseconds minimum (1.25 MHz device), 50 nanoseconds minimum (2.0 MHz devices).

FIGURE 6 — BUS TIMING



NOTES:
1. Voltage levels shown are $V_L \leq 0.5$ V, $V_H \geq 2.4$ V, unless otherwise specified.
2. Measurement points shown are 0.8 V and 2.0 V, unless otherwise specified.
3. Usable access time is computed by: 12 + 3 − 17 + 4.
4. Memory devices should be enabled only during E high to avoid port 3 bus contention.

FIGURE 7 — CMOS LOAD

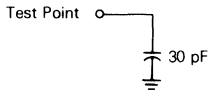
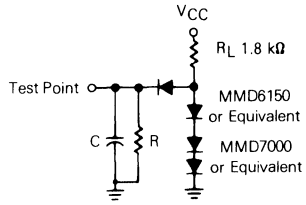


FIGURE 8 — TIMING TEST LOAD PORTS 1, 2, 3, 4



C = 90 pF for P30-P37, P40-P47, E, SC1, SC2
= 30 pF for P10-P17, P20-P24
R = 37 kΩ for P40-P47, SC1, SC2
= 24 kΩ for P10-P17, P20-P24
= 24 kΩ for P30-P37, E

3

INTRODUCTION

The MC6801 is an 8-bit monolithic microcomputer which can be configured to function in a wide variety of applications. The facility which provides this extraordinary flexibility is its ability to be hardware programmed into eight different operating modes. The operating mode controls the configuration of 18 of the 40 MCU pins, available on-chip resources, memory map, location (internal or external) of interrupt vectors, and type of external bus. The configuration of the remaining 22 pins is not dependent on the operating mode.

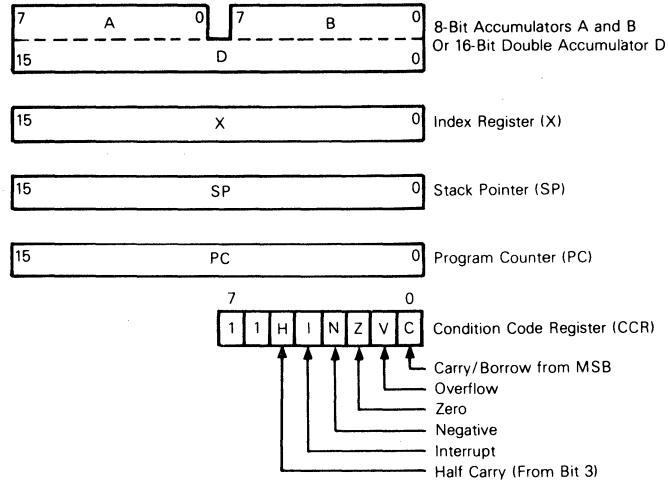
Twenty-nine pins are organized as three 8-bit ports and one 5-bit port. Each port consists of at least a data register and a write-only data direction register. The data direction register is used to define whether corresponding bits in the data register are configured as an input (clear) or output (set).

The term "port," by itself, refers to all of the hardware associated with the port. When the port is used as a "data port" or "I/O port," it is controlled by the port data direction register and the programmer has direct access to the port pins using the port data register. Port pins are labeled as Pij where i identifies one of four ports and j indicates the particular bit.

The microprocessor unit (MPU) is an enhanced MC6800 MPU with additional capabilities and greater throughput. It is upward source and object code compatible with the MC6800. The programming model is depicted in Figure 9, where accumulator D is a concatenation of accumulators A and B. A list of new operations added to the M6800 instruction set are shown in Table 1.

The MC6803 can be considered an MC6801 that operates in Modes 2 and 3 only.

FIGURE 9 — PROGRAMMING MODEL



3

OPERATING MODES

The MC6801 provides eight different operating modes (0 through 7) and the MC6803 provides two operating modes (2 and 3). The operating modes are hardware selectable and determine the device memory map, the configuration of port 3, port 4, SC1, SC2, and the physical location of the interrupt vectors.

FUNDAMENTAL MODES

The eight operating modes can be grouped into three fundamental modes which refer to the type of bus it supports: single chip, expanded non-multiplexed, and expanded multiplexed. Single-chip modes include 4 and 7, expanded non-multiplexed mode is 5, and the remaining five modes are expanded multiplexed modes. Table 2 summarizes the characteristics of the operating modes.

MC6801 Single-Chip Modes (4, 7)

In the single-chip mode, the four MCU ports are configured as parallel input/output data ports, as shown in Figure 10. The MCU functions as a monolithic microcomputer in these two modes without external address or data buses. A maximum of 29 I/O lines and two port 3 control lines are provided. Peripherals or another MCU can be interfaced to port 3 in a loosely coupled dual processor configuration, as shown in Figure 11.

TABLE 1 — NEW INSTRUCTIONS

Instruction	Description
ABX	Unsigned addition of accumulator B to index register
ADDD	Adds (without carry) the double accumulator to memory and leaves the sum in the double accumulator
ASLD or LSLD	Shifts the double accumulator left (towards MSB) one bit; the LSB is cleared and the MSB is shifted into the C bit
BHS	Branch if higher or same; unsigned conditional branch (same as BCC)
BLO	Branch if lower; unsigned conditional branch (same as BCS)
BRN	Branch never
JSR	Additional addressing mode: direct
LDD	Loads double accumulator from memory
LSL	Shifts memory or accumulator left (towards MSB) one bit; the LSB is cleared and the MSB is shifted into the C bit (same as ASL)
LSRD	Shifts the double accumulator right (towards LSB) one bit; the MSB is cleared and the LSB is shifted into the C bit
MUL	Unsigned multiply; multiplies the two accumulators and leaves the product in the double accumulator
PSHX	Pushes the index register to stack
PULX	Pulls the index register from stack
STD	Stores the double accumulator to memory
SUBD	Subtracts memory from the double accumulator and leaves the difference in the double accumulator
CPX	Internal processing modified to permit its use with any conditional branch instruction

In single-chip test mode (4), the RAM responds to $XX80 through $XXFF and the ROM is removed from the internal address map. A test program must first be loaded into the RAM using modes 0, 1, 2, or 6. If the MCU is reset and then programmed into mode 4, execution will begin at $XXFE:XXFF. Mode 5 can be irreversibly entered from mode 4 without asserting $\overline{\text{RESET}}$ by setting bit 5 of the port 2 data register. This mode is used primarily to test ports 3 and 4 in the single-chip and non-multiplexed modes.

MC6801 Expanded Non-Multiplexed Mode (5)

A modest amount of external memory space is provided in the expanded non-multiplexed mode while significant on-chip resources are retained. Port 3 functions as an 8-bit bidirectional data bus and port 4 is configured initially as an input data port. Any combination of the eight least-significant address lines may be obtained by writing to the port 4 data direction register. Stated alternatively, any combination of A0 to A7 may be provided while retaining the remainder as input data lines. Internal pullup resistors pull the port 4 lines high until the port is configured.

Figure 12 illustrates a typical system configuration in the expanded non-multiplexed mode. The MCU interfaces directly with M6800 Family parts and can access 256 bytes of external address space at $100 through $1FF. IOS provides an address decode of external memory ($100-$1FF) and can be used as a memory-page select or chip-select line

TABLE 2 — SUMMARY OF MC6801/03 OPERATING MODES

Common to all Modes: Reserved Register Area Port 1 Port 2 Programmable Timer Serial Communications Interface
Single Chip Mode 7 128 bytes of RAM; 2048 bytes of ROM Port 3 is a parallel I/O port with two control lines Port 4 is a parallel I/O port SC1 is Input Strobe 3 ($\overline{\text{IS3}}$) SC2 is Output Strobe 3 ($\overline{\text{OS3}}$)
Expanded Non-Multiplexed Mode 5 128 bytes of RAM; 2048 bytes of ROM 256 bytes of external memory space Port 3 is an 8-bit data bus Port 4 is an input port/address bus SC1 is Input/Output Select ($\overline{\text{IOS}}$) SC2 is Read/Write (R/$\overline{\text{W}}$)
Expanded Multiplexed Modes 1, 2, 3, 6* Four memory space options (64K address space): (1) No internal RAM or ROM (Mode 3) (2) Internal RAM, no ROM (Mode 2) (3) Internal RAM and ROM (Mode 1) (4) Internal RAM, ROM with partial address bus (Mode 6) Port 3 is a multiplexed address/data bus Port 4 is an address bus (inputs/address in Mode 6) SC1 is Address Strobe (AS) SC2 is Read/Write (R/$\overline{\text{W}}$)
Test Modes 0 and 4 Expanded Multiplexed Test Mode 0 May be used to test RAM and ROM Single Chip and Non-Multiplexed Test Mode 4 (1) May be changed to Mode 5 without going through Reset (2) May be used to test Ports 3 and 4 as I/O ports

* The MC6803 operates only in modes 2 and 3.

FIGURE 10 — SINGLE-CHIP MODE

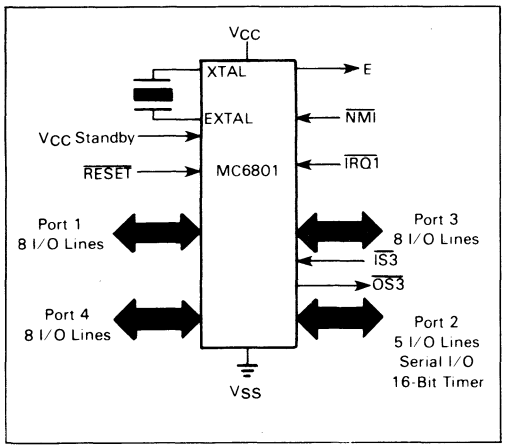


FIGURE 11 — SINGLE-CHIP DUAL PROCESSOR CONFIGURATION

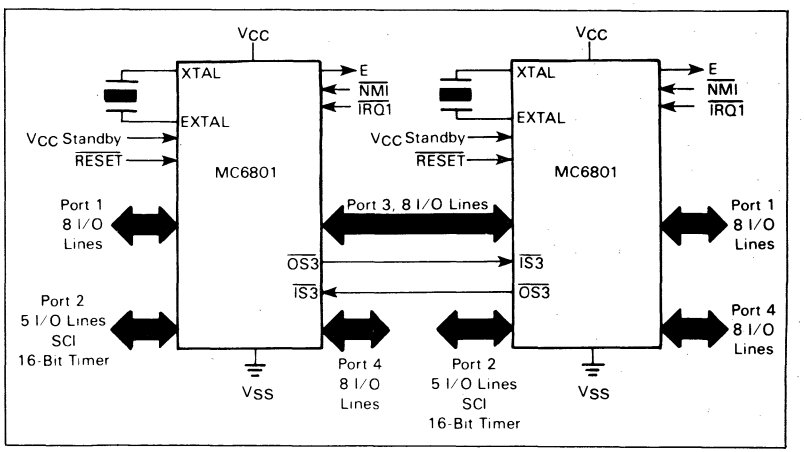
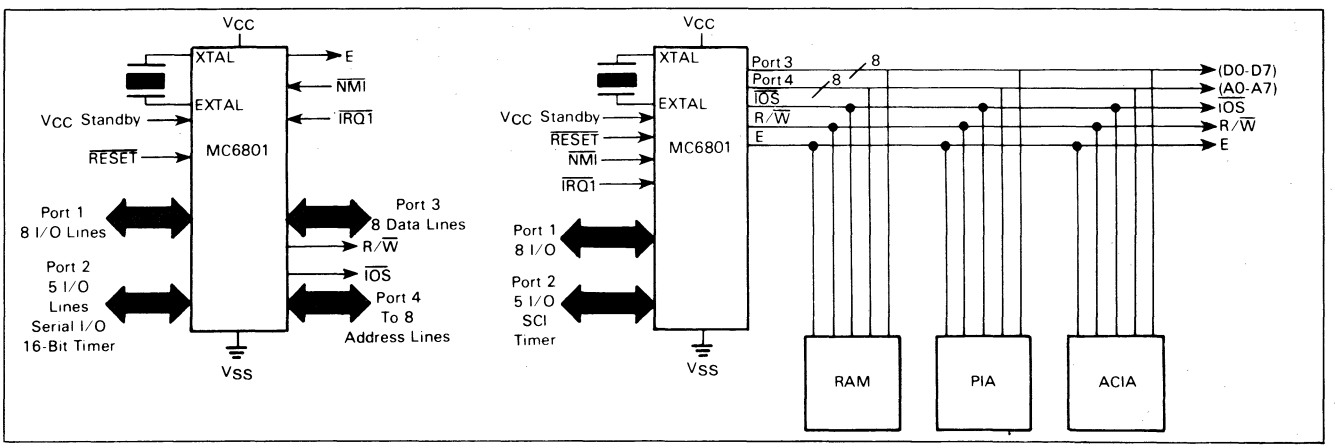


FIGURE 12 — EXPANDED NON-MULTIPLEXED CONFIGURATION



Expanded-Multiplex Modes (0, 1, 2, 3, 6)

A 64K byte memory space is provided in the expanded-multiplex modes. In each of the expanded-multiplexed modes port 3 functions as a time multiplexed address/data bus with address valid on the negative edge of address strobe (AS), and data valid while E is high. In modes 0 to 3, port 4 provides address lines A8 to A15. In mode 6, however, port 4 initially is configured at $\overline{\text{RESET}}$ as an input data port. The port 4 data direction register can then be changed to provide any combination of address lines, A8 to A15. Stated alternatively, any subset of A8 to A15 can be provided while retaining the remaining port 4 lines as input data lines. Internal pullup resistors pull the port 4 lines high until software configures the port.

In mode 0, the reset vector is external for the first two E cycles after the positive edge of $\overline{\text{RESET}}$, and internal thereafter. In addition, the internal and external data buses are connected so there must be no memory map overlap in order to avoid potential bus conflicts. Mode 0 is used primarily to verify the ROM pattern and monitor the internal data bus with the automated test equipment.

Only the MC6801 can operate in each of the expanded-multiplexed modes. The MC6803 operates only in modes 2 and 3.

Figure 13 depicts a typical configuration for the expanded-multiplexed modes. Address strobe can be used to control a transparent D-type latch to capture addresses A0-A7, as shown in Figure 14. This allows port 3 to function as a data bus when E is high.

PROGRAMMING THE MODE

The operating mode is determined at $\overline{\text{RESET}}$ by the levels asserted on P22, P21, and P20. These levels are latched into PC2, PC1, and PC0 of the program control register on the positive edge of $\overline{\text{RESET}}$. The operating mode may be read from the port 2 data register as shown below, and programming levels and timing must be met as shown in Figure 15. A brief outline of the operating modes is shown in Table 3. Note that if diodes are used to program the mode, the diode forward voltage drop must not exceed the V_{MPDD} minimum.

PORT 2 DATA REGISTER

7	6	5	4	3	2	1	0	
PC2	PC1	PC0	P24	P23	P22	P21	P20	$0003

Circuitry to provide the programming levels is dependent primarily on the normal system usage of the three pins. If configured as outputs, the circuit shown in Figure 16 may be used; otherwise, three-state buffers can be used to provide isolation while programming the mode.

3

TABLE 3 — MODE SELECTION SUMMARY

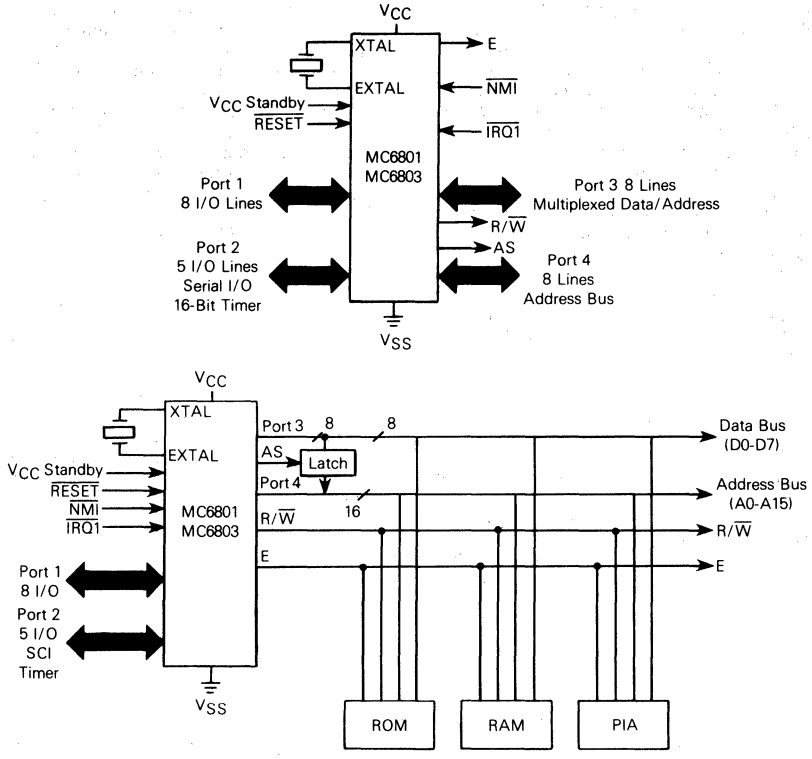
Mode*	P22 PC2	P21 PC1	P20 PC0	ROM	RAM	Interrupt Vectors	Bus Mode	Operating Mode
7	H	H	H	I	I	I	I	Single Chip
6	H	H	L	I	I	I	MUX(5, 6)	Multiplexed/Partial Decode
5	H	L	H	I	I	I	NMUX(5,6)	Non-Multiplexed/Partial Decode
4	H	L	L	I(2)	I(1)	I	I	Single-Chip Test
3	L	H	H	E	E	E	MUX(4)	Multiplexed/No RAM or ROM
2	L	H	L	E	I	E	MUX(4)	Multiplexed/RAM
1	L	L	H	I	I	E	MUX(4)	Multiplexed/RAM and ROM
0	L	L	L	I	I	I(3)	MUX(4)	Multiplexed Test

Legend:
- I — Internal
- E — External
- MUX — Multiplexed
- NMUX — Non-Multiplexed
- L — Logic Zero
- H — Logic One

NOTES:
(1) Internal RAM is addressed at $XX80.
(2) Internal ROM is disabled.
(3) $\overline{\text{RESET}}$ vector is external for two cycles after $\overline{\text{RESET}}$ goes high.
(4) Addresses associated with ports 3 and 4 are considered external in modes 0, 1, 2, and 3.
(5) Addresses associated with port 3 are considered external in modes 5 and 6.
(6) Port 4 default is user data input; address output is optional by writing to port 4 data direction register.

* The MC6803 operates only in modes 2 and 3.

FIGURE 13 — EXPANDED MULTIPLEXED CONFIGURATION



NOTE: To avoid data bus (port 3) contention in the expanded multiplexed modes, memory devices should be enabled only during E high time.

FIGURE 14 — TYPICAL LATCH ARRANGEMENT

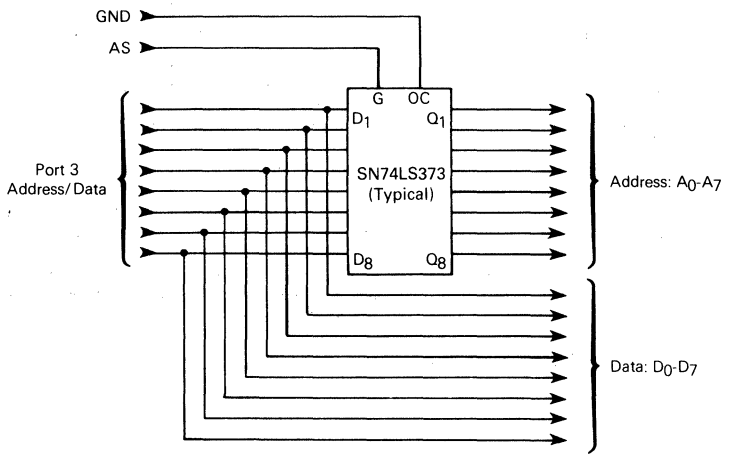
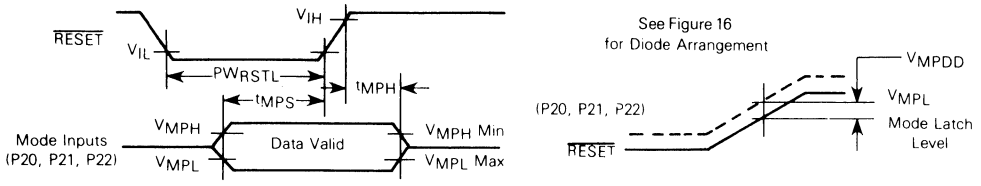


FIGURE 15 — MODE PROGRAMMING TIMING

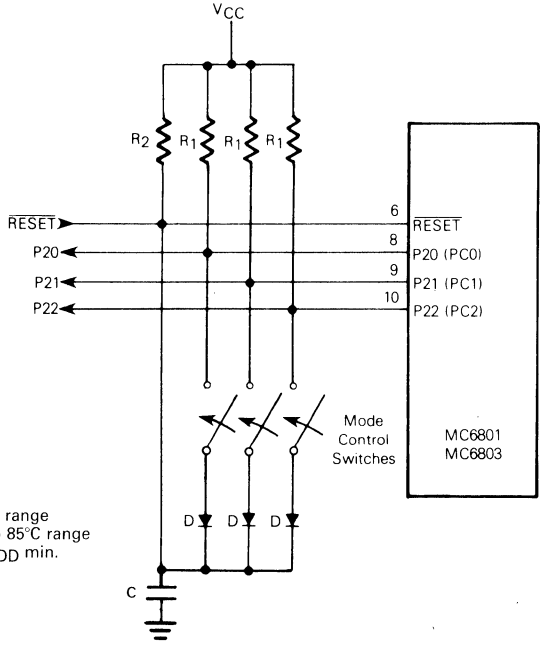


MODE PROGRAMMING (Refer to Figure 15)

Characteristic	Symbol	Min	Max	Unit
Mode Programming Input Voltage Low* (for T_A = 0 to 70°C)	V_{MPL}	—	1.7	V
Mode Programming Input Voltage High	V_{MPH}	4.0	—	V
Mode Programming Diode Differential (If Diodes are Used) (for T_A = 0 to 70°C)	V_{MPDD}	0.4	—	V
RESET Low Pulse Width	PW_{RSTL}	3.0	—	E Cycles
Mode Programming Setup Time	t_{MPS}	2.0	—	E Cycles
Mode Programming Hold Time RESET Rise Time≥1 μs RESET Rise Time<1 μs	t_{MPH}	0 100	— —	ns

Note: For T_A = −40 to 85°C, Maximum V_{MPL} = 1.7, and Minimum V_{MPDD} = 0.4.

FIGURE 16 — TYPICAL MODE PROGRAMMING CIRCUIT

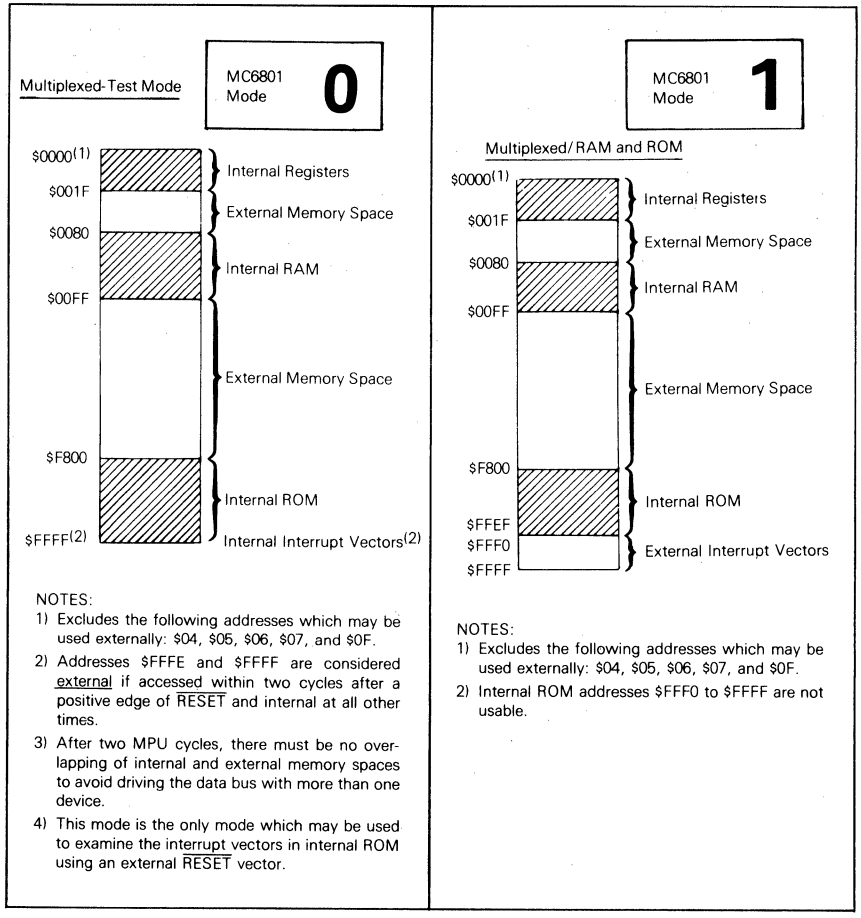


NOTES:
1. Mode 7 as shown
2. $R_2 \cdot C$ = Reset time constant
3. R_1 = 10 k (typical)
4. D = IN914, IN4001 in the 0 to 70°C range
   D = 1N270, MBD201 in the −40 to 85°C range
5. Diode V_f should not exceed V_{MPDD} min.

MEMORY MAPS

The M6801 Family can provide up to 64K byte address space depending on the operating mode. A memory map for each operating mode is shown in Figure 17. The first 32 locations of each map are reserved for the internal register area, as shown in Table 4, with exceptions as indicated.

FIGURE 17 — MC6801/03 MEMORY MAPS (Sheet 1 of 3)



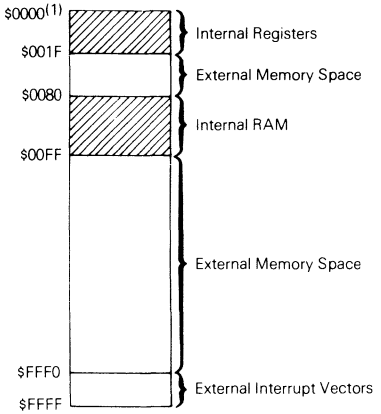
NOTES:
1) Excludes the following addresses which may be used externally: $04, $05, $06, $07, and $0F.
2) Addresses $FFFE and $FFFF are considered external if accessed within two cycles after a positive edge of $\overline{\text{RESET}}$ and internal at all other times.
3) After two MPU cycles, there must be no overlapping of internal and external memory spaces to avoid driving the data bus with more than one device.
4) This mode is the only mode which may be used to examine the interrupt vectors in internal ROM using an external $\overline{\text{RESET}}$ vector.

NOTES:
1) Excludes the following addresses which may be used externally: $04, $05, $06, $07, and $0F.
2) Internal ROM addresses $FFF0 to $FFFF are not usable.

FIGURE 17 — MC6801/03 MEMORY MAPS (Sheet 2 of 3)

MC6801
MC6803
Mode **2**

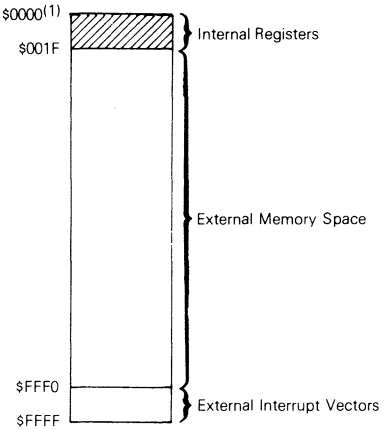
Multiplexed/RAM



NOTES:
1) Excludes the following addresses which may be used externally: $04, $05, $06, $07, and $0F.

MC6801
MC6803
Mode **3**

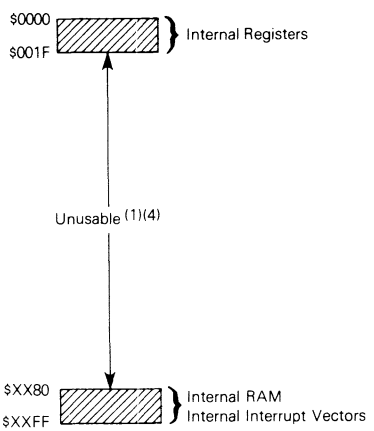
Multiplexed/No RAM or ROM



NOTES:
1) Excludes the following addresses which may be used externally: $04, $05, $06, $07, and $0F.

MC6801
Mode **4**

Single-Chip Test

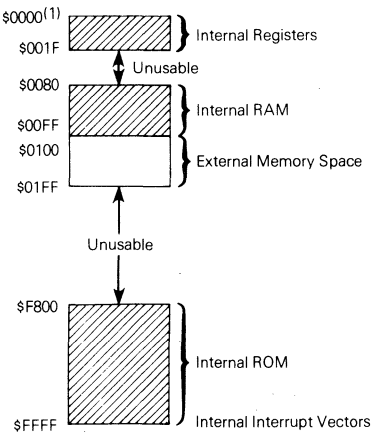


NOTES:
1) The internal ROM is disabled.
2) Mode 4 may be changed to Mode 5 without having to assert $\overline{\text{RESET}}$ by writing a one into the PCO bit of the port 2 data register.
3) Addresses A8 to A15 are treated as "don't cares" to decode internal RAM.
4) Internal RAM will appear at $XX80 to $XXFF.

FIGURE 17 — MC6801/03 MEMORY MAPS (Sheet 3 of 3)

MC6801 Mode 5

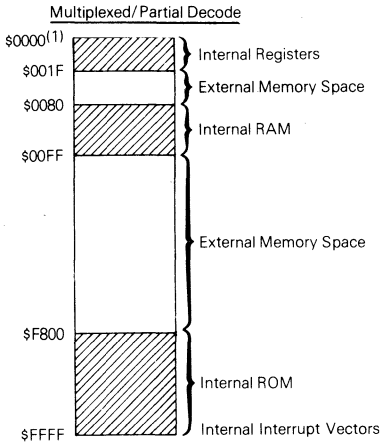
Non-Multiplexed/Partial Decode



NOTES:
1) Excludes the following addresses which may **not** be used externally: $04, $06, and $0F (no $\overline{IQS}$).
2) This mode may be entered without going through $\overline{RESET}$ by using mode 4 and subsequently writing a one into the PCO bit of the port 2 data register.
3) Address lines A0 to A7 will not contain addresses until the data direction register for port 4 has been written with ones in the appropriate bits. These address lines will assert ones until made outputs by writing the data direction register.

MC6801 Mode 6

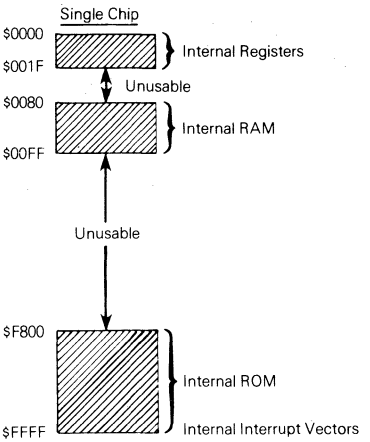
Multiplexed/Partial Decode



NOTES:
1) Excludes the following addresses which may be used externally: $04, $06, and $0F.
2) Address lines A8-A15 will not contain addresses until the data direction register for port 4 has been written with ones in the appropriate bits. These address lines will assert ones until made outputs by writing the data direction register.

MC6801 Mode 7

Single Chip



MC6801/03 INTERRUPTS

The M6801 Family supports two types of interrupt requests: maskable and non-maskable. A non-maskable interrupt ($\overline{NMI}$) is always recognized and acted upon at the completion of the current instruction. Maskable interrupts are controlled by the condition code register I bit and by individual enable bits. The I bit controls all maskable interrupts. Of the maskable interrupts, there are two types: $\overline{IRQ1}$ and $\overline{IRQ2}$. The programmable timer and serial communications interface use an internal $\overline{IRQ2}$ interrupt line, as shown in Figure 1. External devices (and IS3) use $\overline{IRQ1}$. An $\overline{IRQ1}$ interrupt is serviced before $\overline{IRQ2}$ if both are pending.

All $\overline{IRQ2}$ interrupts use hardware prioritized vectors. The single SCI interrupt and three timer interrupts are serviced in a prioritized order and each is vectored to a separate location. All interrupt vector locations are shown in Table 5.

The interrupt flowchart is depicted in Figure 18 and is common to every interrupt excluding reset. During interrupt servicing the program counter, index register, A accumulator, B accumulator, and condition code register are pushed to the stack. The I bit is set to inhibit maskable interrupts and a vector is fetched corresponding to the current highest priority interrupt. The vector is transferred to the program counter and instruction execution is resumed. Interrupt and $\overline{RESET}$ timing are illustrated in Figures 19 and 20.

FUNCTIONAL PIN DESCRIPTIONS

V_{CC} AND V_{SS}

V_{CC} and V_{SS} provide power to a large portion of the MCU. The power supply should provide +5 volts (±5%) to V_{CC}, and V_{SS} should be tied to ground. Total power dissipation (including V_{CC} standby), will not exceed P_D milliwatts.

V_{CC} STANDBY

V_{CC} standby provides power to the standby portion ($80 through $BF) of the RAM and the STBY PWR and RAME bits of the RAM control register. Voltage requirements depend on whether the device is in a powerup or powerdown state. In the powerup state, the power supply should provide +5 volts (±5%) and must reach V_{SB} volts before $\overline{RESET}$ reaches 4.0 volts. During powerdown, V_{CC} standby must remain above V_{SBB} (min) to sustain the standby RAM and STBY PWR bit. While in powerdown operation, the standby current will not exceed I_{SBB}.

It is typical to power both V_{CC} and V_{CC} standby from the same source during normal operation. A diode must be used between them to prevent supplying power to V_{CC} during powerdown operation. V_{CC} standby should be tied to ground in mode 3.

TABLE 4 — INTERNAL REGISTER AREA

Register	Address
Port 1 Data Direction Register***	00
Port 2 Data Direction Register***	01
Port 1 Data Register	02
Port 2 Data Register	03
Port 3 Data Direction Register***	04*
Port 4 Data Direction Register***	05**
Port 3 Data Register	06*
Port 4 Data Register	07**
Timer Control and Status Register	08
Counter (High Byte)	09
Counter (Low Byte)	0A
Output Compare Register (High Byte)	0B
Output Compare Register (Low Byte)	0C
Input Capture Register (High Byte)	0D
Input Capture Register (Low Byte)	0E
Port 3 Control and Status Register	0F*
Rate and Mode Control Register	10
Transmit/Receive Control and Status Register	11
Receive Data Register	12
Transmit Data Register	13
RAM Control Register	14
Reserved	15-1F

* External addresses in modes 0, 1, 2, 3, 5, and 6; cannot be accessed in mode 5 (no $\overline{IOS}$).

** External addresses in modes 0, 1, 2, and 3.

*** 1 = Output, 0 = Input.

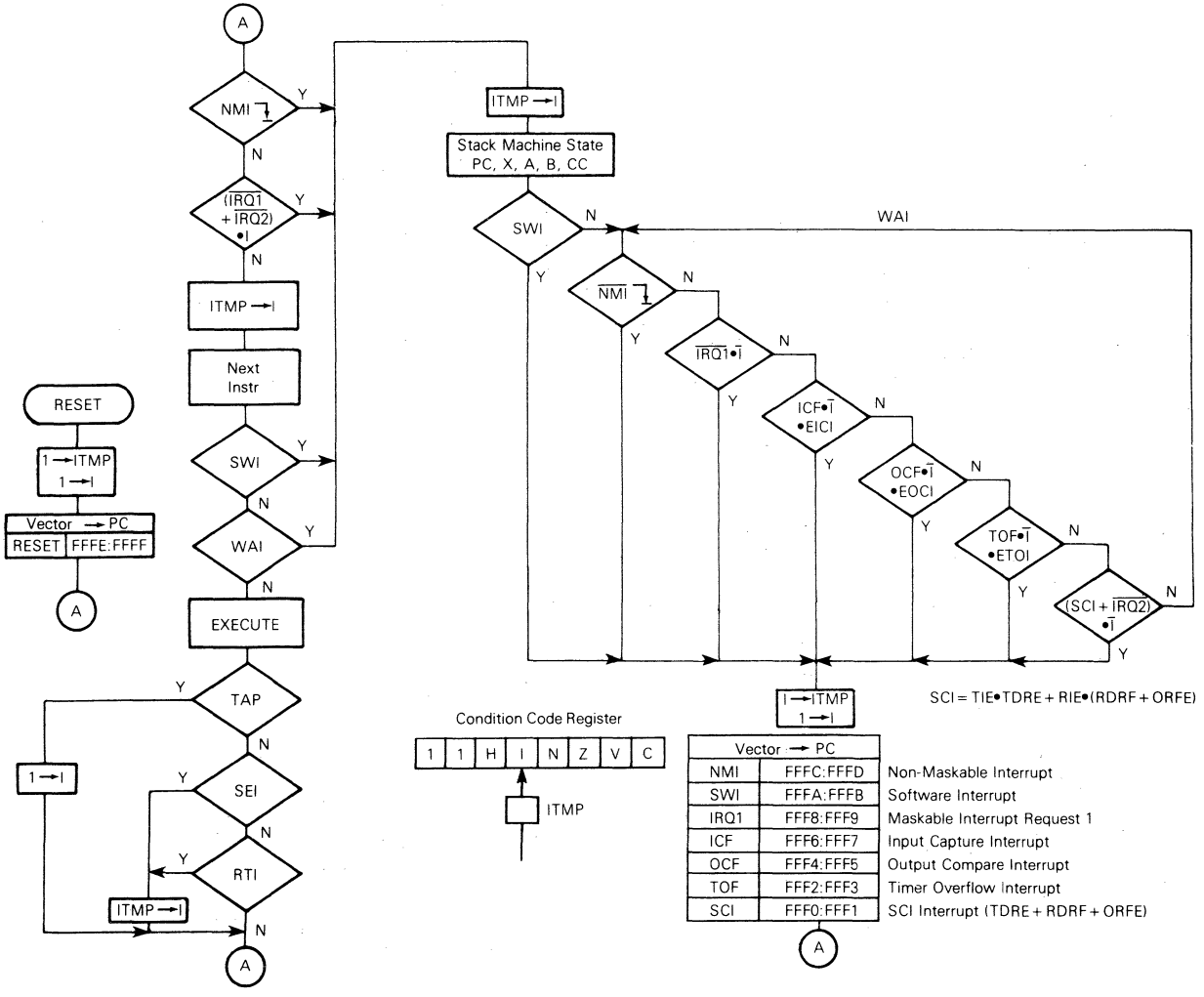
TABLE 5 — MCU INTERRUPT VECTOR LOCATIONS

MSB	LSB	Interrupt
FFFE	FFFF	$\overline{RESET}$
FFFC	FFFD	$\overline{NMI}$
FFFA	FFFB	Software Interrupt (SWI)
FFF8	FFF9	$\overline{IRQ1}$ (or $\overline{IS3}$)
FFF6	FFF7	ICF (Input Capture)*
FFF4	FFF5	OCF (Output Capture)*
FFF2	FFF3	TOF (Timer Overflow)*
FFF0	FFF1	SCI (RDRF + ORFE + TDRE)*

* $\overline{IRQ2}$ Interrupt

3

FIGURE 18 — INTERRUPT FLOWCHART



Vector → PC		
NMI	FFFC:FFFD	Non-Maskable Interrupt
SWI	FFFA:FFF8	Software Interrupt
IRQ1	FFF8:FFF9	Maskable Interrupt Request 1
ICF	FFF6:FFF7	Input Capture Interrupt
OCF	FFF4:FFF5	Output Compare Interrupt
TOF	FFF2:FFF3	Timer Overflow Interrupt
SCI	FFF0:FFF1	SCI Interrupt (TDRE + RDRF + ORFE)

FIGURE 19 — INTERRUPT SEQUENCE

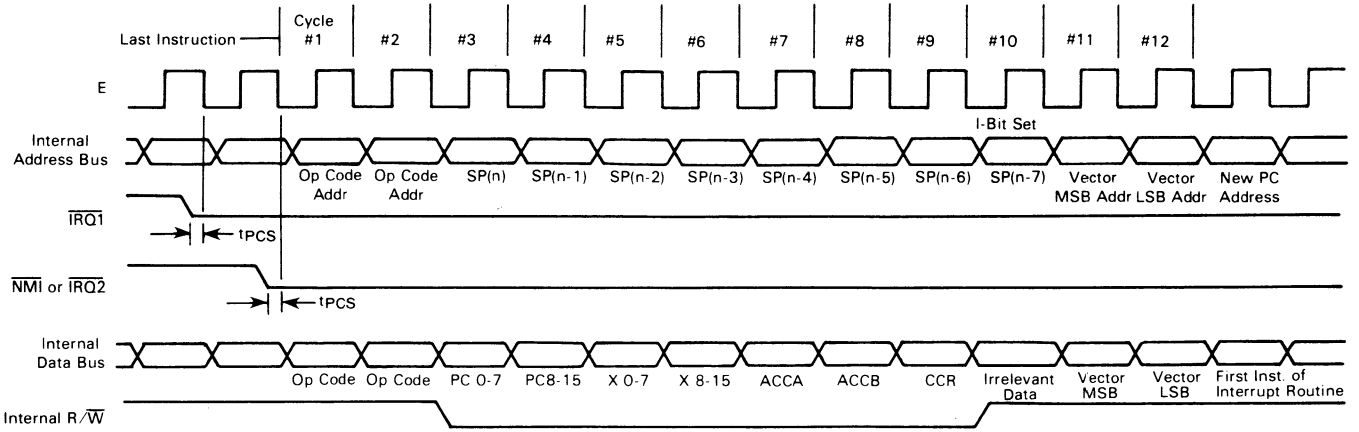
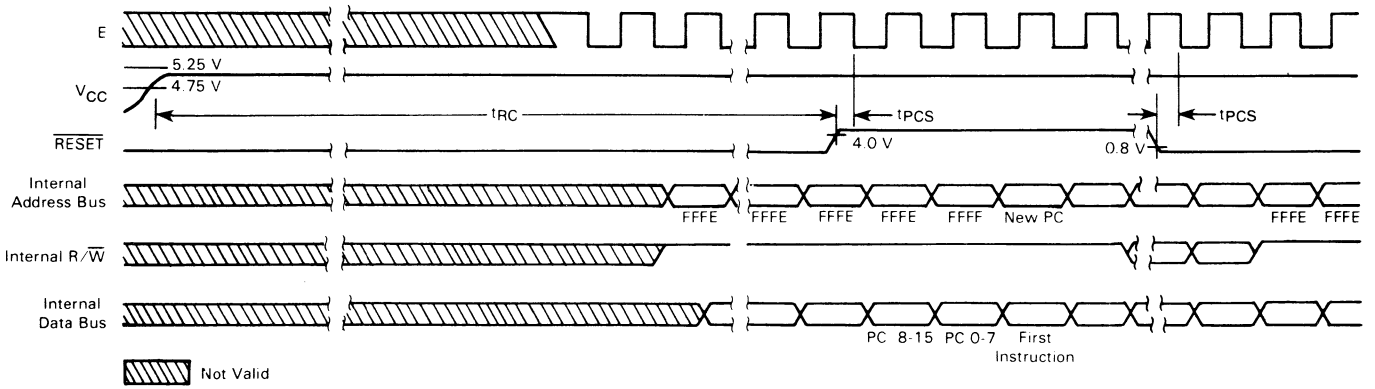


FIGURE 20 — RESET TIMING



XTAL AND EXTAL

These two input pins interface either a crystal or TTL-compatible clock to the MCU internal clock generator. Divide-by-four circuitry is included which allows use of the inexpensive 3.58 MHz or 4.4336 MHz Color Burst TV crystals. A 20 pF capacitor should be tied from each crystal pin to ground to ensure reliable startup and operation. Alternatively, EXTAL may be driven by an external TTL-compatible clock at $4f_O$ with a duty cycle of 50% ($\pm$5%) with XTAL connected to ground.

The internal oscillator is designed to interface with an AT-cut quartz crystal resonator operated in parallel resonance mode in the frequency range specified for f_{XTAL}. The crystal should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time.* The MCU is compatible with most commercially available crystals. Nominal crystal parameters are shown in Figure 21.

$\overline{\text{RESET}}$

This input is used to reset the internal state of the device and provide an orderly startup procedure. During powerup, $\overline{\text{RESET}}$ must be held below 0.8 volts: (1) at least t_{RC} after V_{CC} reaches 4.75 volts in order to provide sufficient time for the clock generator to stabilize, and (2) until V_{CC} standby reaches 4.75 volts. $\overline{\text{RESET}}$ must be held low at least three E cycles if asserted during powerup operation.

E (ENABLE)

This is an output clock used primarily for bus synchronization. It is TTL compatible and is the slightly skewed divide-by-four result of the device input clock frequency. It will drive one Schottky TTL load and 90 pF, and all data given in cycles is referenced to this clock unless otherwise noted.

NON-MASKABLE INTERRUPT ($\overline{\text{NMI}}$)

An $\overline{\text{NMI}}$ negative edge requests an MCU interrupt sequence, but the current instruction will be completed before it responds to the request. The MCU will then begin an interrupt sequence. Finally, a vector is fetched from $FFFC and $FFFD, transferred to the program counter and instruction execution is resumed. $\overline{\text{NMI}}$ typically requires a 3.3 kΩ (nominal) resistor to V_{CC}. There is no internal $\overline{\text{NMI}}$ pullup resistor. $\overline{\text{NMI}}$ must be held low for at least one E cycle to be recognized under all conditions.

MASKABLE INTERRUPT REQUEST 1 ($\overline{\text{IRQ1}}$)

$\overline{\text{IRQ1}}$ is a level-sensitive input which can be used to request an interrupt sequence. The MPU will complete the current instruction before it responds to the request. If the interrupt mask bit (I bit) in the condition code register is clear, the MCU will begin an interrupt sequence. A vector is fetched from $FFF8 and $FFF9, transferred to the program counter, and instruction execution is resumed.

$\overline{\text{IRQ1}}$ typically requires an external 3.3 kΩ (nominal) resistor to V_{CC} for wire-OR applications. $\overline{\text{IRQ1}}$ has no internal pullup resistor.

STROBE CONTROL 1 AND 2 (SC1 AND SC2)

The function of SC1 and SC2 depends on the operating mode. SC1 is configured as an output in all modes except single-chip mode, whereas SC2 is always an output. SC1 and SC2 can drive one Schottky load and 90 pF.

SC1 and SC2 In Single-Chip Mode

In single-chip mode, SC1 and SC2 are configured as an input and output, respectively, and both function as port 3 control lines. SC1 functions as $\overline{\text{IS3}}$ and can be used to indicate that port 3 input data is ready or output data has been accepted. Three options associated with $\overline{\text{IS3}}$ are controlled by port 3 control and status register and are discussed in the **PORT 3 (P30-P37)**. If unused, $\overline{\text{IS3}}$ can remain unconnected.

SC2 is configured as $\overline{\text{OS3}}$ and can be used to strobe output data or acknowledge input data. It is controlled by output strobe select (OSS) in the port 3 control and status register. The strobe is generated by a read (OSS = 0) or write (OSS = 1) to the port 3 data register. $\overline{\text{OS3}}$ timing is shown in Figure 4.

SC1 and SC2 In Expanded Non-Multiplexed Mode

In the expanded non-multiplexed mode, both SC1 and SC2 are configured as outputs. SC1 functions as input/output select ($\overline{\text{IOS}}$) and is asserted only when $0100 through $01FF is sensed on the internal address bus.

SC2 is configured as read/write and is used to control the direction of data bus transfers. An MPU read is enabled when read/write and E are high.

SC1 and SC2 In Expanded-Multiplexed Mode

In the expanded-multiplexed mode, both SC1 and SC2 are configured as outputs. SC1 functions as address strobe and can be used to demultiplex the eight least-significant addresses and the data bus. A latch controlled by address strobe captures address on the negative edge, as shown in Figure 14.

SC2 is configured as read/write and is used to control the direction of data bus transfers. An MPU read is enabled when read/write and E are high.

PORT 1 (P10-P17)

Port 1 is a mode independent 8-bit I/O port with each line an input or output as defined by the port 1 data direction register. The TTL compatible three-state output buffers can drive one Schottky TTL load and 30 pF, Darlington transistors, or CMOS devices using external pullup resistors. It is configured as a data input port by $\overline{\text{RESET}}$. Unused lines can remain unconnected.

PORT 2 (P20-P24)

PORT 2 DATA REGISTER

7	6	5	4	3	2	1	0	
PC2	PC1	PC0	P24	P23	P22	P21	P20	$0003

Port 2 is a mode-independent, 5-bit, multi-purpose I/O port. The voltage levels present on P20, P21, and P22 on the rising edge of $\overline{\text{RESET}}$ determine the operating mode of the MCU. The entire port is then configured as a data input port. The port 2 lines can be selectively configured as data output lines by setting the appropriate bits in the port 2 data direction register. The port 2 data register is used to move data through the port. However, if P21 is configured as an output, it will be tied to the timer output compare function and cannot be used to provide output from the port 2 data register.

*Devices made with masks subsequent to M5G, M8D, and T5P incorporate an advanced clock with improved startup characteristics.

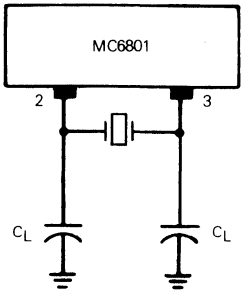
FIGURE 21 — M6801 FAMILY OSCILLATOR CHARACTERISTICS

(a) Nominal Recommended Crystal Parameters

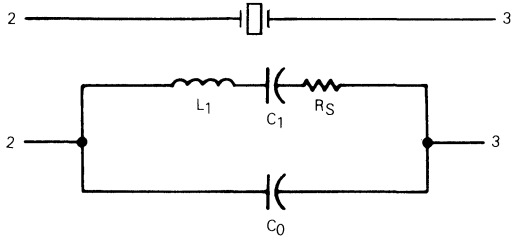
Nominal Crystal Parameters*

	3.58 MHz	4.00 MHz	5.0 MHz	6.0 MHz	8.0 MHz
R_S	60 Ω	50 Ω	30-50 Ω	30-50 Ω	20-40 Ω
C_0	3.5 pF	6.5 pF	4-6 pF	4-6 pF	4-6 pF
C_1	0.015 pF	0.025 pF	0.01-0.02 pF	0.01-0.02 pF	0.01-0.02 pF
Q	>40 K	>30 K	>20 K	>20 K	>20 K

*NOTE: These are representative AT-cut crystal parameters only. Crystals of other types of cut may also be used.



C_L = 20 pF (typical)



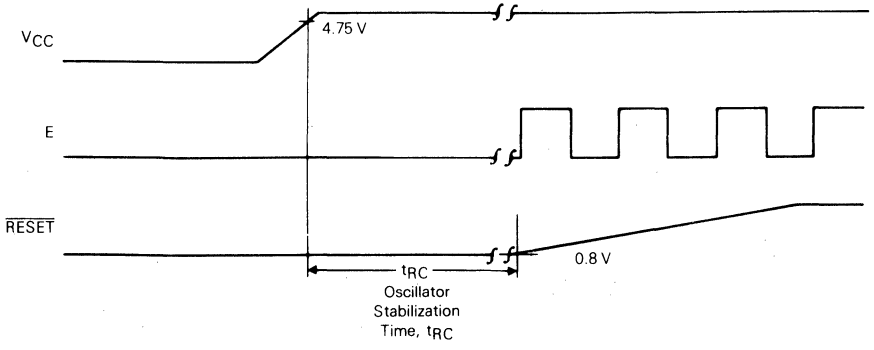
Equivalent Circuit

NOTE

TTL-compatible oscillators may be obtained from:

Motorola Component Products
Attn: Data Clock Sales
2553 N. Edgington St.
Franklin Park, IL 60131
Tel: 312-451-1000
Telex: 433-0067

(b) Oscillator Stabilization Time (t_{RC})



Port 2 can also be used to provide an interface for the serial communications interface and the timer input edge function. These configurations are described in **PROGRAMMABLE TIMER** and **SERIAL COMMUNICATIONS INTERFACE (SCI).**

The port 2 high-impedance TTL-compatible output buffers are capable of driving one Schottky TTL load and 30 pF, or CMOS devices using external pullup resistors.

PORT 3 (P30-P37)

Port 3 can be configured as an I/O port, a bidirectional 8-bit data bus, or a multiplexed address/data bus depending on the operating mode. The TTL-compatible high-impedance output buffers can drive one Schottky TTL load and 90 pF. Unused lines can remain unconnected.

Port 3 In Single-Chip Mode

Port 3 is an 8-bit I/O port in the single-chip mode, with each line configured by the port 3 data direction register. There are also two lines, $\overline{IS3}$ and $\overline{OS3}$, which can be used to control port 3 data transfers.

Three port 3 options are controlled by the port 3 control and status register and are available only in single-chip mode: (1) port 3 input data can be latched using $\overline{IS3}$ as a control signal, (2) $\overline{OS3}$ can be generated by either an MPU read or write to the port 3 data register, and (3) an $\overline{IRQ1}$ interrupt can be enabled by an $\overline{IS3}$ negative edge. Port 3 latch timing is shown in Figure 5.

PORT 3 CONTROL AND STATUS REGISTER

7	6	5	4	3	2	1	0	
IS3 Flag	IS3 IRQ1 Enable	X	OSS	Latch Enable	X	X	X	$000F

Bit 0-2 Not used.

Bit 3 LATCH ENABLE. This bit controls the input latch for port 3. If set, input data is latched by an $\overline{IS3}$ negative edge. The latch is transparent after a read of the port 3 data register. LATCH ENABLE is cleared during reset.

Bit 4 OSS (Output Strobe Select). This bit determines whether $\overline{OS3}$ will be generated by a read or write of the port 3 data register. When clear, the strobe is generated by a read; when set, it is generated by a write. OSS is cleared during reset.

Bit 5 Not used.

Bit 6 IS3 IRQ1 ENABLE. When set, an $\overline{IRQ1}$ interrupt will be enabled whenever IS3 FLAG is set; when clear, the interrupt is inhibited. This bit is cleared during reset.

Bit 7 IS3 FLAG. This read-only status bit is set by an $\overline{IS3}$ negative edge. It is cleared by a read of the port 3 control and status register (with IS3 FLAG set) followed by a read or write to the port 3 data register or during reset.

Port 3 In Expanded Non-Multiplexed Mode

Port 3 is configured as a bidirectional data bus (D7-D0) in the expanded non-multiplexed mode. The direction of data transfers is controlled by read/write (SC2). Data is clocked by E (enable).

Port 3 In Expanded-Multiplexed Mode

Port 3 is configured as a time multiplexed address (A0-A7) and data bus (D7-D0) in the expanded-multiplexed modes, where address strobe (AS) can be used to demultiplex the two buses. Port 3 is held in a high-impedance state between valid address and data to prevent bus conflicts.

PORT 4 (P40-P47)

Port 4 is configured as an 8-bit I/O port, as address outputs, or as data inputs depending on the operating mode. Port 4 can drive one Schottky TTL load and 90 pF and is the only port with internal pullup resistors. Unused lines can remain unconnected.

Port 4 In Single-Chip Mode

In single-chip mode, port 4 functions as an 8-bit I/O port with each line configured by the port 4 data direction register. Internal pullup resistors allow the port to directly interface with CMOS at 5 volt levels. External pullup resistors to more than 5 volts, however, cannot be used.

Port 4 In Expanded Non-Multiplexed Mode

Port 4 is configured from reset as an 8-bit input port, where the port 4 data direction register can be written to provide any or all of eight address lines, A0 to A7. Internal pullup resistors pull the lines high until the port 4 data direction register is configured.

Port 4 In Expanded-Multiplexed Mode

In all expanded-multiplexed modes except mode 6, port 4 functions as half of the address bus and provides A8 to A15. In mode 6, the port is configured from reset as an 8-bit parallel input port, where the port 4 data direction register can be written to provide any or all of upper address lines A8 to A15. Internal pullup resistors pull the lines high until the port 4 data direction register is configured, where bit 0 controls A8.

RESIDENT MEMORY

The MC6801 provides 2048 bytes of on-chip ROM and 128 bytes of on-chip RAM.

One half of the RAM is powered through the V_{CC} standby pin and is maintainable during V_{CC} powerdown. This standby portion of the RAM consists of 64 bytes located from $80 through $BF.

Power must be supplied to V_{CC} standby if the internal RAM is to be used regardless of whether standby power operation is anticipated.

The RAM is controlled by the RAM control register.

RAM CONTROL REGISTER ($14)

The RAM control register includes two bits which can be used to control RAM accesses and determine the adequacy of the standby power source during powerdown operation. It is intended that RAME be cleared and STBY PWR be set as part of a powerdown procedure.

RAM CONTROL REGISTER

7	6	5	4	3	2	1	0
STBY PWR	RAME	X	X	X	X	X	X

Bit 0-5 Not used.

Bit 6 RAME RAM Enable. This read/write bit can be used to remove the entire RAM from the internal memory map. RAME is set (enabled) during reset provided standby power is available on the positive edge of $\overline{RESET}$. If RAME is clear, any access to a RAM address is external. If RAME is set and not in mode 3, the RAM is included in the internal map.

Bit 7 STBY PWR Standby Power. This bit is a read/write status bit which, when once set, remains set as long as V_{CC} standby remains above V_{SBB} (minimum). As long as this bit is set following a period of standby operation, the standby power supply has adequately preserved the data in the standby RAM. If this bit is cleared during a period of standby operation, it indicates that V_{CC} standby had fallen to a level sufficiently below V_{SBB} (minimum) to suspect that data in the standby RAM is not valid. This bit can be set only by software and is not affected during reset.

PROGRAMMABLE TIMER

The programmable timer can be used to perform input waveform measurements while independently generating an output waveform. Pulse widths can vary from several microseconds to many seconds. A block diagram of the timer is shown in Figure 22.

COUNTER ($09:0A)

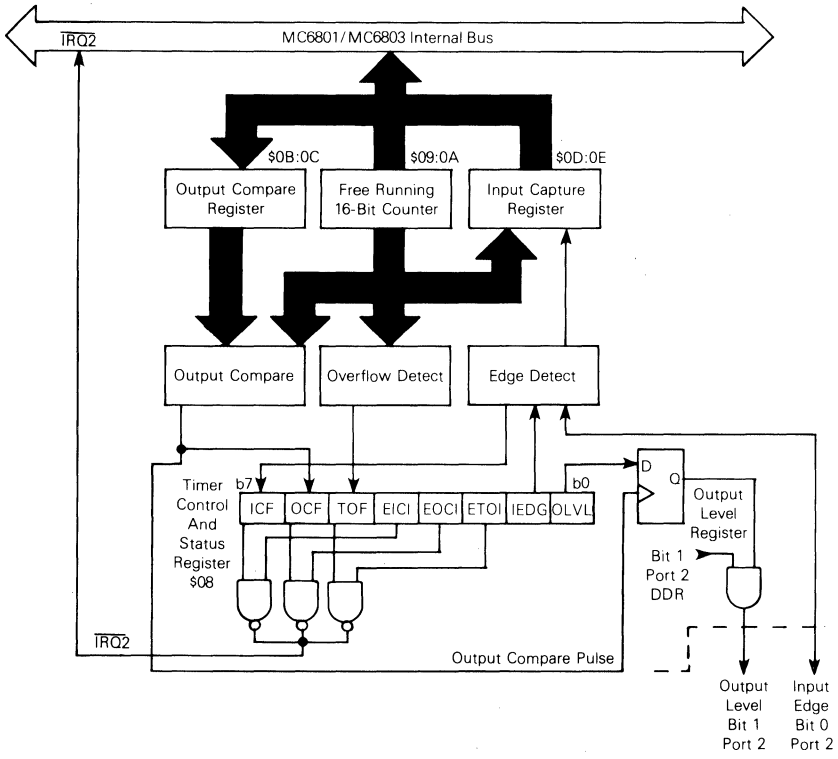
The key timer element is a 16-bit free-running counter which is incremented by E (enable). It is cleared during reset and is read-only with one exception: a write to the counter ($09) will preset it to $FFF8. This feature, intended for testing, can disturb serial operations because the counter provides the SCI internal bit rate clock. TOF is set whenever the counter contains all ones.

OUTPUT COMPARE REGISTER ($0B:0C)

The output compare register is a 16-bit read/write register used to control an output waveform or provide an arbitrary timeout flag. It is compared with the free-running counter on each E cycle. When a match occurs, OCF is set and OLVL is clocked to an output level register. If port 2, bit 1, is configured as an output, OLVL will appear at P21 and the output compare register and OLVL can then be changed for the next

3

FIGURE 22 — BLOCK DIAGRAM OF PROGRAMMABLE TIMER



compare. The function is inhibited for one cycle after a write to its high byte ($0B) to ensure a valid compare. The output compare register is set to $FFFF at $\overline{RESET}$.

INPUT CAPTURE REGISTER ($0D:0E)

The input capture register is a 16-bit read-only register used to store the free-running counter when a "proper" input transition occurs as defined by IEDG. Port 2, bit 0 should be configured as an input, but the edge detect circuit always senses P20 even when configured as an output. An input capture can occur independently of ICF: the register always contains the most current value. Counter transfer is inhibited, however, between accesses of a double byte MPU read. The input pulse width must be at least two E cycles to ensure an input capture under all conditions.

TIMER CONTROL AND STATUS REGISTER ($08)

The timer control and status register (TCSR) is an 8-bit register of which all bits are readable, while only bits 0-4 can be written. The three most-significant bits provide the timer status and indicate if:

- a proper level transition has been detected,
- a match has occurred between the free-running counter and the output compare register, and
- the free-running counter has overflowed.

Each of the three events can generate an $\overline{IRQ2}$ interrupt and is controlled by an individual enable bit in the TCSR.

TIMER CONTROL AND STATUS REGISTER (TCSR)

7	6	5	4	3	2	1	0	
ICF	OCF	TOF	EICI	EOCI	ETOI	IEDG	OLVL	$0008

Bit 0 OLVL — Output Level. OLVL is clocked to the output level register by a successful output compare and will appear at P21 if bit 1 of the port 2 data direction register is set. It is cleared during reset.

Bit 1 EIDG — Input Edge. IEDG is cleared during reset and controls which level transition will trigger a counter transfer to the input capture register:
IEDG = 0 Transfer on a negative-edge
IEDG = 1 Transfer on a positive-edge.

Bit 2 ETOI — Enable Timer Overflow Interrupt. When set, an $\overline{IRQ2}$ interrupt is enabled for a timer overflow; when clear, the interrupt is inhibited. It is cleared during reset.

Bit 3 EOCI — Enable Output Compare Interrupt. When set, an $\overline{IRQ2}$ interrupt is enabled for an output compare; when clear, the interrupt is inhibited. It is cleared during reset.

Bit 4 EICI — Enable Input Capture Interrupt. When set, an $\overline{IRQ2}$ interrupt is enabled for an input capture; when clear, the interrupt is inhibited. It is cleared during reset.

Bit 5 TOF — Timer Overflow Flag. TOF is set when the counter contains all ones. It is cleared by reading the TCSR (with TOF set) then reading the counter high byte ($09), or during reset.

Bit 6 OCF — Output Compare Flag. OCF is set when the output compare register matches the free-running counter. It is cleared by reading the TCSR (with OCF set) and then writing to the output compare register ($0B or $0C), or during reset.

Bit 7 ICF — Input Capture Flag. ICF is set to indicate a proper level transition; it is cleared by reading the TCSR (with ICF set) and then the input capture register high byte ($0D), or during reset.

SERIAL COMMUNICATIONS INTERFACE (SCI)

A full-duplex asynchronous serial communications interface (SCI) is provided with two data formats and a variety of rates. The SCI transmitter and receiver are functionally independent, but use the same data format and bit rate. Serial data formats include standard mark/space (NRZ) and Biphase and both provide one start bit, eight data bits, and one stop bit. "Baud" and "bit rate" are used synonymously in the following description.

WAKE-UP FEATURE

In a typical serial loop multi-processor configuration, the software protocol will usually identify the addressee(s) at the beginning of the message. In order to permit uninterested MPU's to ignore the remainder of the message, a wake-up feature is included whereby all further SCI receiver flag (and interrupt) processing can be inhibited until its data line goes idle. An SCI receiver is re-enabled by an idle string of eleven consecutive ones or during reset. Software must provide for the required idle string between consecutive messages and prevent it within messages.

PROGRAMMABLE OPTIONS

The following features of the SCI are programmable:

- format: standard mark/space (NRZ) or Bi-phase
- clock: external or internal bit rate clock
- Baud: one of four per E clock frequency, or external clock (×8 desired baud)
- wake-up feature: enabled or disabled
- interrupt requests: enabled individually for transmitter and receiver
- clock output: internal bit rate clock enabled or disabled to P22

SERIAL COMMUNICATIONS REGISTERS

The serial communications interface includes four addressable registers as depicted in Figure 23. It is controlled by the rate and mode control register and the transmit/receive control and status register. Data is transmitted and

received utilizing a write-only transmit register and a read-only receive register. The shift registers are not accessible to software.

Rate and Mode Control Registers (RMCR) ($10)

The rate and mode control register controls the SCI bit rate, format, clock source, and under certain conditions, the configuration of P22. The register consists of four write-only bits which are cleared during reset. The two least-significant bits control the bit rate of the internal clock and the remaining two bits control the format and clock source.

RATE AND MODE CONTROL REGISTER (RMCR)

7	6	5	4	3	2	1	0	
X	X	X	X	CC1	CC0	SS1	SS0	$0010

Bit 1:Bit 0 — SS1:SS0 Speed Select. These two bits select the baud rate when using the internal clock. Four rates may be selected which are a function of the MCU input frequency. Table 6 lists bit time and rates for three selected MCU frequencies.

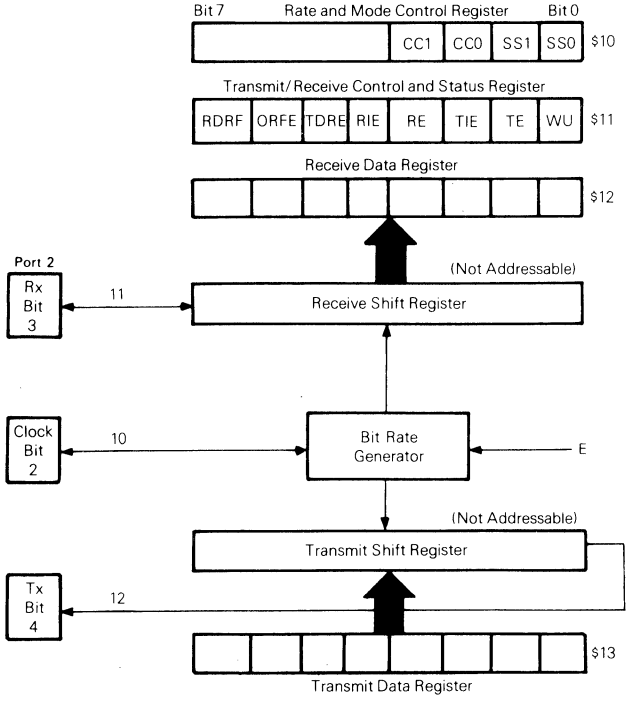
Bit 3:Bit 2 — CC1:CC0 Clock Control and Format Select. These two bits control the format and select the serial clock source. If CC1 is set, the DDR value for P22 is forced to the complement of CC0 and cannot be altered until CC1 is cleared. If CC1 is cleared after having been set, its DDR value is unchanged. Table 7 defines the formats, clock source, and use of P22.

If both CC1 and CC0 are set, an external TTL-compatible clock must be connected to P22 at eight times (8X) the desired bit rate, but not greater than E, with a duty cycle of 50% (±10%). If CC1:CC0=10, the internal bit rate clock is provided at P22 regardless of the values for TE or RE.

NOTE: The source of SCI internal bit rate clock is the timer free-running counter. An MPU write to the counter can disturb serial operations.

3

FIGURE 23 – SCI REGISTERS



Transmit/Receive Control And Status Register (TRCSR) ($11)

The transmit/receive control and status register controls the transmitter, receiver, wake-up feature, and two individual interrupts and monitors the status of serial operations. All eight bits are readable while bits 0 to 4 are also writable. The register is initialized to $20 by $\overline{\text{RESET}}$.

TRANSMIT/RECEIVE CONTROL AND STATUS REGISTER (TRCSR)

7	6	5	4	3	2	1	0	
RDRF	ORFE	TDRE	RIE	RE	TIE	TE	WU	$0011

Bit 0 WU — "Wake-up" on Idle Line. When set, WU enables the wake-up function; it is cleared by eleven consecutive ones or during reset. WU will not set if the line is idle.

Bit 1 TE — Transmit Enable. When set, P24 DDR bit is set, cannot be changed, and will remain set if TE is subsequently cleared. When TE is changed from clear to set, the transmitter is connected to P24 and a preamble of nine consecutive ones is transmitted. TE is cleared during reset.

Bit 2 TIE — Transmit Interrupt Enable. When set, an $\overline{\text{IRQ2}}$ interrupt is enabled when TDRE is set; when clear, the interrupt is inhibited. TE is cleared during reset.

Bit 3 RE — Receive Enable. When set, the P23 DDR bit is cleared, cannot be changed, and will remain clear if RE is subsequently cleared. While RE is set, the SCI receiver is enabled. RE is cleared during reset.

Bit 4 RIE — Receiver Interrupt Enable. When set, an $\overline{\text{IRQ2}}$ interrupt is enabled when RDRF and/or ORFE is set; when clear, the interrupt is inhibited. RIE is cleared during reset.

Bit 5 TDRE — Transmit Data Register Empty. TDRE is set when the transmit data register is transferred to the output serial shift register or during reset. It is cleared by reading the TRCSR (with TDRE set) and then writing to the transmit data register. Additional data will be transmitted only if TDRE has been cleared.

Bit 6 ORFE — Overrun Framing Error. If set, ORFE indicates either an overrun or framing error. An overrun is a new byte ready to transfer to the receiver data register with RDRF still set. A receiver framing error has occurred when the byte boundaries of the bit stream are not synchronized to the bit counter. An overrun can be distinguished from a framing error by the state of RDRF: if RDRF is set, then an overrun has occurred; otherwise a framing error has been detected. Data is not transferred to the receive data register in an overrun condition. Unframed data causing a framing error is transferred to the receive data register. However, subsequent data transfer is blocked until the framing error flag is cleared.* ORFE is cleared by reading the TRCSR (with ORFE set) then the receive data register, or during reset.

Bit 7 RDRF — Receive Data Register Full. RDRF is set when the input serial shift register is transferred to the receive data register. It is cleared by reading the TRCSR (with RDRF set), and then the receive data register, or during reset.

TABLE 6 — SCI BIT TIMES AND RATES

SS1:SS0		$4f_O \rightarrow$	2.4576 MHz	4.0 MHz	4.9152 MHz
		E	614.4 kHz	1.0 MHz	1.2288 MHz
0	0	÷16	26 μs/38,400 Baud	16 μs/62,500 Baud	13.0 μs/76,800 Baud
0	1	÷128	208 μs/4,800 Baud	128 μs/7812.5 Baud	104.2 μs/9,600 Baud
1	0	÷1024	1.67 ms/600 Baud	1.024 ms/976.6 Baud	833.3 μs/1,200 Baud
1	1	÷4096	6.67 ms/150 Baud	4.096 ms/244.1 Baud	3.33 ms/300 Baud
*External (P22)			13.0 μs/76,800 Baud	8.0 μs/125,000 Baud	6.5 μs/153,600 Baud

*Using maximum clock rate

TABLE 7 — SCI FORMAT AND CLOCK SOURCE CONTROL

CC1:CC0	Format	Clock Source	Port 2 Bit 2
00	Bi-Phase	Internal	Not Used
01	NRZ	Internal	Not Used
10	NRZ	Internal	Output
11	NRZ	External	Input

*Devices made with mask number M5G, M8D, and T5P do not transfer unframed data to the receive data register.

SERIAL OPERATIONS

The SCI is initialized by writing control bytes first to the rate and mode control register and then to the transmit/receive control and status register. When TE is set, the output of the transmit serial shift register is connected to P24 and serial output is initiated by transmitting a 9-bit preamble of ones.

At this point one of two situations exist: 1) if the transmit data register is empty (TDRE = 1), a continuous string of ones will be sent indicating an idle line, or 2) if a byte has been written to the transmit-data register (TDRE = 0), it will be transferred to the output serial shift register (synchronized with the bit rate clock), TDRE will be set, and transmission will begin.

The start bit (0), eight data bits (beginning with bit 0) and a stop bit (1), will be transmitted. If TDRE is still set when the next byte transfer should occur, ones will be sent until more data is provided. In Bi-phase format, the output toggles at the start of each bit and at half-bit time when a one is sent. Receive operation is controlled by RE which configures P23 as an input and enables the receiver. SCI data formats are illustrated in Figure 24.

INSTRUCTION SET

The MC6801/03 is upward source and object code compatible with the MC6800. Execution times of key instructions have been reduced and several new instructions have been added, including a hardware multiply. A list of new operations added to the MC6800 instruction set is shown in Table 1.

In addition, two new special opcodes, 4E and 5E, are provided for test purposes. These opcodes force the program counter to increment like a 16-bit counter, causing address lines used in the expanded modes to increment until the device is reset. These opcodes have no mnemonics.

The coding of the first (or only) byte corresponding to an executable instruction is sufficient to identify the instruction and the addressing mode. The hexadecimal equivalents of the binary codes, which result from the translation of the 82 instructions in all valid modes of addressing, are shown in Table 8. There are 220 valid machine codes, 34 unassigned codes, and 2 codes reserved for test purposes.

PROGRAMMING MODEL

A programming model for the MC6801/03 is shown in Figure 10. Accumulator A can be concatenated with accumulator B and jointly referred to as accumulator D where A is the most-significant byte. Any operation which modifies the double accumulator will also modify accumulator A and/or B. Other registers are defined as follows:

Program Counter – The program counter is a 16-bit register which always points to the next instruction.

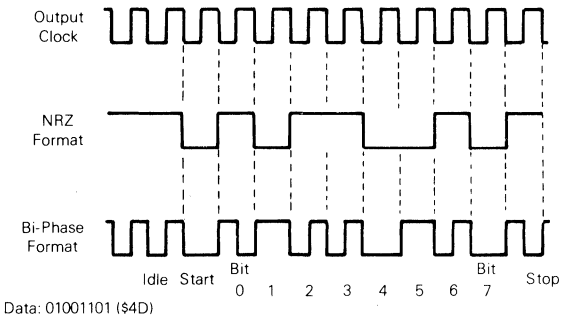
Stack Pointer – The stack pointer is a 16-bit register which contains the address of the next available location in a pushdown/pullup (LIFO) queue. The stack resides in random access memory at a location defined by the programmer.

Index Register – The index register is a 16-bit register which can be used to store data or provide an address for the indexed mode of addressing.

Accumulators – The MPU contains two 8-bit accumulators, A and B, which are used to store operands and results from the arithmetic logic unit (ALU). They can also be concatenated and referred to as the D (double) accumulator.

Condition Code Registers – The condition code register indicates the results of an instruction and includes the following five condition bits: negative (N), zero (Z), overflow (V), carry/borrow from MSB (C), and half carry from bit 3 (H). These bits are testable by the conditional branch instructions. Bit 4 is the interrupt mask (I bit) and inhibits all maskable interrupts when set. The two unused bits, B6 and B7, are read as ones.

FIGURE 24 – SCI DATA FORMATS



ADDRESSING MODES

Six addressing modes can be used to reference memory. A summary of addressing modes for all instructions is present in Tables 9 through 12, where execution times are provided in E cycles. Instruction execution times are summarized in Table 13. With an input frequency of 4 MHz, E cycles are equivalent to microseconds. A cycle-by-cycle description of bus activity for each instruction is provided in Table 14 and a description of selected instructions is shown in Figure 25.

Immediate Addressing — The operand or "immediate byte(s)" is contained in the following byte(s) of the instruction where the number of bytes matches the size of the register. These are two or three byte instructions.

Direct Addressing — The least-significant byte of the operand address is contained in the second byte of the instruction and the most-significant byte is assumed to be $00. Direct addressing allows the user to access $00 through $FF using two byte instructions and execution time is reduced by eliminating the additional memory access. In most applications, the 256-byte area is reserved for frequently referenced data.

Extended Addressing — The second and third bytes of the instruction contain the absolute address of the operand. These are three byte instructions.

Indexed Addressing — The unsigned offset contained in the second byte of the instruction is added with carry to the index register and used to reference memory without changing the index register. These are two byte instructions.

Inherent Addressing — The operand(s) are registers and no memory reference is required. These are single byte instructions.

Relative Addressing — Relative addressing is used only for branch instructions. If the branch condition is true, the program counter is overwritten with the sum of a signed single byte displacement in the second byte of the instruction and the current program counter. This provides a branch range of −126 to +129 bytes from the first byte of the instruction. These are two byte instructions.

TABLE 8 — CPU INSTRUCTION MAP

OP	MNEM	MODE	~	#	OP	MNEM	MODE	~	#	OP	MNEM	MODE	~	#	OP	MNEM	MODE	~	#	OP	MNEM	MODE	~	#
00	•				34	DES	INHER	3	1	68	ASL	INDXD	6	2	9C	CPX	DIR	5	2	D0	SUBB	DIR	3	2
01	NOP	INHER	2	1	35	TXS		3	1	69	ROL		6	2	9D	JSR		5	2	D1	CMPB		3	2
02	•				36	PSHA		3	1	6A	DEC		6	2	9E	LDS		4	2	D2	SBCB		3	2
03	•				37	PSHB		3	1	6B	•				9F	STS	DIR	4	2	D3	ADDD		5	2
04	LSRD		3	1	38	PULX		5	1	6C	INC		6	2	A0	SUBA	INDXD	4	2	D4	ANDB		3	2
05	ASLD		3	1	39	RTS		5	1	6D	TST		6	2	A1	CMPA		4	2	D5	BITB		3	2
06	TAP		2	1	3A	ABX		3	1	6E	JMP		3	2	A2	SBCA		4	2	D6	LDAB		3	2
07	TPA		2	1	3B	RTI		10	1	6F	CLR	INDXD	6	2	A3	SUBD		6	2	D7	STAB		3	2
08	INX		3	1	3C	PSHX		4	1	70	NEG	EXTND	6	3	A4	ANDA		4	2	D8	EORB		3	2
09	DEX		3	1	3D	MUL		10	1	71	•				A5	BITA		4	2	D9	ADCB		3	2
0A	CLV		2	1	3E	WAI		9	1	72	•				A6	LDAA		4	2	DA	ORAB		3	2
0B	SEV		2	1	3F	SWI		12	1	73	COM		6	3	A7	STAA		4	2	DB	ADDB		3	2
0C	CLC		2	1	40	NEGA		2	1	74	LSR		6	3	A8	EORA		4	2	DC	LDD		4	2
0D	SEC		2	1	41	•				75	•				A9	ADCA		4	2	DD	STD		4	2
0E	CLI		2	1	42	•				76	ROR		6	3	AA	ORAA		4	2	DE	LDX		4	2
0F	SEI		2	1	43	COMA		2	1	77	ASR		6	3	AB	ADDA		4	2	DF	STX	DIR	4	2
10	SBA		2	1	44	LSRA		2	1	78	ASL		6	3	AC	CPX		6	2	E0	SUBB	INDXD	4	2
11	CBA		2	1	45	•				79	ROL		6	3	AD	JSR		6	2	E1	CMPB		4	2
12	•				46	RORA		2	1	7A	DEC		6	3	AE	LDS		5	2	E2	SBCB		4	2
13	•				47	ASRA		2	1	7B	•				AF	STS	INDXD	5	2	E3	ADDD		6	2
14	•				48	ASLA		2	1	7C	INC		6	3	B0	SUBA	EXTND	4	3	E4	ANDB		4	2
15	•				49	ROLA		2	1	7D	TST		6	3	B1	CMPA		4	3	E5	BITB		4	2
16	TAB		2	1	4A	DECA		2	1	7E	JMP		3	3	B2	SBCA		4	3	E6	LDAB		4	2
17	TBA		2	1	4B	•				7F	CLR	EXTND	6	3	B3	SUBD		6	3	E7	STAB		4	2
18	•				4C	INCA		2	1	80	SUBA	IMMED	2	2	B4	ANDA		4	3	E8	EORB		4	2
19	DAA	INHER	2	1	4D	TSTA		2	1	81	CMPA		2	2	B5	BITA		4	3	E9	ADCB		4	2
1A	•				4E	T				82	SBCA		2	2	B6	LDAA		4	3	EA	ORAB		4	2
1B	ABA	INHER	2	1	4F	CLRA		2	1	83	SUBD		4	3	B7	STAA		4	3	EB	ADDB		4	2
1C	•				50	NEGB		2	1	84	ANDA		2	2	B8	EORA		4	3	EC	LDD		5	2
1D	•				51	•				85	BITA		2	2	B9	ADCA		4	3	ED	STD		5	2
1E	•				52	•				86	LDAA		2	2	BA	ORAA		4	3	EE	LDX		5	2
1F	•				53	COMB		2	1	87	•				BB	ADDA		4	3	EF	STX	INDXD	5	2
20	BRA	REL	3	2	54	LSRB		2	1	88	EORA		2	2	BC	CPX		6	3	F0	SUBB	EXTND	4	3
21	BRN		3	2	55	•				89	ADCA		2	2	BD	JSR		6	3	F1	CMPB		4	3
22	BHI		3	2	56	RORB		2	1	8A	ORAA		2	2	BE	LDS		5	3	F2	SBCB		4	3
23	BLS		3	2	57	ASRB		2	1	8B	ADDA		2	2	BF	STS	EXTND	5	3	F3	ADDD		6	3
24	BCC		3	2	58	ASLB		2	1	8C	CPX	IMMED	4	3	C0	SUBB	IMMED	2	2	F4	ANDB		4	3
25	BCS		3	2	59	ROLB		2	1	8D	BSR	REL	6	2	C1	CMPB		2	2	F5	BITB		4	3
26	BNE		3	2	5A	DECB		2	1	8E	LDS	IMMED	3	3	C2	SBCB		2	2	F6	LDAB		4	3
27	BEQ		3	2	5B	•				8F	•				C3	ADDD		4	3	F7	STAB		4	3
28	BVC		3	2	5C	INCB		2	1	90	SUBA	DIR	3	2	C4	ANDB		2	2	F8	EORB		4	3
29	BVS		3	2	5D	TSTB		2	1	91	CMPA		3	2	C5	BITB		2	2	F9	ADCB		4	3
2A	BPL		3	2	5E	T				92	SBCA		3	2	C6	LDAB		2	2	FA	ORAB		4	3
2B	BMI		3	2	5F	CLRB	INHER	2	1	93	SUBD		5	2	C7	•				FB	ADDB		4	3
2C	BGE		3	2	60	NEG	INDXD	6	2	94	ANDA		3	2	C8	EORB		2	2	FC	LDD		5	3
2D	BLT		3	2	61	•				95	BITA		3	2	C9	ADCB		2	2	FD	STD		5	3
2E	BGT		3	2	62	•				96	LDAA		3	2	CA	ORAB		2	2	FE	LDX		5	3
2F	BLE	REL	3	2	63	COM		6	2	97	STAA		3	2	CB	ADDB		2	2	FF	STX	EXTND	5	3
30	TSX	INHER	3	1	64	LSR		6	2	98	EORA		3	2	CC	LDD		3	3					
31	INS		3	1	65	•				99	ADCA		3	2	CD	•								
32	PULA		4	1	66	ROR		6	2	9A	ORAA		3	2	CE	LDX	IMMED	3	3					
33	PULB		4	1	67	ASR	INDXD	6	2	9B	ADDA		3	2	CF	•								

* UNDEFINED OP CODE

NOTES: 1. Addressing Modes
INHER ≡ Inherent INDXD ≡ Indexed IMMED ≡ Immediate
REL ≡ Relative EXTND ≡ Extended DIR ≡ Direct
2. Unassigned opcodes are indicated by "•" and should not be executed.
3. Codes marked by "T" force the PC to function as a 16-bit counter.

TABLE 9 — INDEX REGISTER AND STACK MANIPULATION INSTRUCTIONS

Pointer Operations	MNEM	Immed			Direct			Index			Extnd			Inherent			Boolean/ Arithmetic Operation	Condition Codes					
		Op	~	#	Op	~	#	Op	~	#	Op	~	#	Op	~	#		5	4	3	2	1	0
																		H	I	N	Z	V	C
Compare Index Register	CPX	8C	4	3	9C	5	2	AC	6	2	BC	6	3				X − M:M + 1	•	•	↕	↕	↕	↕
Decrement Index Register	DEX													09	3	1	X − 1 → X	•	•	•	↕	•	•
Decrement Stack Pointer	DES													34	3	1	SP − 1 → SP	•	•	•	•	•	•
Increment Index Register	INX													08	3	1	X + 1 → X	•	•	•	↕	•	•
Increment Stack Pointer	INS													31	3	1	1 SP + 1 → SP	•	•	•	•	•	•
Load Index Register	LDX	CE	3	3	DE	4	2	EE	5	2	FE	5	3				M → X_H, (M + 1) → X_L	•	•	↕	↕	R	•
Load Stack Pointer	LDS	8E	3	3	9E	4	2	AE	5	2	BE	5	3				M → SP_H, (M + 1) → SP_L	•	•	↕	↕	R	•
Store Index Register	STX				DF	4	2	EF	5	2	FF	5	3				X_H → M, X_L → (M + 1)	•	•	↕	↕	R	•
Store Stack Pointer	STS				9F	4	2	AF	5	2	BF	5	3				SP_H → M, SP_L → (M + 1)	•	•	↕	↕	R	•
Index Reg → Stack Pointer	TXS													35	3	1	X − 1 → SP	•	•	•	•	•	•
Stack Pntr → Index Register	TSX													30	3	1	SP + 1 → X	•	•	•	•	•	•
Add	ABX													3A	3	1	B + X → X	•	•	•	•	•	•
Push Data	PSHX													3C	4	1	X_L → M_{SP}, SP − 1 → SP X_H → M_{SP}, SP − 1 → SP	•	•	•	•	•	•
Pull Data	PULX													38	5	1	SP + 1 → SP, M_{SP} → X_H SP + 1 → SP, M_{SP} → X_L	•	•	•	•	•	•

3

TABLE 10 — ACCUMULATOR AND MEMORY INSTRUCTIONS (Sheet 1 of 2)

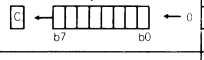
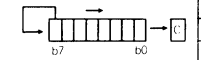
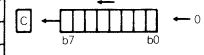
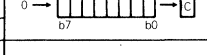
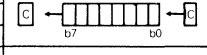
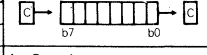
Accumulator and Memory Operations	MNEM	Immed			Direct			Index			Extend			Inher			Boolean Expression	Condition Codes					
		Op	~	#	Op	~	#	Op	~	#	Op	~	#	Op	~	#		5	4	3	2	1	0
																		H	I	N	Z	V	C
Add Accumulators	ABA													1B	2	1	A + B → A	↕	•	↕	↕	↕	↕
Add B to X	ABX													3A	3	1	00:B + X → X	•	•	•	•	•	•
Add with Carry	ADCA	89	2	2	99	3	2	A9	4	2	B9	4	3				A + M + C → A	↕	•	↕	↕	↕	↕
	ADCB	C9	2	2	D9	3	2	E9	4	2	F9	4	3				B + M + C → B	↕	•	↕	↕	↕	↕
Add	ADDA	8B	2	2	9B	3	2	AB	4	2	BB	4	3				A + M → A	↕	•	↕	↕	↕	↕
	ADDB	CB	2	2	DB	3	2	EB	4	2	FB	4	3				B + M → A	↕	•	↕	↕	↕	↕
Add Double	ADDD	C3	4	3	D3	5	2	E3	6	2	F3	6	3				D + M:M + 1 → D	•	•	↕	↕	↕	↕
And	ANDA	84	2	2	94	3	2	A4	4	2	B4	4	3				A•M → A	•	•	↕	↕	R	•
	ANDB	C4	2	2	D4	3	2	E4	4	2	F4	4	3				B•M → B	•	•	↕	↕	R	•
Shift Left, Arithmetic	ASL							68	6	2	78	6	3					•	•	↕	↕	↕	↕
	ASLA													48	2	1		•	•	↕	↕	↕	↕
	ASLB													58	2	1		•	•	↕	↕	↕	↕
Shift Left Double	ASLD													05	3	1		•	•	↕	↕	↕	↕
Shift Right, Arithmetic	ASR							67	6	2	77	6	3					•	•	↕	↕	↕	↕
	ASRA													47	2	1		•	•	↕	↕	↕	↕
	ASRB													57	2	1		•	•	↕	↕	↕	↕
Bit Test	BITA	85	2	2	95	3	2	A5	4	2	B5	4	3				A•M	•	•	↕	↕	R	•
	BITB	C5	2	2	D5	3	2	E5	4	2	F5	4	3				B•M	•	•	↕	↕	R	•
Compare Accumulators	CBA													11	2	1	A − B	•	•	↕	↕	↕	↕
Clear	CLR							6F	6	2	7F	6	3				00 → M	•	•	R	S	R	R
	CLRA													4F	2	1	00 → A	•	•	R	S	R	R
	CLRB													5F	2	1	00 → B	•	•	R	S	R	R
Compare	CMPA	81	2	2	91	3	2	A1	4	2	B1	4	3				A − M	•	•	↕	↕	↕	↕
	CMPB	C1	2	2	D1	3	2	E1	4	2	F1	4	3				B − M	•	•	↕	↕	↕	↕
1's Complement	COM							63	6	2	73	6	3				M → M	•	•	↕	↕	R	S
	COMA													43	2	1	A → A	•	•	↕	↕	R	S
	COMB													53	2	1	B → B	•	•	↕	↕	R	S

TABLE 10 — ACCUMULATOR AND MEMORY INSTRUCTIONS (Sheet 2 of 2)

Accumulator and Memory Operations		Immed			Direct			Index			Extend			Inher			Boolean Expression	Condition Codes					
																		5	4	3	2	1	0
	MNEM	Op	~	#	Op	~	#	Op	~	#	Op	~	#	Op	~	#		H	I	N	Z	V	C
Decimal Adjust, A	DAA													19	2	1	Adj binary sum to BCD	•	•	↕	↕	↕	↕
Decrement	DEC							6A	6	2	7A	6	3				M – 1 → M	•	•	↕	↕	↕	•
	DECA													4A	2	1	A – 1 → A	•	•	↕	↕	↕	•
	DECB													5A	2	1	B – 1 → B	•	•	↕	↕	↕	•
Exclusive OR	EORA	88	2	2	98	3	2	A8	4	2	B8	4	3				A ⊕ M → A	•	•	↕	↕	R	•
	EORB	C8	2	2	D8	3	2	E8	4	2	F8	4	3				B ⊕ M → B	•	•	↕	↕	R	•
Increment	INC							6C	6	2	7C	6	3				M + 1 → M	•	•	↕	↕	↕	•
	INCA													4C	2	1	A + 1 → A	•	•	↕	↕	↕	•
	INCB													5C	2	1	B + 1 → B	•	•	↕	↕	↕	•
Load Accumulators	LDAA	86	2	2	96	3	2	A6	4	2	B6	4	3				M → A	•	•	↕	↕	R	•
	LDAB	C6	2	2	D6	3	2	E6	4	2	F6	4	3				M → B	•	•	↕	↕	R	•
Load Double	LDD	CC	3	3	DC	4	2	EC	5	2	FC	5	3				M:M + 1 → D	•	•	↕	↕	R	•
Logical Shift, Left	LSL							68	6	2	78	6	3					•	•	↕	↕	↕	↕
	LSLA													48	2	1		•	•	↕	↕	↕	↕
	LSLB													58	2	1		•	•	↕	↕	↕	↕
	LSLD													05	3	2		•	•	↕	↕	↕	↕
Shift Right, Logical	LSR							64	6	2	74	6	3					•	•	R	↕	↕	↕
	LSRA													44	2	1		•	•	R	↕	↕	↕
	LSRB													54	2	1		•	•	R	↕	↕	↕
	LSRD													04	3	1		•	•	R	↕	↕	↕
Multiply	MUL													3D	10	1	A × B → D	•	•	•	•	•	↕
2's Complement (Negate)	NEG							60	6	2	70	6	3				00 – M → M	•	•	↕	↕	↕	↕
	NEGA													40	2	1	00 – A → A	•	•	↕	↕	↕	↕
	NEGB													50	2	1	00 – B → B	•	•	↕	↕	↕	↕
No Operation	NOP													01	2	1	PC + 1 → PC	•	•	•	•	•	•
Inclusive OR	ORAA	8A	2	2	9A	3	2	AA	4	2	BA	4	3				A + M → A	•	•	↕	↕	R	•
	ORAB	CA	2	2	DA	3	2	EA	4	2	FA	4	3				B + M → B	•	•	↕	↕	R	•
Push Data	PSHA													36	3	1	A → Stack	•	•	•	•	•	•
	PSHB													37	3	1	B → Stack	•	•	•	•	•	•
Pull Data	PULA													32	4	1	Stack → A	•	•	•	•	•	•
	PULB													33	4	1	Stack → B	•	•	•	•	•	•
Rotate Left	ROL							69	6	2	79	6	3					•	•	↕	↕	↕	↕
	ROLA													49	2	1		•	•	↕	↕	↕	↕
	ROLB													59	2	1		•	•	↕	↕	↕	↕
Rotate Right	ROR							66	6	2	76	6	3					•	•	↕	↕	↕	↕
	RORA													46	2	1		•	•	↕	↕	↕	↕
	RORB													56	2	1		•	•	↕	↕	↕	↕
Subtract Accumulator	SBA													10	2	1	A – B → A	•	•	↕	↕	↕	↕
Subtract with Carry	SBCA	82	2	2	92	3	2	A2	4	2	B2	4	3				A – M – C → A	•	•	↕	↕	↕	↕
	SBCB	C2	2	2	D2	3	2	E2	4	2	F2	4	3				B – M – C → B	•	•	↕	↕	↕	↕
Store Accumulators	STAA				97	3	2	A7	4	2	B7	4	3				A → M	•	•	↕	↕	R	•
	STAB				D7	3	2	E7	4	2	F7	4	3				B → M	•	•	↕	↕	R	•
	STD				DD	4	2	ED	5	2	FD	5	3				D → M:M + 1	•	•	↕	↕	R	•
Subtract	SUBA	80	2	2	90	3	2	A0	4	2	B0	4	3				A – M → A	•	•	↕	↕	↕	↕
	SUBB	C0	2	2	D0	3	2	E0	4	2	F0	4	3				B – M → B	•	•	↕	↕	↕	↕
Subtract Double	SUBD	83	4	3	93	5	2	A3	6	2	B3	6	3				D – M:M + 1 → D	•	•	↕	↕	↕	↕
Transfer Accumulator	TAB													16	2	1	A → B	•	•	↕	↕	R	•
	TBA													17	2	1	B → A	•	•	↕	↕	R	•
Test, Zero or Minus	TST							6D	6	2	7D	6	3				M – 00	•	•	↕	↕	R	R
	TSTA													4D	2	1	A – 00	•	•	↕	↕	R	R
	TSTB													5D	2	1	B – 00	•	•	↕	↕	R	R

The condition code register notes are listed after Table 12.

TABLE 11 — JUMP AND BRANCH INSTRUCTIONS

Operations	MNEM	Direct			Relative			Index			Extend			Inherent			Branch Test	Condition Code Reg.					
		Op	~	#	Op	~	#	Op	~	#	Op	~	#	Op	~	#		5 H	4 I	3 N	2 Z	1 V	0 C
Branch Always	BRA				20	3	2										None	•	•	•	•	•	•
Branch Never	BRN				21	3	2										None	•	•	•	•	•	•
Branch If Carry Clear	BCC				24	3	2										C = 0	•	•	•	•	•	•
Branch If Carry Set	BCS				25	3	2										C = 1	•	•	•	•	•	•
Branch If = Zero	BEQ				27	3	2										Z = 1	•	•	•	•	•	•
Branch If ≥ Zero	BGE				2C	3	2										N ⊕ V = 0	•	•	•	•	•	•
Branch if > Zero	BGT				2E	3	2										Z + (N ⊕ V) = 0	•	•	•	•	•	•
Branch If Higher	BHI				22	3	2										C + Z = 0	•	•	•	•	•	•
Branch if Higher or Same	BHS				24	3	2										C = 0	•	•	•	•	•	•
Branch If ≤ Zero	BLE				2F	3	2										Z + (N ⊕ V) = 1	•	•	•	•	•	•
Branch If Carry Set	BLO				25	3	2										C = 1	•	•	•	•	•	•
Branch If Lower Or Same	BLS				23	3	2										C + Z = 1	•	•	•	•	•	•
Branch If < Zero	BLT				2D	3	2										N ⊕ V = 1	•	•	•	•	•	•
Branch If Minus	BMI				2B	3	2										N = 1	•	•	•	•	•	•
Branch If Not Equal Zero	BNE				26	3	2										Z = 0	•	•	•	•	•	•
Branch If Overflow Clear	BVC				28	3	2										V = 0	•	•	•	•	•	•
Branch If Overflow Set	BVS				29	3	2										V = 1	•	•	•	•	•	•
Branch If Plus	BPL				2A	3	2										N = 0	•	•	•	•	•	•
Branch To Subroutine	BSR				8D	6	2											•	•	•	•	•	•
Jump	JMP							6E	3	2	7E	3	3				See Special Operations-Figure 25	•	•	•	•	•	•
Jump To Subroutine	JSR	9D	5	2				AD	6	2	BD	6	3					•	•	•	•	•	•
No Operation	NOP													01	2	1		•	•	•	•	•	•
Return From Interrupt	RTI													3B	10	1		↕	↕	↕	↕	↕	↕
Return From Subroutine	RTS													39	5	1	See Special Operations-Figure 25	•	•	•	•	•	•
Software Interrupt	SWI													3F	12	1		•	S	•	•	•	•
Wait For Interrupt	WAI													3E	9	1		•	•	•	•	•	•

TABLE 12 — CONDITION CODE REGISTER MANIPULATION INSTRUCTIONS

Operations	Inherent				Boolean Operation	Condition Code Register					
	MNEM	Op	~	#		5 H	4 I	3 N	2 Z	1 V	0 C
Clear Carry	CLC	0C	2	1	0 → C	•	•	•	•	•	R
Clear Interrupt Mask	CLI	0E	2	1	0 → I	•	R	•	•	•	•
Clear Overflow	CLV	0A	2	1	0 → V	•	•	•	•	R	•
Set Carry	SEC	0D	2	1	1 → C	•	•	•	•	•	S
Set Interrupt Mask	SEI	0F	2	1	1 → I	•	S	•	•	•	•
Set Overflow	SEV	0B	2	1	1 → V	•	•	•	•	S	•
Accumulator A → CCR	TAP	06	2	1	A → CCR	↕	↕	↕	↕	↕	↕
CCR → Accumulator A	TPA	07	2	1	CCR → A	•	•	•	•	•	•

LEGEND

- Op Operation Code (Hexadecimal)
- ~ Number of MPU Cycles
- M_{SP} Contents of memory location pointed to by Stack Pointer
- # Number of Program Bytes
- + Arithmetic Plus
- − Arithmetic Minus
- • Boolean AND
- X Arithmetic Multiply
- + Boolean Inclusive OR
- ⊕ Boolean Exclusive OR
- M Complement of M
- → Transfer Into
- 0 Bit = Zero
- 00 Byte = Zero

CONDITION CODE SYMBOLS

- H Half-carry from bit 3
- I Interrupt mask
- N Negative (sign bit)
- Z Zero (byte)
- V Overflow, 2's complement
- C Carry/Borrow from MSB
- R Reset Always
- S Set Always
- ↕ Affected
- • Not Affected

TABLE 13 — INSTRUCTION EXECUTION TIMES IN E CYCLES

	ADDRESSING MODE					
	Immediate	Direct	Extended	Indexed	Inherent	Relative
ABA	●	●	●	●	2	●
ABX	●	●	●	●	3	●
ADC	2	3	4	4	●	●
ADD	2	3	4	4	●	●
ADDD	4	5	6	6	●	●
AND	2	3	4	4	●	●
ASL	●	●	6	6	2	●
ASLD	●	●	●	●	3	●
ASR	●	●	6	6	2	●
BCC	●	●	●	●	●	3
BCS	●	●	●	●	●	3
BEQ	●	●	●	●	●	3
BGE	●	●	●	●	●	3
BGT	●	●	●	●	●	3
BHI	●	●	●	●	●	3
BHS	●	●	●	●	●	3
BIT	2	3	4	4	●	●
BLE	●	●	●	●	●	3
BLO	●	●	●	●	●	3
BLS	●	●	●	●	●	3
BLT	●	●	●	●	●	3
BMI	●	●	●	●	●	3
BNE	●	●	●	●	●	3
BPL	●	●	●	●	●	3
BRA	●	●	●	●	●	3
BRN	●	●	●	●	●	3
BSR	●	●	●	●	●	6
BVC	●	●	●	●	●	3
BVS	●	●	●	●	●	3
CBA	●	●	●	●	2	●
CLC	●	●	●	●	2	●
CLI	●	●	●	●	2	●
CLR	●	●	6	6	2	●
CLV	●	●	●	●	2	●
CMP	2	3	4	4	●	●
COM	●	●	6	6	2	●
CPX	4	5	6	6	●	●
DAA	●	●	●	●	2	●
DEC	●	●	6	6	2	●
DES	●	●	●	●	3	●
DEX	●	●	●	●	3	●
EOR	2	3	4	4	●	●
INC	●	●	6	6	●	●
INS	●	●	●	●	3	●

	ADDRESSING MODE					
	Immediate	Direct	Extended	Indexed	Inherent	Relative
INX	●	●	●	●	3	●
JMP	●	●	3	3	●	●
JSR	●	5	6	6	●	●
LDA	2	3	4	4	●	●
LDD	3	4	5	5	●	●
LDS	3	4	5	5	●	●
LDX	3	4	5	5	●	●
LSL	●	●	6	6	2	●
LSLD	●	●	●	●	3	●
LSR	●	●	6	6	2	●
LSRD	●	●	●	●	3	●
MUL	●	●	●	●	10	●
NEG	●	●	6	6	2	●
NOP	●	●	●	●	2	●
ORA	2	3	4	4	●	●
PSH	●	●	●	●	3	●
PSHX	●	●	●	●	4	●
PUL	●	●	●	●	4	●
PULX	●	●	●	●	5	●
ROL	●	●	6	6	2	●
ROR	●	●	6	6	2	●
RTI	●	●	●	●	10	●
RTS	●	●	●	●	5	●
SBA	●	●	●	●	2	●
SBC	2	3	4	4	●	●
SEC	●	●	●	●	2	●
SEI	●	●	●	●	2	●
SEV	●	●	●	●	2	●
STA	●	3	4	4	●	●
STD	●	4	5	5	●	●
STS	●	4	5	5	●	●
STX	●	4	5	5	●	●
SUB	2	3	4	4	●	●
SUBD	4	5	6	6	●	●
SWI	●	●	●	●	12	●
TAB	●	●	●	●	2	●
TAP	●	●	●	●	2	●
TBA	●	●	●	●	2	●
TPA	●	●	●	●	2	●
TST	●	●	6	6	2	●
TSX	●	●	●	●	3	●
TXS	●	●	●	●	3	●
WAI	●	●	●	●	9	●

SUMMARY OF CYCLE-BY-CYCLE OPERATION

Table 14 provides a detailed description of the information present on the address bus, data bus, and the read/write (R/$\overline{W}$) line during each cycle of each instruction.

The information is useful in comparing actual with expected results during debug of both software and hardware as the program is executed. The information is categorized in groups according to addressing mode and number of cycles per instruction. In general, instructions with the same addressing mode and number of cycles execute in the same manner. Exceptions are indicated in the table.

Note that during MPU reads of internal locations, the resultant value will not appear on the external data bus except in mode 0. "High order" byte refers to the most-significant byte of a 16-bit value.

TABLE 14 — CYCLE-BY-CYCLE OPERATION (Sheet 1 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
IMMEDIATE					
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	2	1 2	Opcode Address Opcode Address + 1	1 1	Opcode Operand Data
LDS LDX LDD	3	1 2 3	Opcode Address Opcode Address + 1 Opcode Address + 2	1 1 1	Opcode Operand Data (High Order Byte) Operand Data (Low Order Byte)
CPX SUBD ADDD	4	1 2 3 4	Opcode Address Opcode Address + 1 Opcode Address + 2 Address Bus FFFF	1 1 1 1	Opcode Operand Data (High Order Byte) Operand Data (Low Order Byte) Low Byte of Restart Vector
DIRECT					
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	3	1 2 3	Opcode Address Opcode Address + 1 Address of Operand	1 1 1	Opcode Address of Operand Operand Data
STA	3	1 2 3	Opcode Address Opcode Address + 1 Destination Address	1 1 0	Opcode Destination Address Data from Accumulator
LDS LDX LDD	4	1 2 3 4	Opcode Address Opcode Address + 1 Address of Operand Operand Address + 1	1 1 1 1	Opcode Address of Operand Operand Data (High Order Byte) Operand Data (Low Order Byte)
STS STX STD	4	1 2 3 4	Opcode Address Opcode Address + 1 Address of Operand Address of Operand + 1	1 1 0 0	Opcode Address of Operand Register Data (High Order Byte) Register Data (Low Order Byte)
CPX SUBD ADDD	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Operand Address Operand Address + 1 Address Bus FFFF	1 1 1 1 1	Opcode Address of Operand Operand Data (High Order Byte) Operand Data (Low Order Byte) Low Byte of Restart Vector
JSR	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Subroutine Address Stack Pointer Stack Pointer − 1	1 1 1 0 0	Opcode Irrelevant Data First Subroutine Opcode Return Address (Low Order Byte) Return Address (High Order Byte)

TABLE 14 — CYCLE-BY-CYCLE OPERATION (Sheet 2 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
EXTENDED					
JMP	3	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Jump Address (High Order Byte)
		3	Opcode Address + 2	1	Jump Address (Low Order Byte)
ADC EOR	4	1	Opcode Address	1	Opcode
ADD LDA		2	Opcode Address + 1	1	Address of Operand
AND ORA		3	Opcode Address + 2	1	Address of Operand (Low Order Byte)
BIT SBC		4	Address of Operand	1	Operand Data
CMP SUB					
STA	4	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Destination Address (High Order Byte)
		3	Opcode Address + 2	1	Destination Address (Low Order Byte)
		4	Operand Destination Address	0	Data from Accumulator
LDS	5	1	Opcode Address	1	Opcode
LDX		2	Opcode Address + 1	1	Address of Operand (High Order Byte)
LDD		3	Opcode Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	1	Operand Data (High Order Byte)
		5	Address of Operand + 1	1	Operand Data (Low Order Byte)
STS	5	1	Opcode Address	1	Opcode
STX		2	Opcode Address + 1	1	Address of Operand (High Order Byte)
STD		3	Opcode Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	0	Operand Data (High Order Byte)
		5	Address of Operand + 1	0	Operand Data (Low Order Byte)
ASL LSR	6	1	Opcode Address	1	Opcode
ASR NEG		2	Opcode Address + 1	1	Address of Operand (High Order Byte)
CLR ROL		3	Opcode Address + 2	1	Address of Operand (Low Order Byte)
COM ROR		4	Address of Operand	1	Current Operand Data
DEC TST*		5	Address Bus FFFF	1	Low Byte of Restart Vector
INC		6	Address of Operand	0	New Operand Data
CPX	6	1	Opcode Address	1	Opcode
SUBD		2	Opcode Address + 1	1	Operand Address (High Order Byte)
ADDD		3	Opcode Address + 2	1	Operand Address (Low Order Byte)
		4	Operand Address	1	Operand Data (High Order Byte)
		5	Operand Address + 1	1	Operand Data (Low Order Byte)
		6	Address Bus FFFF	1	Low Byte of Restart Vector
JSR	6	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Address of Subroutine (High Order Byte)
		3	Opcode Address + 2	1	Address of Subroutine (Low Order Byte)
		4	Subroutine Starting Address	1	Opcode of Next Instruction
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer − 1	0	Return Address (High Order Byte)

*TST does not perform the write cycle during the sixth cycle. The sixth cycle is another address bus = $FFFF.

TABLE 14 — CYCLE-BY-CYCLE OPERATION (Sheet 3 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
INDEXED					
JMP	3	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
ADC EOR	4	1	Opcode Address	1	Opcode
ADD LDA		2	Opcode Address + 1	1	Offset
AND ORA		3	Address Bus FFFF	1	Low Byte of Restart Vector
BIT SBC		4	Index Register Plus Offset	1	Operand Data
CMP SUB					
STA	4	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	0	Operand Data
LDS	5	1	Opcode Address	1	Opcode
LDX		2	Opcode Address + 1	1	Offset
LDD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	1	Operand Data (High Order Byte)
		5	Index Register Plus Offset + 1	1	Operand Data (Low Order Byte)
STS	5	1	Opcode Address	1	Opcode
STX		2	Opcode Address + 1	1	Offset
STD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	0	Operand Data (High Order Byte)
		5	Index Register Plus Offset + 1	0	Operand Data (Low Order Byte)
ASL LSR	6	1	Opcode Address	1	Opcode
ASR NEG		2	Opcode Address + 1	1	Offset
CLR ROL		3	Address Bus FFFF	1	Low Byte of Restart Vector
COM ROR		4	Index Register Plus Offset	1	Current Operand Data
DEC TST*		5	Address Bus FFFF	1	Low Byte of Restart Vector
INC		6	Index Register Plus Offset	0	New Operand Data
CPX	6	1	Opcode Address	1	Opcode
SUBD		2	Opcode Address + 1	1	Offset
ADDD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register + Offset	1	Operand Data (High Order Byte)
		5	Index Register + Offset + 1	1	Operand Data (Low Order Byte)
		6	Address Bus FFFF		Low Byte of Restart Vector
JSR	6	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register + Offset	1	First Subroutine Opcode
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer − 1	0	Return Address (High Order Byte)

*TST does not perform the write cycle during the sixth cycle. The sixth cycle is another address bus = $FFFF.

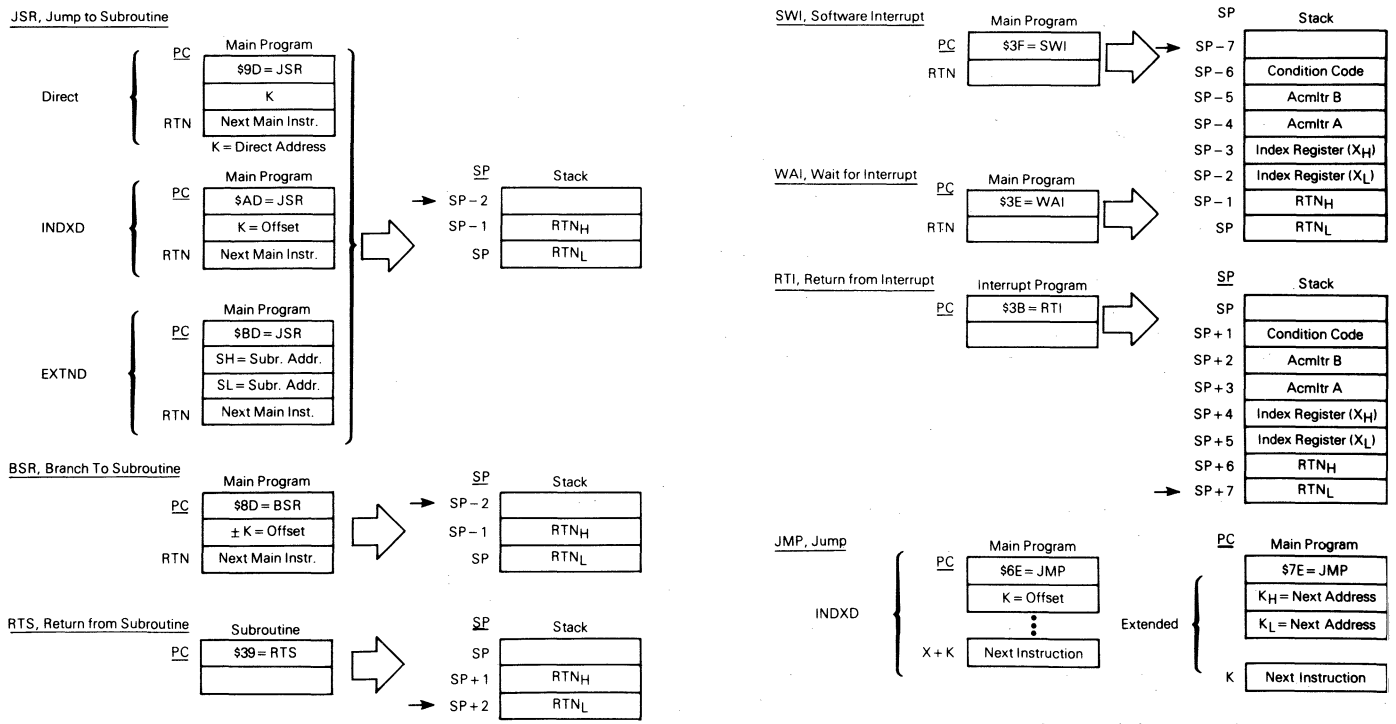
TABLE 14 — CYCLE-BY-CYCLE OPERATION (Sheet 4 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
INHERENT					
ABA DAA SEC ASL DEC SEI ASR INC SEV CBA LSR TAB CLC NEG TAP CLI NOP TBA CLR ROL TPA CLV ROR TST COM SBA	2	1 2	Opcode Address Opcode Address + 1	1 1	Opcode Opcode of Next Instruction
ABX	3	1 2 3	Opcode Address Opcode Address + 1 Address Bus FFFF	1 1 1	Opcode Irrelevant Data Low Byte of Restart Vector
ASLD LSRD	3	1 2 3	Opcode Address Opcode Address + 1 Address Bus FFFF	1 1 1	Opcode Irrelevant Data Low Byte of Restart Vector
DES INS	3	1 2 3	Opcode Address Opcode Address + 1 Previous Stack Pointer Contents	1 1 1	Opcode Opcode of Next Instruction Irrelevant Data
INX DEX	3	1 2 3	Opcode Address Opcode Address + 1 Address Bus FFFF	1 1 1	Opcode Opcode of Next Instruction Low Byte of Restart Vector
PSHA PSHB	3	1 2 3	Opcode Address Opcode Address + 1 Stack Pointer	1 1 0	Opcode Opcode of Next Instruction Accumulator Data
TSX	3	1 2 3	Opcode Address Opcode Address + 1 Stack Pointer	1 1 1	Opcode Opcode of Next Instruction Irrelevant Data
TXS	3	1 2 3	Opcode Address Opcode Address + 1 Address Bus FFFF	1 1 1	Opcode Opcode of Next Instruction Low Byte of Restart Vector
PULA PULB	4	1 2 3 4	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer + 1	1 1 1 1	Opcode Opcode of Next Instruction Irrelevant Data Operand Data from Stack
PSHX	4	1 2 3 4	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer – 1	1 1 0 0	Opcode Irrelevant Data Index Register (Low Order Byte) Index Register (High Order Byte)
PULX	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer + 1 Stack Pointer + 2	1 1 1 1 1	Opcode Irrelevant Data Irrelevant Data Index Register (High Order Byte) Index Register (Low Order Byte)
RTS	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer + 1 Stack Pointer + 2	1 1 1 1 1	Opcode Irrelevant Data Irrelevant Data Address of Next Instruction (High Order Byte) Address of Next Instruction (Low Order Byte)
WAI	9	1 2 3 4 5 6 7 8 9	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer – 1 Stack Pointer – 2 Stack Pointer – 3 Stack Pointer – 4 Stack Pointer – 5 Stack Pointer – 6	1 1 0 0 0 0 0 0 0	Opcode Opcode of Next Instruction Return Address (Low Order Byte) Return Address (High Order Byte) Index Register (Low Order Byte) Index Register (High Order Byte) Contents of Accumulator A Contents of Accumulator B Contents of Condition Code Register

TABLE 14 — CYCLE-BY-CYCLE OPERATION (Sheet 5 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
INHERENT					
MUL	10	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Irrelevant Data
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Address Bus FFFF	1	Low Byte of Restart Vector
		5	Address Bus FFFF	1	Low Byte of Restart Vector
		6	Address Bus FFFF	1	Low Byte of Restart Vector
		7	Address Bus FFFF	1	Low Byte of Restart Vector
		8	Address Bus FFFF	1	Low Byte of Restart Vector
		9	Address Bus FFFF	1	Low Byte of Restart Vector
		10	Address Bus FFFF	1	Low Byte of Restart Vector
RTI	10	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Irrelevant Data
		3	Stack Pointer	1	Irrelevant Data
		4	Stack Pointer + 1	1	Contents of Condition Code Register from Stack
		5	Stack Pointer + 2	1	Contents of Accumulator B from Stack
		6	Stack Pointer + 3	1	Contents of Accumulator A from Stack
		7	Stack Pointer + 4	1	Index Register from Stack (High Order Byte)
		8	Stack Pointer + 5	1	Index Register from Stack (Low Order Byte)
		9	Stack Pointer + 6	1	Next Instruction Address from Stack (High Order Byte)
		10	Stack Pointer + 7	1	Next Instruction Address from Stack (Low Order Byte)
SWI	12	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Irrelevant Data
		3	Stack Pointer	0	Return Address (Low Order Byte)
		4	Stack Pointer − 1	0	Return Address (High Order Byte)
		5	Stack Pointer − 2	0	Index Register (Low Order Byte)
		6	Stack Pointer − 3	0	Index Register (High Order Byte)
		7	Stack Pointer − 4	0	Contents of Accumulator A
		8	Stack Pointer − 5	0	Contents of Accumulator B
		9	Stack Pointer − 6	0	Contents of Condition Code Register
		10	Stack Pointer − 7	1	Irrelevant Data
		11	Vector Address FFFA (Hex)	1	Address of Subroutine (High Order Byte)
		12	Vector Address FFFB (Hex)	1	Address of Subroutine (Low Order Byte)
RELATIVE					
BCC BHT BNE BLO BCS BLE BPL BHS BEQ BLS BRA BRN BGE BLT BVC BGT BMI BVS	3	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Branch Offset
		3	Address Buss FFFF	1	Low Byte of Restart Vector
BSR	6	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Branch Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Subroutine Starting Address	1	Opcode of Next Instruction
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer − 1	0	Return Address (High Order Byte)

FIGURE 25 — SPECIAL OPERATIONS



Legend:
RTN = Address of next instruction in Main Program to be executed upon return from subroutine
RTN$_H$ = Most significant byte of Return Address
RTN$_L$ = Least significant byte of Return Address
→ = Stack Pointer After Execution
K = 8-bit Unsigned Value

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola using the following media:

MDOS, disk file
PC-DOS disk file (360K)
EPROM(s) 2516, 2716, MC68701

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, sales person, or a Motorola representative.

FLEXIBLE DISKS

Several types of flexible disks (MDOS™ or PC-DOS disk file) may be submitted for pattern generation. They should be programmed with the customer's program, using positive logic sense for address and data. The diskette should be clearly labeled with the customer's name, date, project or product name, and the filename containing the pattern.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to expedite the process in case of any difficulty with the pattern file.

MDOS Disk File

MDOS is Motorola's Disk Operating System available on the EXORciser® development system. The disk media submitted must be a single-sided, single-density, 8-inch MDOS compatible floppy diskette. The diskette must contain the minimum set of MDOS system files in addition to the pattern file.

The .LO output of the M6801 cross assembler should be furnished. In addition, the file must be produced using the ROLLOUT command, so that it contains the absolute image of the M6801 memory. It is necessary to include the entire memory image of both program and data space. All unused bytes, including those in the user space, must be set in logic zero.

PC-DOS Disk File

PC-DOS is the IBM® Personal Computer Disk Operating System. Disk media submitted must be standard density (360K), double-sided 5-1/4 inch compatible floppy diskette. The diskette must contain the object file code in Motorola's S-record format. The S-record format is a character-based object file format generated by M6801 cross assemblers and linkers on IBM PC style machines.

EPROMS

A single 2K EPROM is necessary to contain the entire MC6801 program. The EPROM is programmed with the customer program using positive logic sense for address and data. All unused bytes, including the user's space, must be set to zero.

If the MC6801 MCU ROM pattern is submitted on a single 2516 or 2716 type EPROM, memory map addressing is one-for-one. The data space ROM runs from EPROM address $000 to $7FF. If an MC68701 is used, the ROM map runs from $F800 to $FFFF.

For shipment to Motorola, EPROMs should be placed in a conductive IC carrier and packed securely. Styrofoam is not acceptable for shipment.

Verification Media

All original pattern media, EPROMs or floppy disks, are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked and the verification form completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for the creation of the customer mask. To aid in the verification process, Motorola will program customer supplied blank EPROM(s) or DOS disks from the data file used to create the custom mask.

ROM Verification Units (RVUs)

Ten MCUs containing the customer's ROM pattern will be sent for program verification. These units will have been made using the custom mask, but are for the purpose of ROM verification only. For expediency, the MCUs are unmarked, packaged in ceramic, and tested with five volts at room temperature. These RVUs are free with the minimum order quantity, but are not production parts. These RVUs are not guaranteed by Motorola Quality Assurance.

Ordering Information

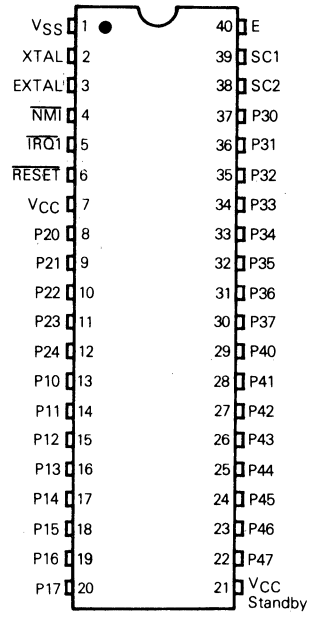
The following table provides generic information pertaining to the package type and temperature for the MC6801/MC6803. This MCU device is available only in the 40-pin dual-in-line (DIP) package in the Cerdip and Plastic packages.

MDOS is a trademark of Motorola Inc.
MS-DOS is a trademark of Microsoft, Inc.
EXORciser is a registered trademark of Motorola Inc.
IBM is a registered trademark of International Business Machines Corporation.

GENERIC INFORMATION

Frequency (MHz)	Temperature (Degrees C)	Cerdip Package (S Suffix)	Plastic Package (P Suffix)
1.0	0 to 70	MC6801S1	MC6801P1
1.0	−40 to +85	MC6801CS1	MC6801CP1
1.25	0 to 70	MC6801S1-1	MC6801P1-1
1.25	−40 to +85	MC6801CS-1	MC6801CP-1
2.0	0 to 70	MC68B01S1	MC68B01P1
1.0	0 to 70	MC6803S	MC6803P
1.0	−40 to +85	MC6803CS	MC6803CP
1.25	0 to 70	MC6803S-1	MC6803P-1
1.25	−40 to +85	MC6803CS-1	MC6803CP-1
2.0	0 to 70	MC68B03S	MC68B03P

PIN ASSIGNMENT



MC6801U4
MC6803U4

Advance Information

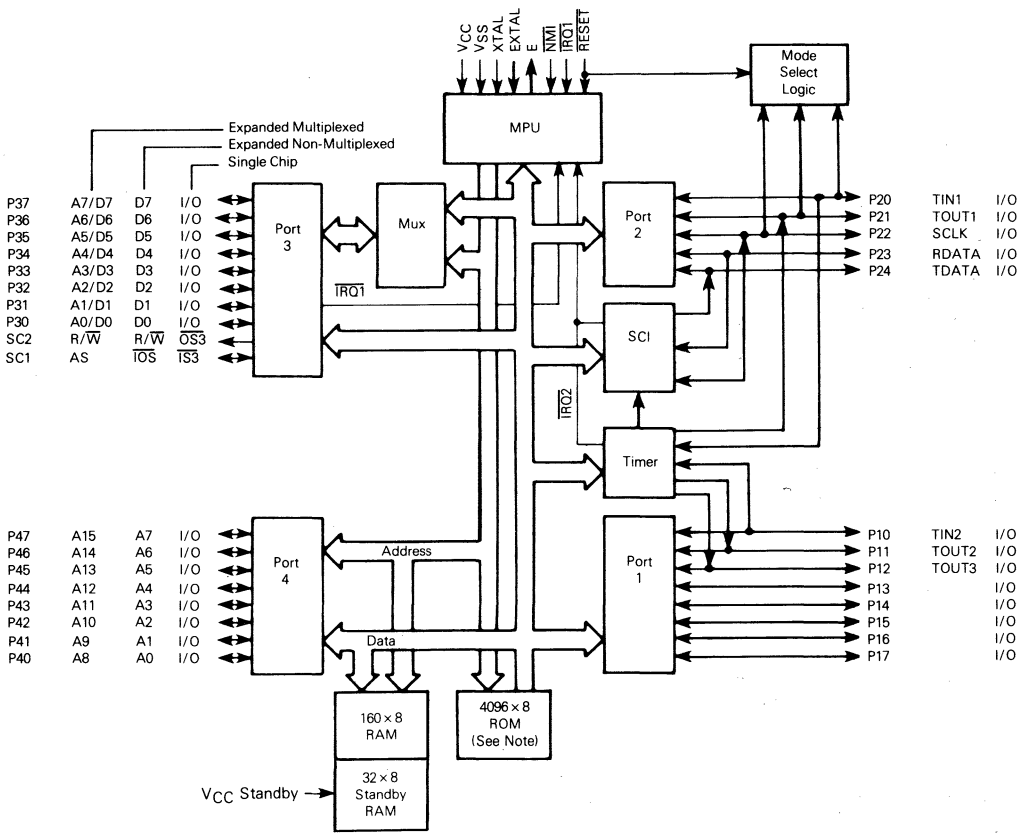
Microcontroller/Microprocessor (MCU/MPU)

The MC6801U4 is an 8-bit single-chip microcontroller unit(MCU) that enhances the capabilities of the MC6801 and significantly enhances the capabilities of the M6800 Family of parts. It includes an MC6801 microprocessor unit (MPU) with direct object-code compatibility and upward object-code compatibility with the MC6800. Execution times of key instructions have been improved over the MC6800, and the new instructions found on the MC6801 are included. The MCU can function as a monolithic microcontroller or can be expanded to a 64K-byte address space. It is TTL compatible and requires one +5-volt power supply. On-chip resources include 4096 bytes of ROM, 192 bytes of RAM, a serial communications interface (SCI), parallel I/O, and a 16-bit six-function programmable timer. The MC6803U4 can be considered an MC6801U4 operating in modes 2 or 3; i.e., those that do not use internal ROM.

- Enhanced MC6800 Instruction Set
- Upward Source and Object Code Compatibility with the MC6800 and MC6801
- Bus Compatibility with the M6800 Family
- 8×8 Multiply Instruction
- Single-Chip or Expanded Operation to 64K-Byte Address Space
- Internal Clock Generator with Divide-by-Four Output
- Serial Communications Interface (SCI)
- 16-Bit Six-Function Programmable Timer
- Three Output Compare Functions
- Two Input Capture Functions
- Counter Alternate Address
- 4096 Bytes of ROM (MC6801U4)
- 192 Bytes of RAM
- 32 Bytes of RAM Retainable During Powerdown
- 29 Parallel I/O and Two Handshake Control Lines
- $\overline{\text{NMI}}$ Inhibited Until Stack Load
- −40°C to 85°C Temperature Range

This document contains information on a new product. Specifications and information herein are subject to change without notice.

MC6801U4 MICROCOMPUTER FAMILY BLOCK DIAGRAM



NOTE: No functioning ROM in MC6803U4.

MC6801U4 MICROCONTROLLER FAMILY BLOCK DIAGRAM

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range MC6801U4, MC6803U4 MC6801U4C, MC6803U4C	T_A	T_L to T_H −0 to 70 −40 to 85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Rating
Thermal Resistance Plastic Ceramic	θ_{JA}	 50 50	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A

CONTROL TIMING ($V_{CC} = 5.0\ V \pm 5\%$, $V_{SS} = 0$)

Characteristic	Symbol	MC6801U4 MC6803U4		MC6801U4-1 MC6803U4-1		Unit
		Min	Max	Min	Max	
Frequency of Operation	f_o	0.5	1.0	0.5	1.25	MHz
Crystal Frequency	f_{XTAL}	2.0	4.0	2.0	5.0	MHz
External Oscillator Frequency	$4\ f_o$	2.0	4.0	2.0	5.0	MHz
Crystal Oscillator Startup Time	t_{rc}	—	100	—	100	ms
Processor Control Setup Time	t_{PCS}	200	—	170	—	ns

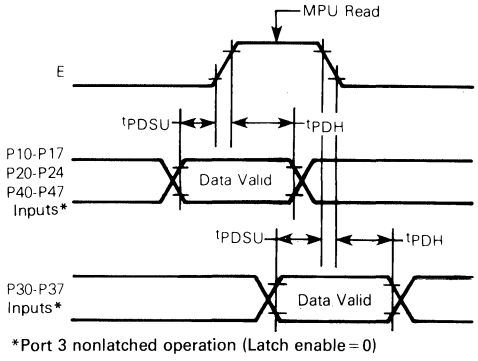
DC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 Vdc ± 5%, V_{SS} = 0, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic		Symbol	MC6801U4, MC6803U4 Min	MC6801U4, MC6803U4 Max	MC6801U4C, MC6803U4C Min	MC6801U4C, MC6803U4C Max	Unit
Input High Voltage	$\overline{RESET}$	V_{IH}	$V_{SS}+4.0$	V_{CC}	$V_{SS}+4.0$	V_{CC}	V
	Other Inputs*		$V_{SS}+2.0$	V_{CC}	$V_{SS}+2.2$	V_{CC}	
Input Low Voltage	All Inputs*	V_{IL}	$V_{SS}-0.3$	$V_{SS}+0.8$	$V_{SS}-0.3$	$V_{SS}+0.8$	V
Input Load Current	Port 4	I_{in}	–	0.5	–	0.8	mA
	SC1		–	0.8	–	1.0	
Input Leakage Current (V_{in} = 0 to 5.5 V)	$\overline{NMI}$, $\overline{IRQ1}$, $\overline{RESET}$	I_{in}	–	2.5	–	5.0	μA
Hi-Z (Off-State) Input Current (V_{in} = 0.5 to 2.4 V)	Port 1, Port 2, Port 3	I_{TSI}	–	10	–	20	μA
Output High Voltage (I_{Load} = −65 μA, V_{CC} = Min)	Port 4, SC1, SC2	V_{OH}	$V_{SS}+2.4$	–	$V_{SS}+2.4$	–	V
(I_{Load} = −100 μA, V_{CC} = Min)	Other Outputs		$V_{SS}+2.4$	–	$V_{SS}+2.4$	–	
Output Low Voltage (I_{Load} = 2.0 mA, V_{CC} = Min)	All Outputs	V_{OL}	–	$V_{SS}+0.5$	–	$V_{SS}+0.6$	V
Darlington Drive Current (V_O = 1.5 V)	Port 1	I_{OH}	1.0	4.0	1.0	5.0	mA
Internal Power Dissipation (Measured at $T_A = T_L$ in Steady-State Operation)***		P_{INT}	–	1200	–	1500	mW
Input Capacitance (V_{in} = 0, T_A = 25°C, f_o = 1.0 MHz)	Port 3, Port 4, SC1	C_{in}	–	12.5	–	12.5	pF
	Other Inputs		–	10.0	–	10.0	
V_{CC} Standby	Powerdown	V_{SBB}	4.0	5.25	4.0	5.25	V
	Powerup	V_{SB}	4.75	5.25	4.75	5.25	
Standby Current	Powerdown	I_{SBB}	–	3.0	–	3.5	mA

*Except mode programming levels; see Figure 16.
**Negotiable to −100 μA (for further information contact the factory).
***For the MC6801U4/MC6803U4 T_L = 0°C and for the MC6801U4C/MC6803U4C T_L = 40°C.

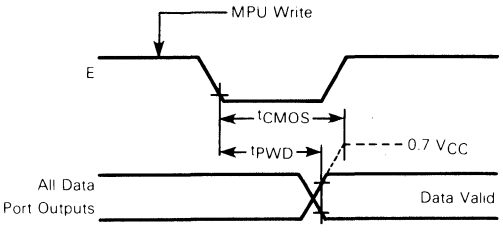
PERIPHERAL PORT TIMING (Refer to Figure 1-4)

Characteristics	Symbol	Min	Typ	Max	Unit
Peripheral Data Setup Time	t_{PDSU}	200	–	–	ns
Peripheral Data Hold Time	t_{PDH}	200	–	–	ns
Delay Time, Enable Positive Transition to $\overline{OS3}$ Negative Transition	t_{OSD1}	–	–	350	ns
Delay Time, Enable Positive Transition to $\overline{OS3}$ Positive Transition	t_{OSD2}	–	–	350	ns
Delay Time, Enable Negative Transition to Peripheral Data Valid Port 1	t_{PWD}	–	–	350	ns
Port 2, 3, 4		–	–	350	
Delay Time, Enable Negative Transition to Peripheral CMOS Data Valid	t_{CMOS}	–	–	2.0	μs
Input Strobe Pulse Width	t_{PWIS}	200	–	–	ns
Input Data Hold Time	t_{IH}	50	–	–	ns
Input Data Setup Time	t_{IS}	20	–	–	ns



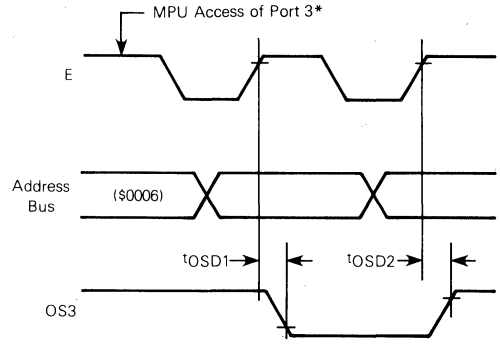
*Port 3 nonlatched operation (Latch enable = 0)

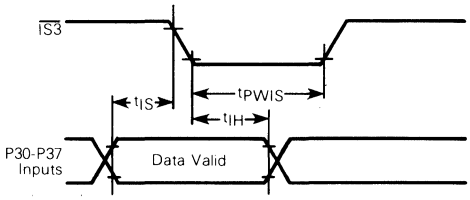
Figure 1. Data Setup and Hold Times (MPU Read)



NOTES:
1. 10 k pullup resistor required for port 2 to reach 0.7 V_{CC}
2. Not applicable to P21
3. Port 4 cannot be pulled above V_{CC}

Figure 2. Data Setup and Hold Times (MPU Write)





*Access matches output strobe select (OSS = 0, a read; OSS = 1, a write)

NOTE: Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted.

Figure 3. Port 3 Output Strobe Timing (MC6801U4 Single-Chip Mode)

Figure 4. Port 3 Latch Timing (MC6801U4 Single-Chip Mode)

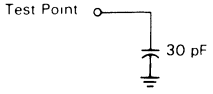
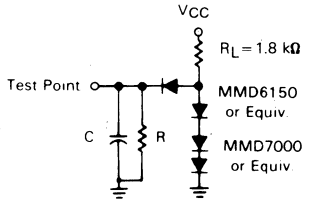


Figure 5. CMOS Load



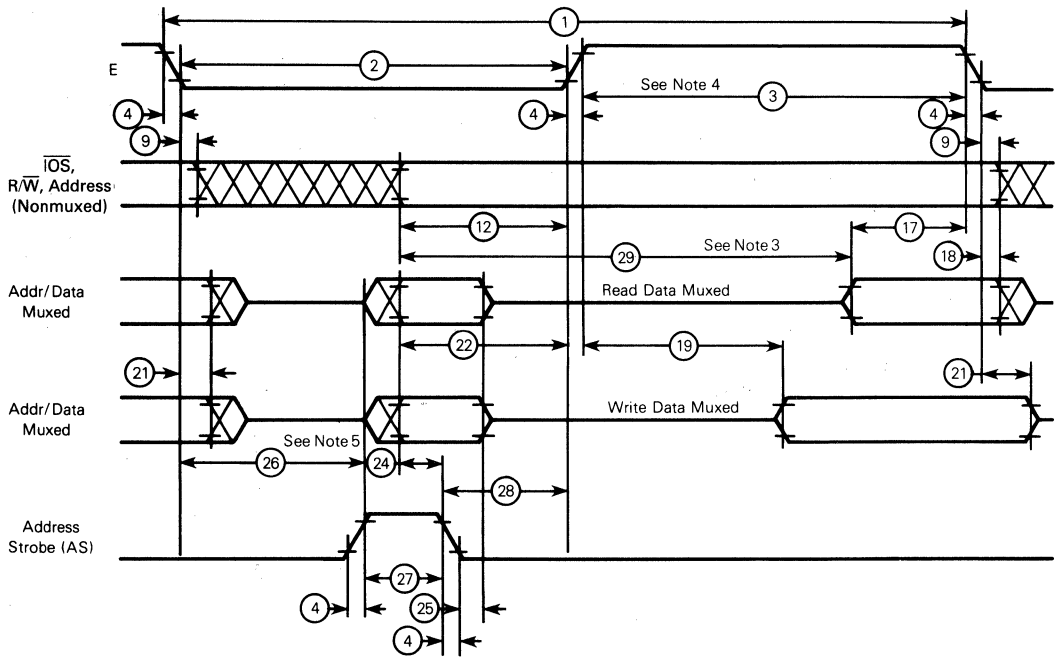
C = 90 pF for P30-P37, P40-P47, E, SC1, SC2
= 30 pF for P10-P17, P20-P24
R = 37 kΩ for P40-P47, SC1, SC2
= 24 kΩ for P10-P17, P20-P24
= 24 kΩ for P30-P37, E

Figure 6. Timing Test Load Ports 1, 2, 3, and 4

BUS TIMING (See Notes 1 and 2, and Figure 7)

Ident. Number	Characteristics	Symbol	MC6801U4 MC6803U4		MC6801U4-1 MC6803U4-1		Unit
			Min	Max	Min	Max	
1	Cycle Time	t_{cyc}	1.0	2.0	0.8	2.0	μs
2	Pulse Width, E Low	PW_{EL}	430	1000	360	1000	ns
3	Pulse Width, E High	PW_{EH}	450	1000	360	1000	ns
4	Clock Rise and Fall Time	t_r, t_f	–	25	–	25	ns
9	Address Hold Time	t_{AH}	20	–	20	–	ns
12	Nonmuxed Address Valid Time to E*	t_{AV}	200	–	150	–	ns
17	Read Data Setup Time	t_{DSR}	80	–	70	–	ns
18	Read Data Hold Time	t_{DHR}	10	–	10	–	ns
19	Write Data Delay Time	t_{DDW}	–	225	–	200	ns
21	Write Data Hold Time	t_{DHW}	20	–	20	–	ns
22	Muxed Address Valid Time to E Rise*	t_{AVM}	160	–	120	–	ns
24	Muxed Address Valid Time to AS Fall*	t_{ASL}	40	–	30	–	ns
25	Muxed Address Hold Time	t_{AHL}	20	–	20	–	ns
26	Delay Time, E to AS Rise*	t_{ASD}	200	–	170	–	ns
27	Pulse Width, AS High*	PW_{ASH}	100	–	80	–	ns
28	Delay Time, AS to E Rise*	t_{ASED}	90	–	70	–	ns
29	Usable Access Time* (See Note 3)	t_{ACC}	555	–	435	–	ns

*At specified cycle time.



NOTES:
1. Voltage levels shown are $V_L \leq 0.5$ V, $V_H \geq 2.4$ V, unless otherwise specified.
2. Measurement points shown are 0.8 V and 2.0 V, unless otherwise specified.
3. Usable access time is computed by 22+3−17+4.
4. Memory devices should be enabled only during E high to avoid port 3 bus contention.
5. Item 26 is different from the MC6801 but it is upward compatible.

Figure 7. Bus Timing

INTRODUCTION

The MC6801U4 is an 8-bit monolithic microcontroller that can be configured to function in a wide variety of applications. The facility that provides this extraordinary flexibility is its ability to be hardware programmed into eight different operating modes. The operating mode controls the configuration of 18 of the 40 MCU pins, available on-chip resources, memory map, location (internal or external) of interrupt vectors, and type of external bus. The configuration of the remaining 22 pins is not dependent on the operating mode.

Twenty-nine pins are organized as three 8-bit ports and one 5-bit port. Each port consists of at least a data register and a write-only data direction register. The data direction register is used to define whether corresponding bits in the data register are configured as an input (clear) or output (set).

The term "port" by itself refers to all of the hardware associated with the port. When the port is used as a "data port" or "I/O port," it is controlled by the port data direction register and the programmer has direct access to the port pins using the port data register. Port pins are labeled as Pij where i identifies one of four ports and j indicates the particular bit.

The MPU is an enhanced MC6800 MPU with additional capabilities and greater throughput. It is upward source and object-code compatible with the MC6800 and the MC6801. The programming model is depicted in Figure 8 where accumulator D is a concatenation of accumulators A and B. A list of new operations added to the MC6800 instruction set are shown in Table 1.

The MC6803U4 can be considered an MC6801U4 that operates in modes 2 and 3 only.

OPERATING MODES

The MC6801U4 provides seven different operating modes (modes 0 through 3 and 5 through 7), and the MC6803U4 provides two operating modes (modes 2 and 3). The operating modes are hardware selectable and determine the device memory map, the configuration of port 3, port 4, SC1, SC2, and the physical location of the interrupt vectors.

FUNDAMENTAL MODES

The seven operating modes (0-3, 5-7) can be grouped into three fundamental modes which refer to the type of bus it supports: single chip, expanded nonmultiplexed, and expanded multiplexed. Single chip is mode 7, expanded nonmultiplexed is mode 5, and the remaining 5 are expanded-multiplexed modes. Table 2 summarizes the characteristics of the operating modes.

MC6801U4 Single-Chip Mode (7)

In the single-chip mode, the four MCU ports are configured as parallel I/O data ports, as shown in Figure 9. The MCU functions as a monolithic microcontroller in this mode without external address or data buses. A maximum of 29 I/O lines and two port 3 control lines are provided. Peripherals or another MCU can be interfaced to port 3 in a loosely coupled dual-processor configuration, as shown in Figure 10.

MC6801U4 Expanded-Nonmultiplexed Mode (5)

A modest amount of external memory space is provided in the expanded-nonmultiplexed mode while significant on-chip resources are retained. Port 3 functions as an 8-bit bidirectional data bus, and port 4 is configured initially as an input data port. Any combination of the eight least-significant address lines may be obtained by writing to the port 4 data direction register. Stated alternatively, and combination of A0 to A7 may be provided while retaining the remainder as input data lines. Internal pullup resistors pull the port 4 lines high until the port is configured.

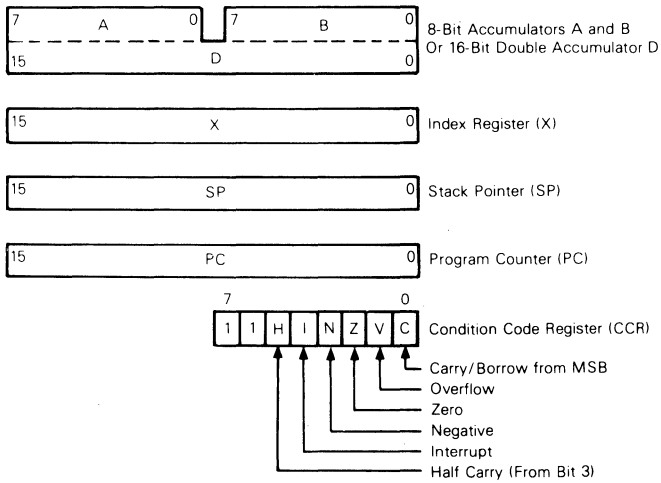


Figure 8. Programming Model

Table 1. New Instructions

Instruction	Description
ABX	Unsigned addition of accumulator B to index register
ADDD	Adds (without carry) the double accumulator to memory and leaves the sum in the double accumulator
ASLD or LSLD	Shifts the double accumulator left (towards MSB) one bit, the LSB is cleared, and the MSB is shifted into the C bit
BHS	Branch if higher or same, unsigned conditional branch (same as BCC)
BLO	Branch if lower, unsigned conditional branch (same as BCS)
BRN	Branch never
JSR	Additional addressing mode direct
LDD	Loads double accumulator from memory
LSL	Shifts memory or accumulator left (towards MSB) one bit, the LSB is cleared, and the MSB is shifted into the C bit (same as ASL)
LSRD	Shifts the double accumulator right (towards LSB) one bit, the MSB is cleared, and the LSB is shifted into the C bit
MUL	Unsigned multiply, multiplies the two accumulators and leaves the product in the double accumulator
PSHX	Pushes the index register to stack
PULX	Pulls the index register from stack
STD	Stores the double accumulator to memory
SUBD	Subtracts memory from the double accumulator and leaves the difference in the double accumulator
CPX	Internal processing modified to permit its use with any conditional branch instruction

3

Table 2. Summary of MC6801U4/MC6803U4 Operating Modes

Single-Chip (Mode 7)
- 192 bytes of RAM, 4096 bytes of ROM
- Port 3 is a parallel I/O port with two control lines
- Port 4 is a parallel I/O port

Expanded Non-Multiplexed (Mode 5)
- 192 bytes of RAM, 4096 bytes of ROM
- 256 bytes of external memory space
- Port 3 is an 8-bit data bus
- Port 4 is an input port/address bus

Expanded Multiplexed (Modes 0, 1, 2, 3, 6*)
- Four memory space options (total 64K address space)
  - (1) Internal RAM and ROM with partial address bus (mode 1)
  - (2) Internal RAM, no ROM (mode 2)
  - (3) Extended addressing of internal I/O and RAM
  - (4) Internal RAM and ROM with partial address bus (mode 6)
- Port 3 is multiplexed address/data bus
- Port 4 is address bus (inputs/address in mode 6)
- Test mode (mode 0):
  - May be used to test internal RAM and ROM
  - May be used to test ports 3 and 4 as I/O ports by writing into mode 7
- Only modes 5, 6, and 7 can be irreversibly entered from mode 0

Resources Common to All Modes
- Reserved register area
- Port 1 input/output operation
- Port 2 input/output operation
- Timer operation
- Serial communications interface operation

*The MC6803U4 operates only in modes 2 and 3.

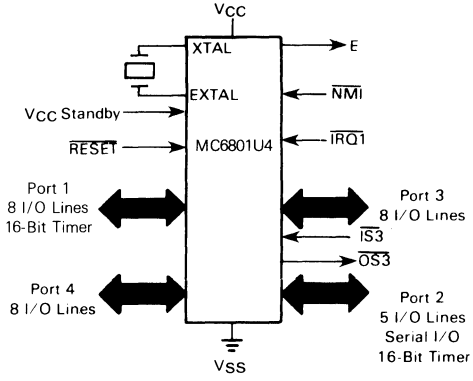


Figure 9. Single-Chip Mode

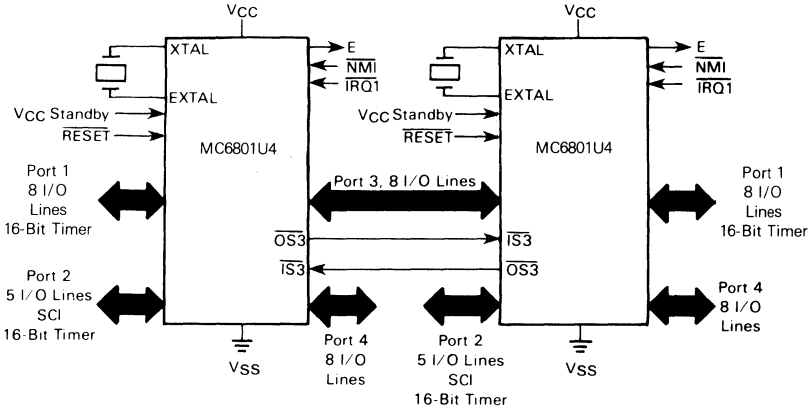


Figure 10. Single-Chip Dual Processor Configuration

Figure 11 illustrates a typical system configuration in the expanded-nonmultiplexed mode. The MCU interfaces directly with M6800 family parts and can access 256 bytes of external address space at $100 through $1FF. IOS provides an address decode of external memory ($100-$1FF) and can be used as a memory-page select or chip-select line.

Expanded-Multiplexed Modes (0, 1, 2, 3, 6)

A 64K-byte memory space is provided in the expanded-multiplexed modes. In each of the expanded-multiplexed modes, port 3 functions as a time-multiplexed address/data bus with address valid on the negative edge of address strobe (AS) and data valid while E is high. In modes 0, 2, and 3, port 4 provides address lines A8 to A15. In modes 1 and 6, however, port 4 initially is configured at reset as an input data port. The port 4 data direction register can then be changed to provide any combination of address lines A8 to A15. Stated alternatively, any subset of A8 to A15 can be provided while retaining the remaining port 4 lines as input data lines. Internal pullup resistors pull the port 4 lines high until software configures the port. In mode 1, the internal pullup resistors will hold the upper address lines high, producing a value of $FFXX for a reset vector. A simple method of getting the desired address lines configured as outputs is to have an external EPROM not fully decoded so it appears at two address locations (i.e., $FXXX and $BXXX). Then, when the reset vector appears as $FFFE, the EPROM will be accessed and can point to an address in the top $100 bytes of the internal or exteral ROM/EPROM that will configure port 4 as desired.

In mode 0, the reset and interrupt vectors are located at $BFF0-$BFFF. In addition, the internal and external data buses are connected; therefore, there must be no memory map overlap to avoid potential bus conflicts. By writing the PC0-PC2 bits in the port 2 data register, modes 5, 6, and 7 can be irreversibly entered from mode 0. Mode 0 is used primarily to verify the ROM pattern and to monitor the internal data bus with the automated test equipment.

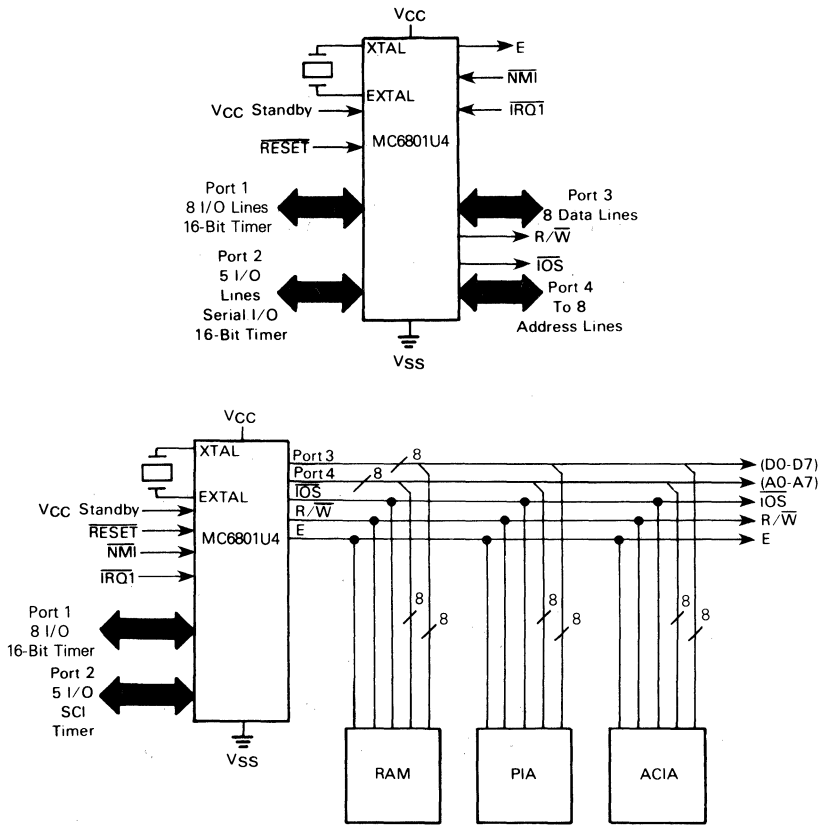


Figure 11. Expanded-Nonmultiplexed Configuration

Only the MC6801U4 can operate in each of the expanded-multiplexed modes. The MC6803U4 operates only in modes 2 and 3.

Figure 12 depicts a typical configuration for the expanded-multiplexed modes. The AS can be used to control a transparent D-type latch to capture addresses A0-A7, which allows port 3 to function as a data bus when E is high, as shown in Figure 13.

PROGRAMMING THE MODE

The operating mode is determined at $\overline{\text{RESET}}$ by the levels asserted on P22, P21, and P20. These levels are latched into PC2, PC1, and PC0 of the program control register on the positive edge of $\overline{\text{RESET}}$. The operating mode may be read from the port 2 data register, as shown below, and programming levels and timing must be met as shown in Figure 14. A brief outline of the operating modes is shown in Table 3.

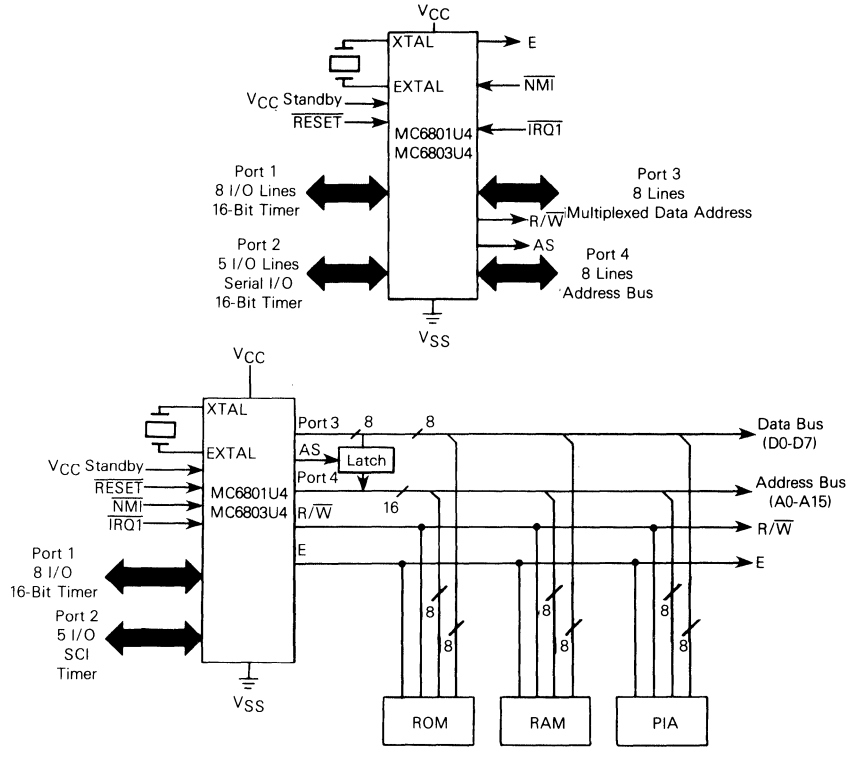
Circuitry to provide the programming levels is dependent primarily on the normal system usage of the three pins. If configured as outputs, the circuit shown in Figure 15 may be used; otherwise, three-state buffers can be used to provide isolation while programming the mode. If diodes are used to program the mode, the diode forward voltage drop must not exceed the V_{MPDD} minimum.

PORT 2 DATA REGISTER

7	6	5	4	3	2	1	0	
PC2	PC1	PC0	P24	P23	P22	P21	P20	$03

MEMORY MAPS

The MC6801U4/MC6803U4 can provide up to 64K-byte address space, depending on the operating mode. A memory map for each operating mode is shown in Figure 16. The first 32 locations of each map are reserved for the internal register area, as shown in Table 4, with exceptions as indicated.



NOTE: To avoid data bus (port 3) contention in the expanded multiplexed modes, memory devices should be enabled only during E high time.

Figure 12. Expanded-Multiplexed Configuration

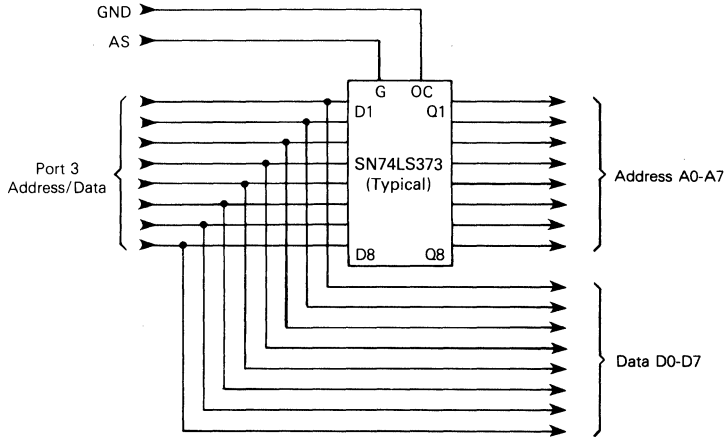
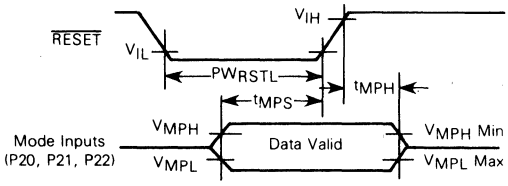
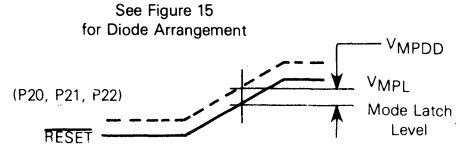


Figure 13. Typical Latch Arrangement





MODE PROGRAMMING (Refer to Figure 14)

Characteristic	Symbol	Min	Max	Unit
Mode Programming Input Voltage Low (For T_A=0–70°C)	V_{MPL}	–	1.8	V
Mode Programming Input Voltage High	V_{MPH}	4.0	–	V
Mode Programming Diode Differential (If Diodes are Used) (For T_A=0–70°C)	V_{MPDD}	0.6	–	V
$\overline{RESET}$ Low Pulse Width	PW_{RSTL}	3.0	–	E Cycles
Mode Programming Setup Time	t_{MPS}	2.0	–	E Cycles
Mode Programming Hold Time $\overline{RESET}$ Rise Time ≥ 1 μs $\overline{RESET}$ Rise Time < 1 μs	t_{MPH}	0 100	– –	ns

NOTE:
For T_A = −40–85°C, V_{MPL} Max = 1.7, and V_{MPDD} Min = 0.4

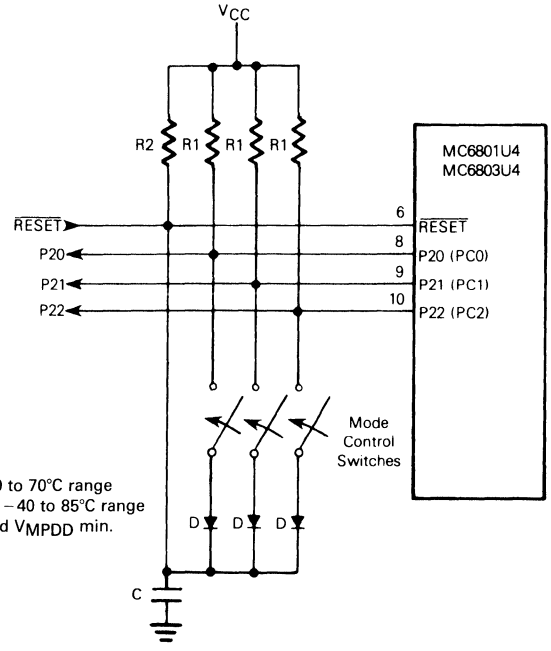
Figure 14. Mode Programming Timing

Table 3. Mode Selection Summary

Mode*	P22 PC2	P21 PC1	P20 PC0	ROM	RAM	Interrupt Vectors	Bus Mode	Operating Mode
7	H	H	H	I	I	I	I	Single Chip
6	H	H	L	I	I	I	MUX(2, 3)	Multiplexed/Partial Decode
5	H	L	H	I	I	I	NMUX(2, 3)	Nonmultiplexed/Partial Decode
4	H	L	L	–	–	–	–	Undefined(4)
3	L	H	H	E	I	E	MUX(1, 5)	Multiplexed/RAM
2	L	H	L	E	I	E	MUX(1)	Multiplexed/RAM
1	L	L	H	I	I	E	MUX (2,3)	Multiplexed/RAM and ROM
0	L	L	L	I	I	E	MUX(1)	Multiplexed Test

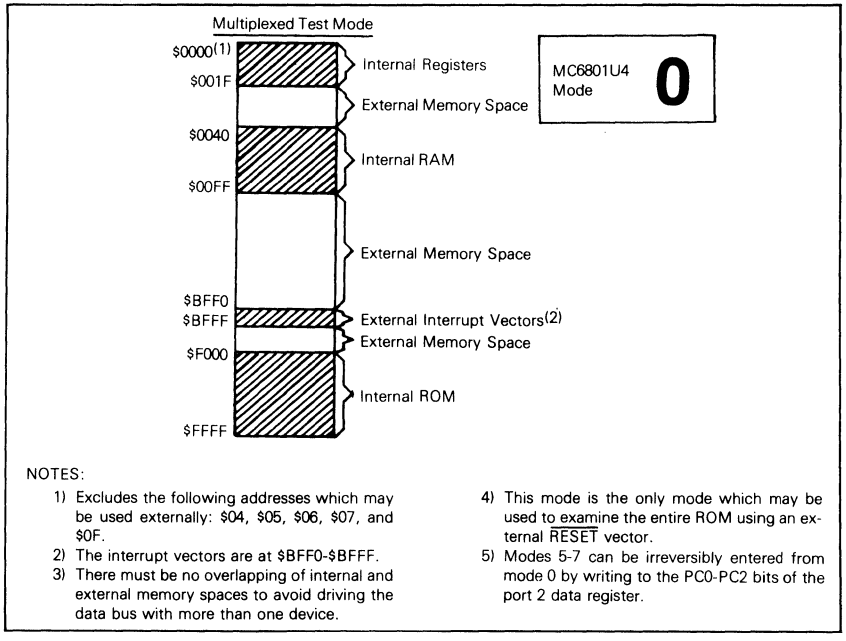
LEGEND
I — Internal
E — External
MUX — Multiplexed
NMUX — Nonmultiplexed
L — Logic "0"
H — Logic "1"

NOTES:
1. Addresses associated with ports 3 and 4 are considered external in modes 0, 2, and 3.
2. Addresses associated with port 3 are considered external in modes 1, 5, and 6.
3. Port 4 default is user data input; address output is optional by writing to port 4 data direction register.
4. Mode 4 is a nonuser mode and should not be used as an operating mode.
5. Mode 3 has the internal RAM and internal registers relocated at $D000-$D0FF.



NOTES:
1. Mode 7 as shown
2. R2•C = Reset time constant
3. R1 = 10 k (typical)
4. D = 1N914, 1N4001 in the 0 to 70°C range
   D = 1N270, MBD201 in the −40 to 85°C range
5. Diode V_f should not exceed V_{MPDD} min.

Figure 15. Typical Mode Programming Circuit



NOTES:
1) Excludes the following addresses which may be used externally: $04, $05, $06, $07, and $0F.
2) The interrupt vectors are at $BFF0-$BFFF.
3) There must be no overlapping of internal and external memory spaces to avoid driving the data bus with more than one device.
4) This mode is the only mode which may be used to examine the entire ROM using an external RESET vector.
5) Modes 5-7 can be irreversibly entered from mode 0 by writing to the PC0-PC2 bits of the port 2 data register.

Figure 16. MC6801U4/MC6803U4 Memory Maps (Sheet 1 of 4)

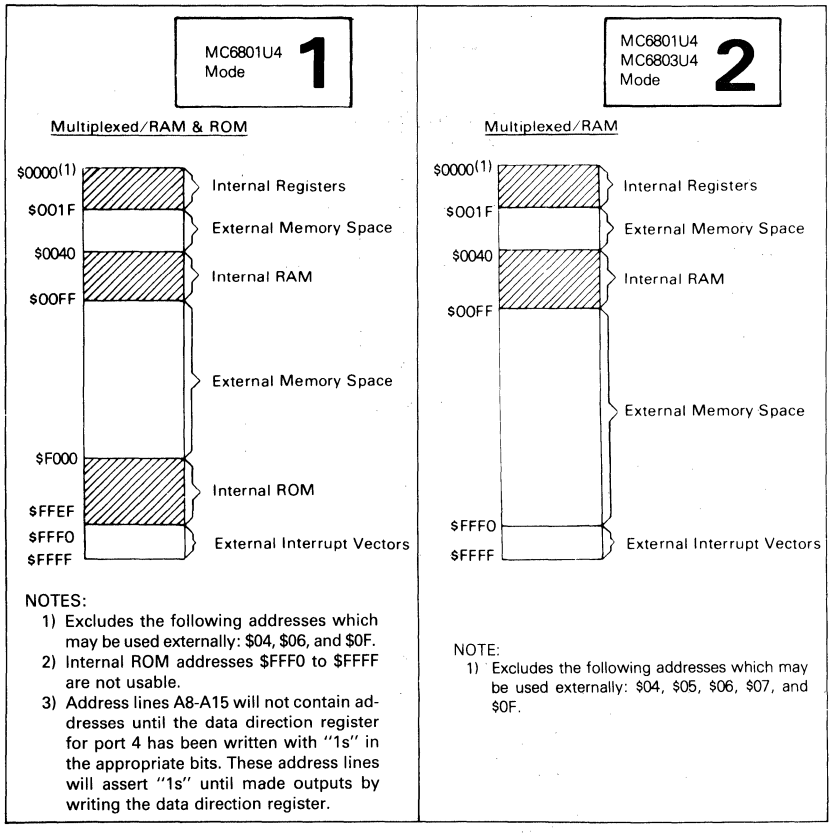


Figure 16. MC6801U4/MC6803U4 Memory Maps (Sheet 2 of 4)

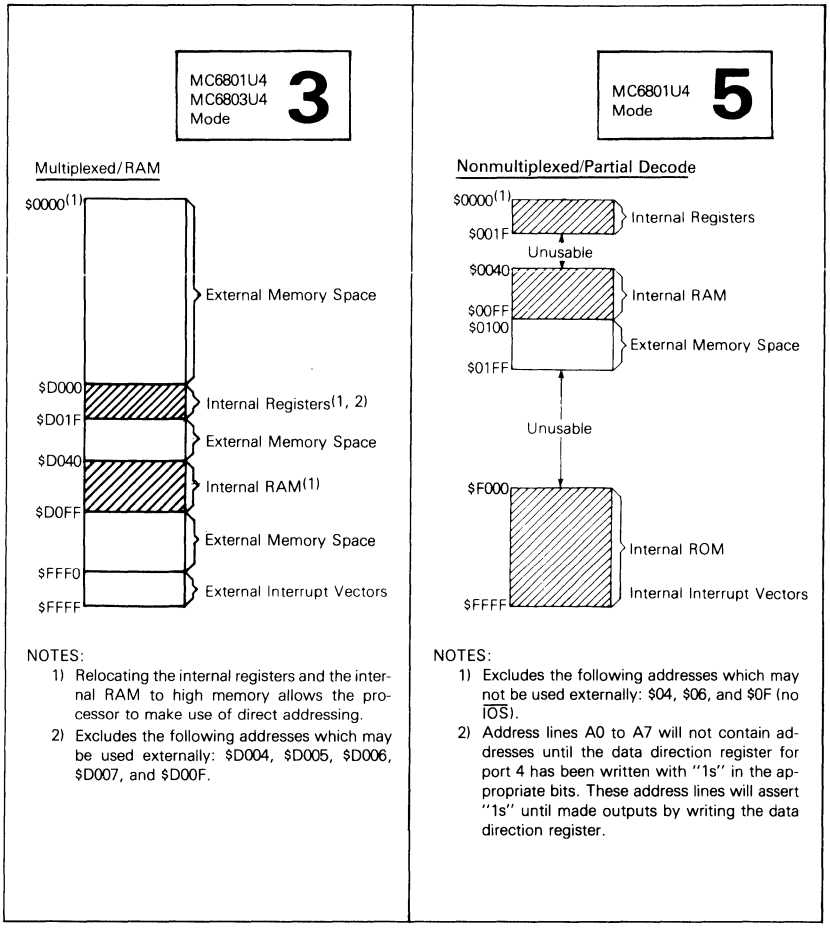
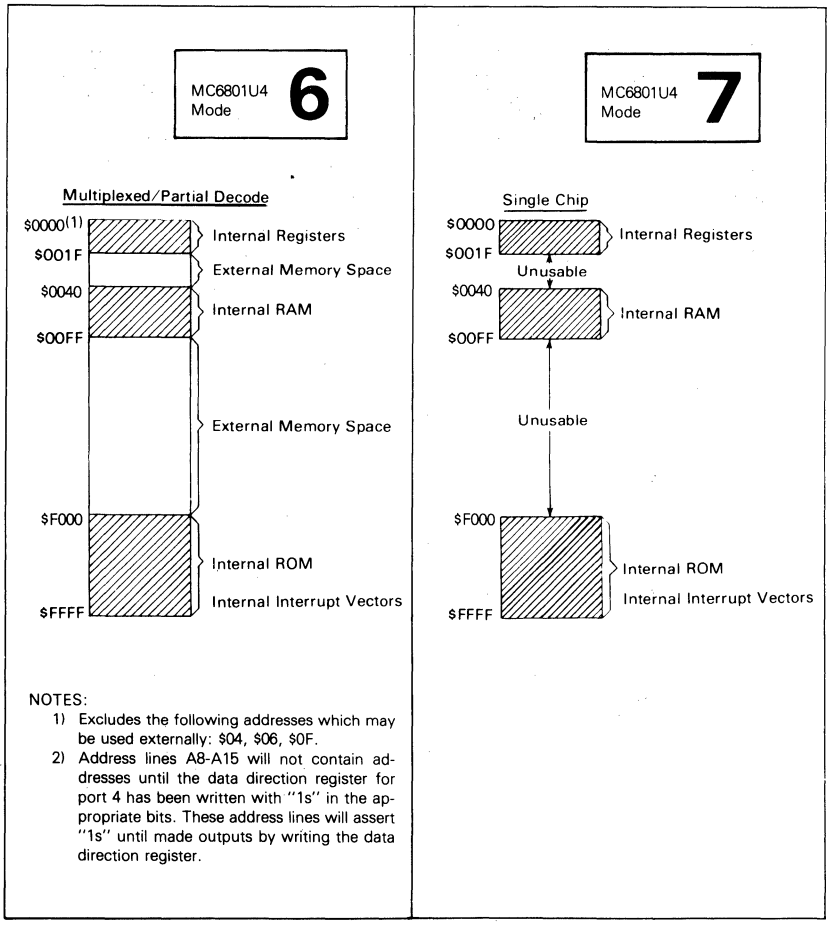


Figure 16. MC6801U4/MC6803U4 Memory Maps (Sheet 3 of 4)



NOTES:
1) Excludes the following addresses which may be used externally: $04, $06, $0F.
2) Address lines A8-A15 will not contain addresses until the data direction register for port 4 has been written with "1s" in the appropriate bits. These address lines will assert "1s" until made outputs by writing the data direction register.

Figure 16. MC6801U4/MC6803U4 Memory Maps (Sheet 4 of 4)

Table 4. Internal Register Area

Register	Address: Other Modes	Address: Mode 3
Port 1 Data Direction Register* * *	0000	D000
Port 2 Data Direction Register* * *	0001	D001
Port 1 Data Register	0002	D002
Port 2 Data Register	0003	D003
Port 3 Data Direction Register* * *	0004*	D004*
Port 4 Data Direction Register* * *	0005* *	D005* *
Port 3 Data Register	0006*	D006*
Port 4 Data Register	0007* *	D007* *
Timer Control and Status Register	0008	D008
Counter (High Byte)	0009	D009
Counter (Low Byte)	000A	D00A
Output Compare Register (High Byte)	000B	D00B
Output Compare Register (Low Byte)	000C	D00C
Input Capture Register (High Byte)	000D	D00D
Input Capture Register (Low Byte)	000E	D00E
Port 3 Control and Status Register	000F*	D00F*
Rate and Mode Control Register	0010	D010
Transmit/Receive Control and Status Register	0011	D011
Receive Data Register	0012	D012
Transmit Data Register	0013	D013
RAM Control Register	0014	D014
Counter Alternate Address (High Byte)	0015	D015
Counter Alternate Address (Low Byte)	0016	D016
Timer Control Register 1	0017	D017
Timer Control Register 2	0018	D018
Timer Status Register	0019	D019
Output Compare Register 2 (High Byte)	001A	D01A
Output Compare Register 2 (Low Byte)	001B	D01B
Output Compare Register 3 (High Byte)	001C	D01C
Output Compare Register 3 (Low Byte)	001D	D01D
Input Capture Register 2 (High Byte)	001E	D01E
Input Capture Register 2 (Low Byte)	001F	D01F

*External addresses in modes 0, 1, 2, 3, 5, and 6 cannot be accessed in mode 5 (no IOS).

**External Addresses in Modes 0, 2, and 3.

***1 = Output, 0 = Input

MC6801U4/MC6803U4 INTERRUPTS

The M6801 Family supports two types of interrupt requests: maskable and nonmaskable. A nonmaskable interrupt ($\overline{NMI}$) is always recognized and acted upon at the completion of the current instruction. Maskable interrupts are controlled by the condition code register I bit and by individual enable bits. The I bit controls all maskable interrupts. Of the maskable interrupts, there are two types: $\overline{IRQ1}$ and $\overline{IRQ2}$. The programmable timer and serial communications interface use an internal $\overline{IRQ2}$ interrupt line, as shown in the block diagram. External devices and IS3 use $\overline{IRQ1}$. An $\overline{IRQ1}$ interrupt is serviced before $\overline{IRQ2}$ if both are pending.

NOTE

After reset, an $\overline{NMI}$ will not be serviced until the first program load of the stack pointer. Any $\overline{NMI}$ generated before this load will be remembered by the processor and serviced subsequent to the stack pointer load.

All $\overline{IRQ2}$ interrupts use hardware-prioritized vectors. The single SCI interrupt and three timer interrupts are serviced in a prioritized order, and each is vectored to a separate location. All interrupt-vector locations are shown in Table 5. In mode 0, reset and interrupt vectors are defined as $BFF0-$BFFF.

The interrupt flowchart, which is depicted in Figure 17, is common to every interrupt excluding reset. During interrupt servicing, the program counter, index register, A accumulator, B accumulator, and condition code register are pushed to the stack. The I bit is set to inhibit maskable interrupts, and a vector is fetched corresponding to the current highest-priority interrupt. The vector is transferred to the program counter, and instruction execution is resumed. Interrupt and $\overline{RESET}$ timing are illustrated in Figures 18 and 19.

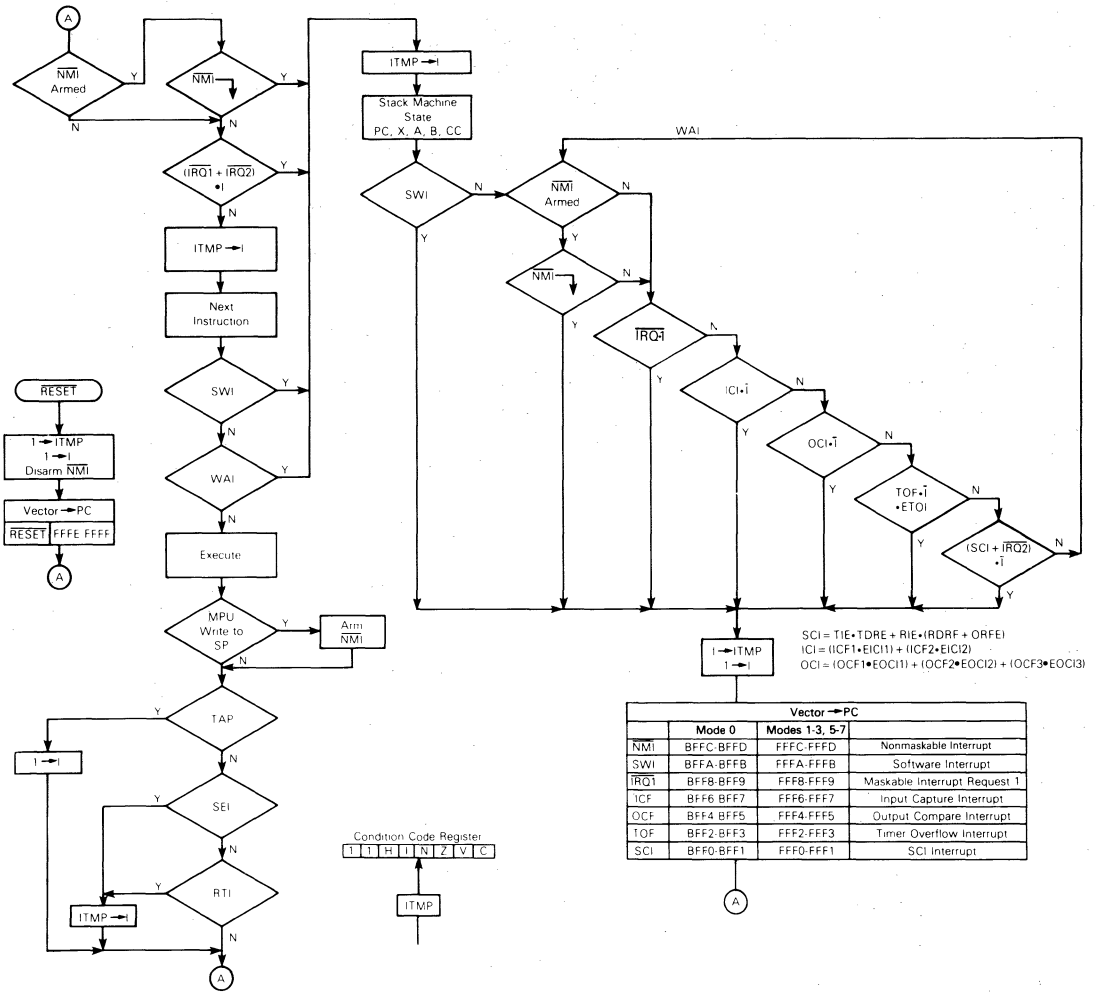
Table 5. MCU Interrupt-Vector Locations

Mode 0 MSB	Mode 0 LSB	Modes 1-3, 5-7 MSB	Modes 1-3, 5-7 LSB	Interrupt* * *
BFFE	BFFF	FFFE	FFFF	$\overline{RESET}$
BFFC	BFFD	FFFC	FFFD	Nonmaskable Interrupt**
BFFA	BFFB	FFFA	FFFB	Software Interrupt
BFF8	BFF9	FFF8	FFF9	Maskable Interrupt Request 1
BFF6	BFF7	FFF6	FFF7	Input Capture Flag*
BFF4	BFF5	FFF4	FFF5	Output Compare Flag*
BFF2	BFF3	FFF2	FFF3	Timer Overflow Flag*
BFF0	BFF1	FFF0	FFF1	Serial Communications Interface*

*$\overline{IRQ2}$ interrupt

**$\overline{NMI}$ must be armed (by accessng stack pointer) before an $\overline{NMI}$ is executed.

*** Mode 4 interrupt vectors are undefined.



Vector → PC			
	Mode 0	Modes 1-3, 5-7	
NMI	BFFC-BFFD	FFFC-FFFD	Nonmaskable Interrupt
SWI	BFFA-BFFB	FFFA-FFFB	Software Interrupt
IRQ1	BFF8-BFF9	FFF8-FFF9	Maskable Interrupt Request 1
ICF	BFF6-BFF7	FFF6-FFF7	Input Capture Interrupt
OCF	BFF4-BFF5	FFF4-FFF5	Output Compare Interrupt
TOF	BFF2-BFF3	FFF2-FFF3	Timer Overflow Interrupt
SCI	BFF0-BFF1	FFF0-FFF1	SCI Interrupt

Figure 17. Interrupt Flowchart

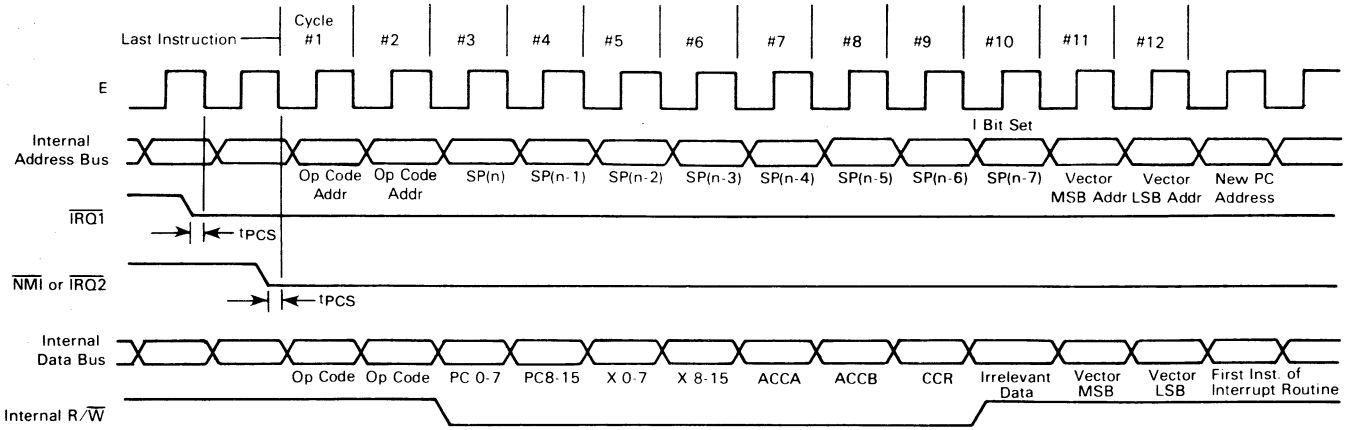


Figure 18. Interrupt Sequence

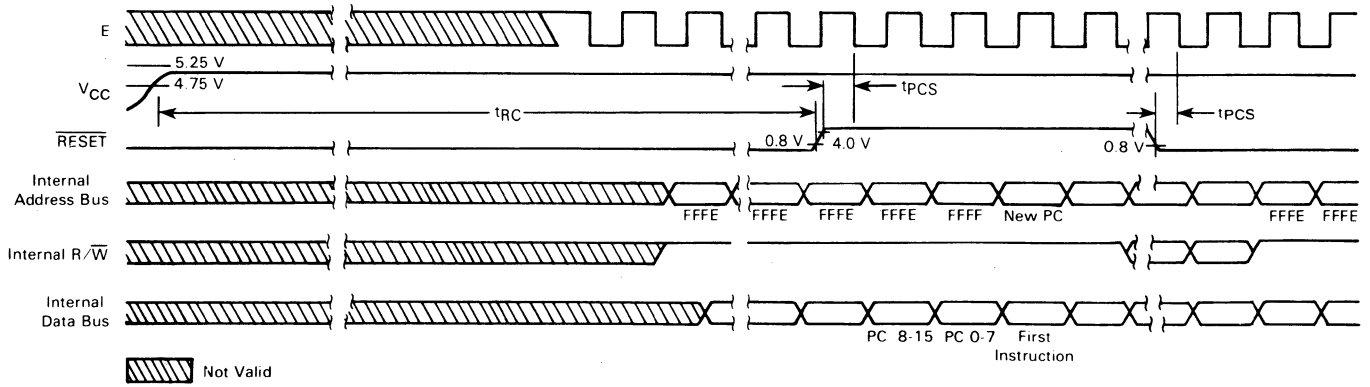


Figure 19. RESET Timing

3

FUNCTIONAL PIN DESCRIPTIONS

V_{CC} AND V_{SS}

V_{CC} and V_{SS} provide power to a large portion of the MCU. The power supply should provide +5 volts (±5%) to V_{CC}, and V_{SS} should be tied to ground. Total power dissipation (incuding V_{CC} standby) will not exceed P_D milliwatts.

V_{CC} STANDBY

V_{CC} standby provides power to the standby portion ($40 through $5F in all modes except mode 3, which is $D040 through $D05F) of the RAM, and the STBY PWR and RAME bits of the RAM control register. Voltage requirements depend on whether the device is in a powerup or powerdown state. In the powerup state, the power supply should provide +5 volts (±5%) and must reach V_{SB} volts before $\overline{RESET}$ reaches 4.0 volts. During powerdown, V_{CC} standby must remain above V_{SBB} (minimum) to sustain the standby RAM and STBY PWR bit. While in powerdown operation, the standby current will not exceed I_{SBB}.

It is typical to power both V_{CC} and V_{CC} standby from the same source during normal operation. A diode must be used between them to prevent supply power to V_{CC} during powerdown operation.

XTAL AND EXTAL

These two input pins interface either a crystal or TTL-compatible clock to the MCU internal clock geneator. Divide-by-four circuitry is included which allows use of the inexpensive 3.58-MHz or 4.4336-MHz color-burst TV crystals. A 20-pF capacitor should be tied from each crystal pin to ground to ensure reliable startup and operation. Alternatively, EXTAL may be driven by an external TTL-compatible clock at 4 f_o with a duty cycle of 50% (±5%) with XTAL connected ground.

The internal oscillator is designed to interface with an AT-cut quartz crystal resonator operated in parallel resonance mode in the frequency range specified for f_{XTAL}. The crystal should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time. The MCU is compatible with most commercially available crystals. Nominal crystal parameters are shown in Figure 20.

$\overline{RESET}$

This input is used to reset the internal state of the device and provide an orderly startup procedure. During powerup, $\overline{RESET}$ must be held below 0.8 volts: 1) at least t_{RC} after V_{CC} reaches 4.75 volts to provide sufficient time for the clock generator to stabilize, and 2) until V_{CC} standby reaches 4.75 volts. $\overline{RESET}$ must be held low at least three E cycles if asserted during powerup operation.

E (ENABLE)

This is an output clock used primarily for bus synchronization. It is TTL compatible and is the slightly skewed divide-by-four result of the device input clock frequency. It will drive one Schottky TTL load and 90 pF, and all data given in cycles are referenced to this clock unless otherwise noted.

$\overline{NMI}$ (NONMASKABLE INTERRUPT)

An $\overline{NMI}$ negative edge requests an MCU interrupt sequence, but the current instrution will be completed before it responds to the request. The MCU will then begin an interrupt sequence. Finally, a vector is fetched from $FFFC and $FFFD ($BFFC and $BFFD in mode 0), transferred to the program counter, and instruction execution is resumed. $\overline{NMI}$ typically requires a 3.3 kΩ (nominal) resistor to V_{CC}. There is no internal $\overline{NMI}$ pullup resistor. $\overline{NMI}$ must be held low for at least one E cycle to be recognized under all conditions.

NOTE

After reset, an $\overline{NMI}$ will not be serviced until the first program load of the stack pointer. Any $\overline{NMI}$ generated before this load will remain pending by the processor.

$\overline{IRQ1}$ (MASKABLE INTERRUPT REQUEST 1)

$\overline{IRQ1}$ is a level-sensitive input that can be used to request an interrupt sequence. The MPU will complete the current instruction before it responds to the request. If the interrupt mask bit (I bit) in the condition code register is clear, the MCU will begin an interrupt sequence. A vector is fetched from $FFF8 and $FFF9 ($BFF8 and $BFF9 in mode 0), transferred to the program counter, and instruction execution is resumed.

$\overline{IRQ1}$ typically requires an external 3.3 kΩ (nominal) resistor to V_{CC} for wire-OR applications. $\overline{IRQ1}$ has no internal pullup resistor.

SC1 AND SC2 (STROBE CONTROL 1 AND 2)

The function of SC1 and SC2 depends on the operating mode. SC1 is configured as an output in all modes except single-chip mode; whereas, SC2 is always an output. SC1 and SC2 can drive one Schottky load and 90 pF.

SC1 and SC2 in Single-Chip Mode

In single-chip mode, SC1 and SC2 are configured as an input and output, respectively, and both function as port 3 control lines. SC1 functions as IS3 and can be used to indicate that port 3 input data is ready or output data has been accepted.Three options associated with IS3 are controlled by the port 3 contol and status register and are discussed in the port 3 description; refer to **P30-P37 (PORT 3).** If unused, IS3 can remain unconnected.

SC2 is configured as OS3 and can be used to strobe output data or acknowledge input data. It is controlled by output strobe select (OSS) in the port 3 control and status register. The strobe is generated by a read (OSS = 0) or write (OSS = 1) to the port 3 data register. OS3 timing is shown in Figure 3.

SC1 and SC2 in Expanded-Nonmultiplexed Mode

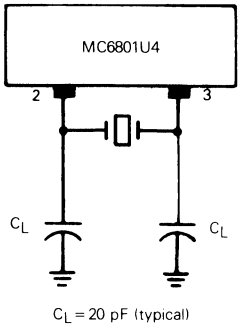
In the expanded-nonmultiplexed mode, both SC1 and SC2 are configured as outputs. SC1 functions as input/output select (IOS) and is asserted only when $0100 through $01FF is sensed on the internal address bus.

(a) Nominal Recommended Crystal Parameters

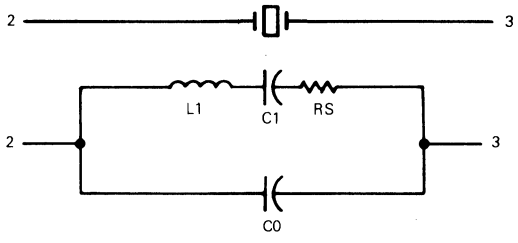
Nominal Crystal Parameters*

	3.58 MHz	4.00 MHz	5.0 MHz
RS	60 Ω	50 Ω	30-50 Ω
C0	3.5 pF	6.5 pF	4-6 pF
C1	0.015 pF	0.025 pF	0.01-0.02 pF
Q	>40 K	>30 K	>20 K

*NOTE: These are representative AT-cut crystal parameters only. Crystals of other types of cut may also be used.



C_L = 20 pF (typical)



Equivalent Circuit

NOTE
TTL-compatible oscillators may be obtain from:

Motorola Component Products
Attn: Crystal Clock Oscillators
2553 N. Edgington St.
Franklin Park, IL 60131
Tel: 312-451-1000
Telex: 433-0067

(b) Oscillator Stabilization Time (t_{RC})

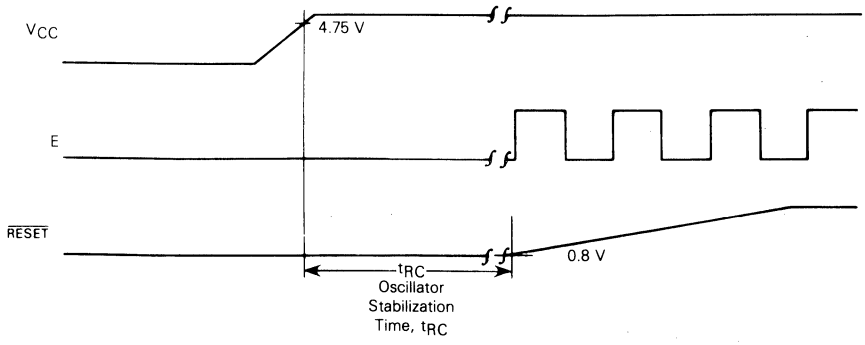


Figure 20. MC6801U4/MC6803U4 Family Oscillator Characteristics

SC2 is configured as read/write and is used to control the direction of data bus transfers. An MPU read is enabled when read/write and E are high.

SC1 and SC2 in Expanded-Multiplexed Mode

In the expanded-multiplexed modes, both SC1 nd SC2 are configured as outputs. SC1 functions as address strobe and can be used to demultiplex the eight least-significant addresses and the data bus. A latch controlled by address strobe captures the lower address on the negative edge, as shown Figure 13.

SC2 is configured as read/write and is used to control the direction of data bus transfers. An MPU read is enabled when read/write and E are high.

P10-P17 (PORT 1)

Port 1 is a mode-independent 8-bit I/O and timer port. Each line can be configured as either an input or output as defined by the port 1 data direction register. Port 1 bits 0, 1, and 2 (P10, P11, and P12) can also be used to exercise one input edge function and two output compare functions of the timer. The TTL-compatible three-state buffers can drive one Schottky TTL load and 30 pF, Darlington transistors or CMOS devices using external pullup resistors. It is configured as a data input port during $\overline{\text{RESET}}$. Unused pins can remain unconnected.

P20-P24 (PORT 2)

Port 2 is a mode-independent 5-bit multipurpose I/O port. The voltage levels present on P20, P21, and P22 on the rising edge of $\overline{\text{RESET}}$ determine the operating mode of the MCU. The entire port is then configured as a data input port. The port 2 lines can be selectively cofigured as data output lines by setting the appropriate bits in the port 2 data direction register. The port 2 data register is used to move data through the port. However, if P21 is configured as an output, it is tied to the timer output compare 1 function and cannot be used to provide output from the port 2 data register unless output enable 1 (OE1) is cleared in timer control register 1.

Port 2 can also be used to provide an interface for the serial communications interface and the timer input edge function. These configurations are described in **SERIAL COMMUNICATIONS INTERFACE** and **PROGRAMMABLE TIMER.**

The port 2 three-state TTL-compatible output buffers are capable of driving one Schottky TTL load and 30 pF or CMOS devices using external pullup resistors.

PORT 2 DATA REGISTER

7	6	5	4	3	2	1	0	
PC2	PC1	PC0	P24	P23	P22	P21	P20	$03

P30-P37 (PORT 3)

Port 3 can be configured as an I/O port, a bidirectional 8-bit data bus, or a multiplexed address/data bus depending on the operating mode. The TTL-compatible three-state output buffers can drive one Schottky TTL load and 90 pF. Unused lines can remain unconnected.

Port 3 in Single-Chip Mode

Port 3 is an 8-bit I/O port in the single-chip mode with each line configured by the port 3 data direction register. There are also two lines, IS3 and OS3, which can be used to control port 3 data transfers.

Three port 3 options are controlled by the port 3 control and status register and are available only in single-chip mode: 1) port 3 input data can be latched using IS3 as a control signal, 2) OS3 can be generated by either an MPU read or write to the port 3 data register, and 3) an $\overline{\text{IRQ1}}$ interrupt can be enabled by an IS3 negative edge. Port 3 latch timing is shown in Figure 4.

PORT 3 CONTROL AND STATUS REGISTER

7	6	5	4	3	2	1	0	
IC3 Flag	IS3 $\overline{\text{IRQ1}}$	X	OSS	Latch Enable	X	X	X	$0F

Bits 0-2 Not used.

Bit 3 **Latch Enable** — This bit controls the input latch for port 3. If set, input data is latched by an IS3 negative edge. The latch is transparent after a read of the port 3 data register. Latch enable is cleared during reset.

Bit 4 **Output Strobe Select (OSS)** — This bit determines whether OS3 will be generated by a read or write of the port 3 data register. When clear, the strobe is generated by a read; when set, it is generated by a write. OSS is cleared during reset.

Bit 5 Not used.

Bit 6 **IS3 $\overline{\text{IRQ1}}$ Enable** — When set, an $\overline{\text{IRQ1}}$ interrupt will be enabled whenever the IS3 flage is set; when clear, the interrupt is inhibited. This bit is cleared during reset.

Bit 7 **IS3 Flag** — This read-only status bit is set by an IS3 negative edge. It is cleared by a read of the port 3 data register or during reset.

Port 3 in Expanded-Nonmultiplexed Mode

Port 3 is configured as a bidirectional data bus (D7-D0) in the expanded-nonmultiplexed mode. The direction of data transfers is controlled by read/write (SC2). Data is clocked by E (enable).

Port 3 in Expanded-Multiplexed Mode

Port 3 is configured as a time-multiplexed address (A7-A0), and data bus (D7-D0) in the expanded-multiplexed mode where AS can be used to demultiplex the two buses. Port 3 is held in a high-impedance state between valid address and data to prevent bus conflicts.

P40-P47 (PORT 4)

Port 4 is configured as an 8-bit I/O port, as address outputs, or as data inputs depending on the operating

mode. Port 4, which can drive one Schottky TTL load and 90 pF, is the only port with internl pullup resistors. Unused lines can remain unconnected.

Port 4 in Single-Chip Mode

In single-chip mode, port 4 functions as an 8-bit I/O port with each line configured by the port 4 data direction register. Internal pullup resistors allow the port to directly interface with CMOS at 5-volt levels. However, external pullup resistors to more than 5 volts cannot be used.

Port 4 in Expanded-Nonmultiplexed Mode

Port 4 is configured from reset as an 8-bit input port where the port 4 data direction register can be written to provide any or all of eight address lines A0 to A7. Internal pullup resistors pull the lines high until the port 4 data direction register is configured.

Port 4 in Expanded-Multiplexed Mode

In all expanded-multiplexed modes except modes 1 and 6, port 4 functions as half of the address bus and provides A8 to A15. In modes 1 and 6, the port is configured from reset as an 8-bit parallel input port where the port 4 data direction register can be written to provide any or all of upper address lines A8 to A15. Internal pullup resistors pull the lines high until the port 4 data direction register is configured where bit 0 controls A8.

RESIDENT MEMORY

The MC6801U4 provides 4096 bytes of on-chip ROM and 192 bytes of on-chip RAM.

Thirty-two bytes of the RAM ae powered through the V_{CC} standby pin and are maintainable during V_{CC} powerdown. This standby portion of the RAM consists of 32 bytes located from $40 through $5F in all modes except mode 3, which is $D040 through $D05F.

Power must be supplied to V_{CC} standby if the internal RAM is to be used, regardless of whether standby power operation is anticipated.

The RAM is controlled by the RAM control register.

RAM CONTROL REGISTER ($14)

The RAM control register includes two bits, which can be used to control RAM accesses and to determine the adequacy of the standby power source during powerdown operation. It is intended that RAME be cleared and STBY PWR be set as part of a powerdown procedure.

RAM CONTROL REGISTER

7	6	5	4	3	2	1	0	
STBY PWR	RAME	X	X	X	X	X	X	$14

Bits 0-5 Not used.

Bit 6 **RAM Enable** — This read/write bit can be used to remove the entire RAM from the internal memory map. RAME is set (enabled) during reset provided standby power is available on the positive edge of $\overline{\text{RESET}}$. If RAME is clear, any access to a RAM address is external. If RAME is set, the RAM is included in the internal map.

Bit 7 **Standby Power** — This bit is a read/write status bit that, when cleared, indicates V_{CC} standby has decreased sufficiently below V_{SBB} (minimum) to make data in the standby RAM suspect. It can be set only by software and is not affected during reset.

PROGRAMMABLE TIMER

The programmable timer can be used to perform measurements on two separate input waveforms while independently generating three output waveforms. Pulse widths can vary from several microseconds to many seconds. A block diagram of the timer is shown in Figure 21.

COUNTER ($09:0A), ($15, $16)

The key timer element is a 16-bit free-running counter that is incremented by E (enable). It is cleared during reset and is read-only with one exception: in mode 0 a write to the counter ($09) will configure it to $FFF8. This feature, intended for testing, can disturb serial operations because the counter provides the SCI internal bit rate clock. The TOF is set whenever the counter contains all ones. If ETOI is set, an interrupt will occur when the TOF is set. The counter may also be read as $15 and $16 to avoid inadvertently clearing the TOF.

OUTPUT COMPARE REGISTERS ($0B:0C), ($1A:1B), ($1C:1D)

The three output compare registers are 16-bit read/write registers, each used to control an output waveform or provide an arbitrary time-out flag. They are compared with the free-running counter during the negative half of each E cycle. When a match occurs, the corresponding output compare flag (OCF) is set, and the corresponding output level (OLVL) is clocked to an output level register. If both the corresponding output enable bit and data direction register bit are set, the value represented in the output level register will appear on the corresponding port pin. The appropriate OLVL bit can then be changed for the next compare.

The function is inhibited for one cycle after a write to its high byte ($0B, $1A, or $1C) to ensure a valid compare after a double-byte write. Writes can be made to either byte of the output compare register without affecting the other byte. The OLVL value will be clocked out independently of whether the OCF had previously been cleared. The output compare registers are set to $FFFF during reset.

INPUT CAPTURE REGISTERS ($0D:0E), ($1E:1F)

The two input capture registers are 16-bit read-only registers used to store the free-running counter when a "proper" input transition occurs as defined by the corresponding input edge bit (IEDG1 or IEDG2). The input pin's data direction register should be configured as an

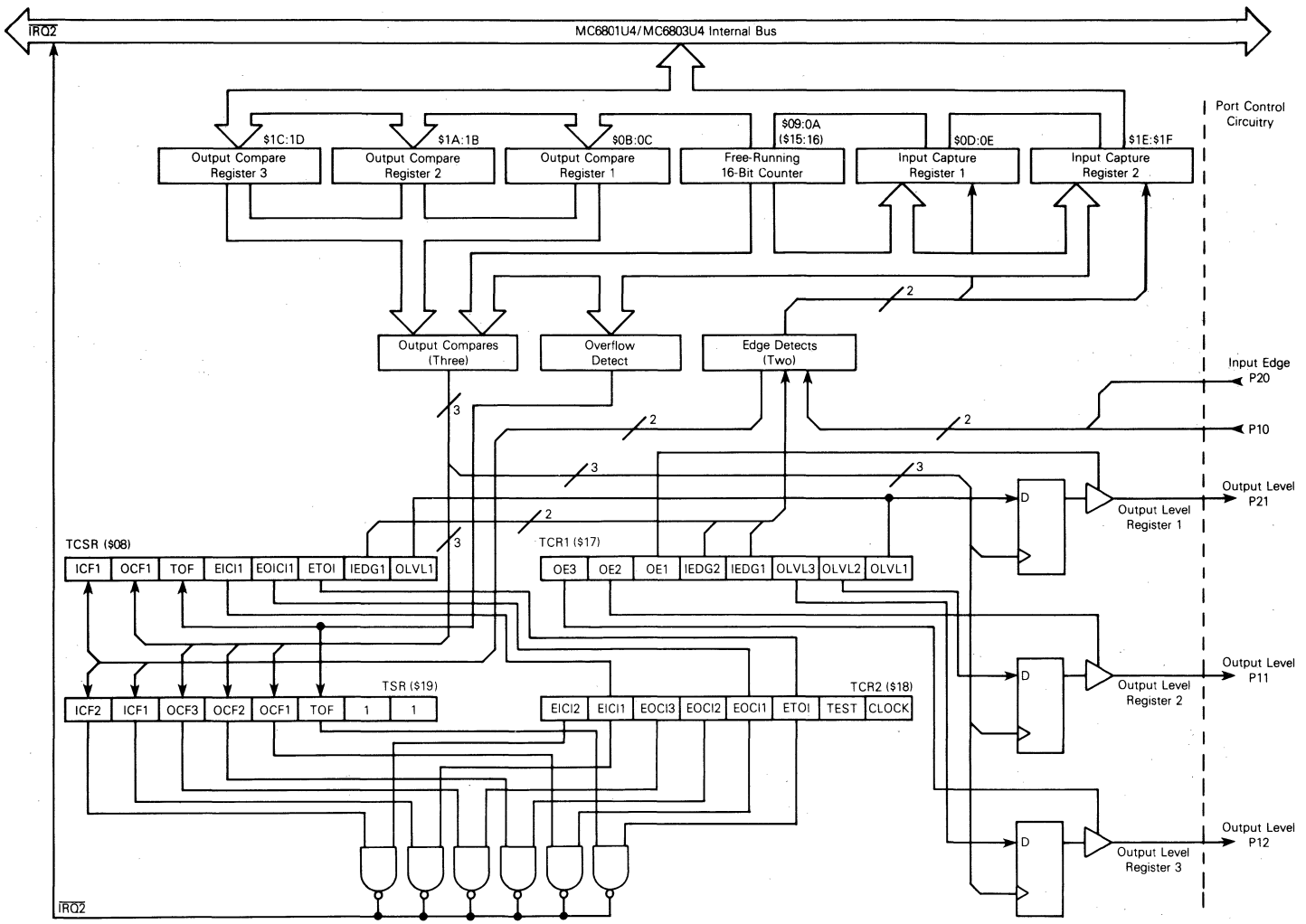


Figure 21. Block Diagram of Programmable Timer

input, but the edge detect circuit always senses P10 and P20 even when configured as an output. The counter value will be latched into the into capture registers on the second negative edge of the E clock following the transition.

As input capture can occur independently of ICF; the register always contains the most current value. However, counter transfer is inhibited between accesses of a double-byte MPU read. The input pulse width must be at least two E cycles to ensure an input capture under all conditions.

TIMER CONTROL AND STATUS REGISTERS

Four registers are used to provide the MC6801U4/MC6803U4 with control and status information about the three output compare functions, the timer overflow function, and the two input edge functions of the timer. They are as follows:

Timer Control and Status Register (TCSR)
Timer Control Register 1 (TCR1)
Timer Control Register 2 (TCR2)
Timer Status Register (TSR)

Timer Control and Status Register (TCSR) ($08)

The timer control and status register is an 8-bit register in which all bits are readable, while only bits 0-4 can be written. All the bits in this register are also accessible through the two timer control registers and the timer status register. The three most significant bits provide the timer status and indicate if

1. a proper level transition has been detected at P20;
2. a match has occurred between the free-running counter and output compare register 1; or
3. the free-running counter has overflowed.

Each of the three events can generate an $\overline{IRQ2}$ interrupt and is controlled by an individual enable bit in the TCSR.

TIMER CONTROL AND STATUS REGISTER

7	6	5	4	3	2	1	0	
ICF1	OCF1	TOF	EICI1	EOCI1	ETOI	IEDG1	OLVL1	$08

Bit 0 **Output Level 1** — OLVL1 is clocked to output level register 1 by a successful output compare and will appear at P21 if bit 1 of the port 2 data direction register is set and the OE1 control bit in timer control register 1 is set. OLVL1 and output level register 1 are cleared during reset. Refer to **TIMER CONTROL REGISTER 1 (TCR1) ($17).**

Bit 1 **Input Edge 1** — IEDG1 is cleared during reset and controls which level transition on P20 will trigger a counter transfer to input capture register 1:
IEDG1 = 0 transfer on a negative-edge
IEDG1 = 1 transfer on a positive-edge
Refer to **TIMER CONTROL REGISTER 1 (TCR1) ($17).**

Bit 2 **Enable Timer Overflow Interrupt** — When set, an $\overline{IRQ2}$ interrupt will be generated when the timer overflow flag is set; when clear, the interrupt is inhibited ETOI is cleared during reset. Refer to **TIMER CONTROL REGISTER 2 (TCR2) ($18).**

Bit 3 **Enable Output Compare Interrupt 1** — When set, an $\overline{IRQ2}$ interrupt will be generated when output compare flag 1 is set; when clear,the interrupt is inhibited. EOCI1 is cleared during reset. Refer to **TIMER CONTROL REGISTER 2 (TCR2) ($18).**

Bit 4 **Enable Output Capture Interrupt 1** — When set, an $\overline{IRQ2}$ interrupt will be generated when input capture flag 1 is set; when clear, the interrupt is inhibited. EICI1 is cleared during reset. Refer to **TIMER CONTROL REGISTER 2 (TCR2) ($18).**

Bit 5 **Timer Overflow Flag** — The TOF is set when the counter contains all ones ($FFFF). It is cleared by reading the TCSR or the TSR (with TOF set) and the counter high byte ($09), or during reset. Refer to **TIMER STATUS REGISTER (TSR) ($19).**

Bit 6 **Output Compare Flag 1** — OCF1 is set when output compare register 1 matches the free-running counter. OCF1 is cleared by reading the TCSR or the TSR (with OCF1 set) and then writing to output compare register 1 ($0B or $0C), or during reset. Refer to **TIMER STATUS REGISTER (TSR) ($19).**

Bit 7 **Input Capture Flag** — ICF1 is set to indicate that a proper level transition has occurred; it is cleared by reading the TCSR or the TSR (with ICF1 set) and the input capture register 1 high byte ($0D), or during reset. Refer to **TIMER STATUS REGISTER (TSR) ($19).**

Timer Control Register 1 (TCR1) ($17)

Timer control register 1 is an 8-bit read/write register containing the control bits for interfacing the output compare and input capture registers to the corresponding I/O pins.

TIMER CONTROL REGISTER 1

7	6	5	4	3	2	1	0	
OE3	OE2	OE1	IEDG2	IEDG1	OLVL3	OLVL2	OLVL1	$17

Bit 0 **Output Level 1** — OLVL1 is clocked to output level register 1 by a successful output compare and will appear at P21 if bit 1 of the port 2 data direction register is set and the OE1 control bit is set. OLVL1 and output level register 1 are cleared during reset. Refer to **TIMER CONTROL AND STATUS REGISTER (TCSR) ($08).**

Bit 1 **Output Level 2** — OLVL2 is clocked to output level register 2 by a successful output compare and will appear at P11 if bit 1 of port 1 data direction register is set and the OE2 control bit is set. OLVL2 and output level register 2 are cleared during reset.

Bit 2 **Output Level 3** — OLVL3 is clocked to output level register 3 by a successful output compare and will appear at P12 if bit 2 of port 1 data direction register is set and the OE3 control bit is set. OLVL3 and output level register 3 are cleared during reset.

Bit 3 **Input Edge 1** — IEDG1 is cleared during reset and controls which level transition on P20 will trigger a counter transfer to inut capture register 1.
IEDG1 = 0 transfer on a negative edge
IEDG1 = 1 transfer on a positive edge
Refer to **TIMER CONTROL AND STATUS REGISTER (TCSR) ($08)**.

Bit 4 **Input Edge 2** — IEDG2 is cleared during reset and controls which level transition on P10 will trigger a counter transfer to input capture register 2.
IEDG2 = 0 transfer on a negative edge
IEDG2 = 1 transfer on a positive edge

Bit 5 **Output Enable 1** — OE1 is set during reset and enables the contents of output level register 1 to be connected to P21 when bit 1 of port 2 data direction register is set.
OE1 = 0 port 2 bit 1 data register output
OE1 = 1 output level register 1

Bit 6 **Output Enable 2** — OE2 is cleared during reset and enables the contents of output level register 2 to be connected to P11 when bit 1 of port 1 data direction register is set.
OE2 = 0 port 1 bit 1 data register output
OE2 = 1 output level register 2

Bit 7 **Output Enable 3** — OE3 is cleared during reset and enables the contents of output level register 3 to be connected to P12 when bit 2 of port 1 data direction register is set
OE3 = 0 port 1 bit 2 data register output
OE3 = 1 output level register 3

Timer Control Register 2 (TCR2) ($18)

Timer control register 2 is an 8-bit read/write register (except bits 0 and 1), which enables the interrupts associated with the free-running counter, the output compare registers, and the input capture registers. In test mode 0, two more bits (clock and test) are available for checking the timer.

TIMER CONTROL REGISTER 2
(Nontest Modes)

7	6	5	4	3	2	1	0	
EICI2	EICI1	EOCI3	EOCI2	EOCI1	ETOI	1	1	$18

Bits 0-1 **Read-Only Bits** — When read, these bits return a value of 1. Refer to **TIIMER CONTROL REGISTER 2 (Test Mode)**.

Bit 2 **Enable Timer Overflow Interrupt** — When set, an $\overline{IRQ2}$ interrupt will be generated when the timer overflow flag is set; when clear, the interrupt inhibited. ETOI is cleared during rest. Refer to **TIMER CONTROL AND STATUS REGISTER (TCSR) ($08)**.

Bit 3 **Enable Output Compare Interrupt 1** — When set, an $\overline{IRQ2}$ interrupt will be generated when the output compare flag 1 is set; when clear, the interrupt is inhibited. EOCI1 is cleared during reset. Refer to **TIMER CONTROL AND STATUS REGISTER (TCSR) ($08)**.

Bit 4 **Enable Output Compare Interrupt 2** — When set, an $\overline{IRQ2}$ interrupt will be generated when the output compare flag 2 is set; when clear, the interrupt is inhibited. EOCI2 is cleared during reset.

Bit 5 **Enable Output Compare Interrupt 3** — When set, an $\overline{IRQ2}$ interrupt will be generated when the output compare flag 3 is set; when clear, the interrupt is inhibited. EOCI3 is cleared during reset.

Bit 6 **Enable Input Capture Interrupt 1** — When set, an $\overline{IRQ2}$ interrupt will be generated when the input capture flag 1 is set; when clear, the interrupt is inhibited. EICI1 is cleared during reset. Refer to **TIMER CONTROL AND STATUS REGISTER (TCSR) ($08)**.

Bit 7 **Enable Input Capture Interrupt 2** — When set, an $\overline{IRQ2}$ interrupt will be generated when the input capture flag 2 is set; when clear, the interrupt is inhibited. EICI2 is cleaed during reset.

The timer test bits (test and clock) allow the free-running counter to be tested as two separate 8-bit counters to speed testing.

TIMER CONTROL REGISTER 2
(Test Modes)

7	6	5	4	3	2	1	0	
EICI2	EICI1	EOCI3	EOCI2	EOCI1	ETOI	TEST	CLOCK	$18

Bit 0 **CLOCK** — The CLOCK control bit selects which half of the 16-bit free-running counter (MSB or LSB) should be clocked with E. The CLOCK bit is a read/write bit only in mode 0 and is set during reset.

CLOCK = 0 — Only the eight most-significant bits of the free-running counter run with TEST = 0.

CLOCK = 1 — Only the eight least-significant bits of the free-running counter run when TEST = 0.

Bit 1 **TEST** — The TEST control bit enables the timer test mode. TEST is a read/write bit in mode 0 and is set during reset.

TEST = 0 — Timer test mode enabled:
a) The timer LSB latch is transparent, which allows the LSB to be read independently of the MSB.

b) Either the MSB or the LSB of the timer is clocked by E, as defined by the CLOCK bit.

TEST = 1 — Timer test mode disabled.

Bits 2-7 See **TIMER CONTROL REGISTER 2 (Nontest Modes)**. (These bits function the same as in the nontest modes.)

Timer Status Register (TSR) ($19)

The timer status register is an 8-bit read-only register which contains the flags associated with the free-running counter, the output compare registers, and the input capture registers.

TIMER STATUS REGISTER

7	6	5	4	3	2	1	0	
ICF2	ICF1	OCF3	OCF2	OCF1	TOF	1	1	$19

Bits 0-1 Not used.

Bit 2 **Timer Overflow Flag** — The TOF is set when the counter contais all ones ($FFFF). It is cleared by reading the TSR or the TCSR (with TOF set) and then the counter high byte ($09), or during reset. Refer to **TIMER CONTROL AND STATUS REGISTER (TCSR) ($08)**.

Bit 3 **Output Compare Flag 1** — OCF1 is set when output compare register 1 matches the free-running counter. OCF1 is cleared by reading the TSR or the TCSR (with OCF1 set) and then writing to output compare register 1 ($0B or $0C), or during reset. Refer to **TIMER CONTROL AND STATUS REGISTER (TCSR) ($08)**.

Bit 4 **Output Compare Flag 2** — OCF2 is set when output compare register 2 matches the free-running counter. OCF2 is cleared by reading the TSR (with OCF2 set) and then writing to output compare register 2 ($1A or $1B), or during reset.

Bit 5 **Output Compare Flag 3** — OCF3 is set when output compare register 3 matches the free-runing counter. OCF3 is cleared by reading the TSR (with OCF3 set) and then writing to output compare register 3 ($1C or $1D), or during reset.

Bit 6 **Input Capture Flag 1** — ICF1 is set to indicate that a proper level transition has occurred; it is cleared by reading the TSR or the TCSR (with ICF1 set) and the input capture register 1 high byte ($0D), or during reset. Refer to **TIMER CONTROL AND STATUS REGISTER (TCSR) ($08)**.

Bit 7 **Input Capture Flag 2** — ICF2 is set to indicate that a proper level transition has occurred; it is cleared by reading the TSR (with ICF2 set) and the input capture register 2 high byte ($1E), or during reset.

SERIAL COMMUNICATIONS INTERFACE

A full-duplex asynchronous SCI is provided with two data formats and a variety of rates. The SCI transmitter and receiver are functionally independent but use the same data format and bit rate. Serial data formats include standard mark/space (NRZ) and biphase; both provide one start bit, eight data bits, and one stop bit. "Baud" and "bit rate" are used synonymously in the following description.

WAKE-UP FEATURE

In a typical serial loop multiprocessor configuration, the software protocol will usually identify the address-see(s) at the begining of the message. To permit uninterested MPUs to ignore the remainder of the message, wake-up feature is included whereby all further SCI receiver flag (and interrupt) processing can be inhibited until its data line goes idle. An SCI receiver is re-enabled by an idle string of eleven consecutive ones or during reset. Software must provide for the required idle string between consecutive messages and must prevent it within messages.

PROGRAMMABLE OPTIONS

The following featues of the SCI are programmable:

- Format: standard mark/space (NRZ) or biphase
- Clock: external or internal bit rate clock
- Baud: one of eight per E clock frequency or external clock (×8 desired baud)
- Wake-Up Feature: enabled or disabled
- Interrupt Requests: enabled individually for transmitter and receiver
- Clock Output: internal bit rate clock enabled or disabled to P22

SERIAL COMMUNICATIONS REGISTERS

The SCI includes four addressable registers as depicted in Figure 22. It is controlled by the rate and mode control register and the transmit/receive control and status register. Data are transmitted and received utilizing a write-only transmit register and a read-only receive register. The shift registes are not accessible to software.

Rate and Mode Control Register (RMCR) ($10)

The rate and mode control register controls the SCI bit rate, format, clock source, and, under certain conditions, the configuration of P22. The register consists of five write-only bits which are cleared during reset. The two least-significant bits, in conjunction with bit 7, control the bit rate of the internal clock, and the remaining two bits control the format and clock source.

RATE AND MODE CONTROL REGISTER

7	6	5	4	3	2	1	0	
EBE	X	X	X	CC1	CC0	SS1	SS0	$10

Bit 1:Bit 0 **SS1:SS0 Speed Select** — These two bits select the baud when using the internal clock. Eight

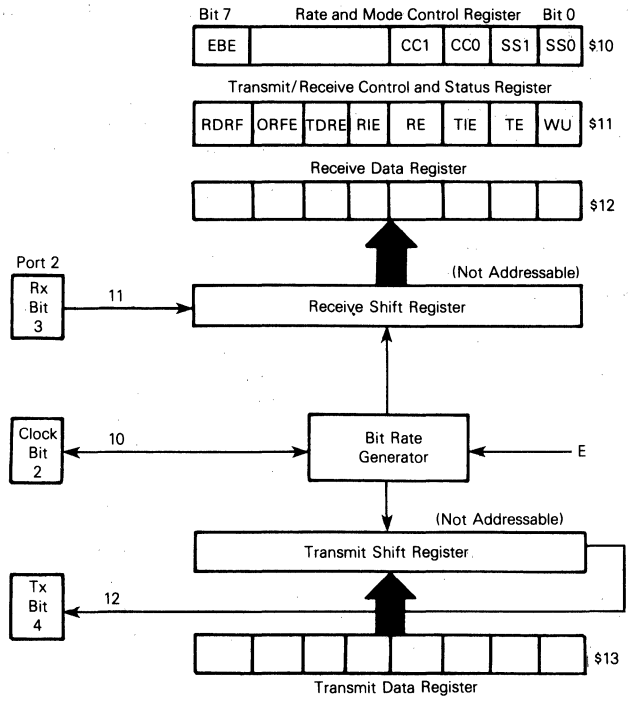


Figure 22. SCI Registers

rates may be selected (in conjunction with bit 7) which are a function of the MCU input frequency. Table 6 lists bit times and rates for three selected MCU frequencies.

Bit 3:Bit 2 **CC1:CC0 Clock Control and Format Select** — These two bits control the format and select the serial clock source. If CC1 is set, the DDR value for P22 is forced to the complement of CC0 and cannot be altered until CC1 is cleared. If CC1 is cleared after having been set, its DDR value is unchanged. Table 7 defines the formats, clock source, and use of P22.

Bits 4-6

Not used.

Bit 7 **EBE Enhanced Baud Enable** — EBE selects the standard MC6801 baud rates when clear and the additional baud rates when set (Table 6). This bit is cleared by reset and is a write-only control bit.

Bit 1:Bit 1 EBE = 0 standard MC6801 baud rates
Bit 1:Bit 1 EBE = 1 additional baud rates

If both CC1 and CC0 are set, an external TTL-compatible clock must be connected to P22 at eight times (8×) the desired bit rate, but not greater than E, with a duty cycle of 50% (±10%). If CC1:CC0 = 10, the internal bit rate clock is provided at P22 regardless of the values for TE or RE.

NOTE

The source of SCI internal bit rate clock is the timer free-running counter. An MPU write to the counter in mode 0 can disturb serial operations.

Transmit/Receive Control and Status Register (TRCSR) ($11)

The transmit/receive control and status register controls the transmitter, receiver, wake-up feature, and two individual interrrupts, and monitors the status of serial operations. All eight bits are readable; bits 0 to 4 are also writable. The register is initialized to $20 by $\overline{\text{RESET}}$.

TRANSMIT/RECEIVE CONTROL AND STATUS REGISTER

7	6	5	4	3	2	1	0	
RDRF	ORFE	TDRE	RIE	RE	TIE	TE	WU	$11

Bit 0 **Wake-Up on Idle Line** — When set, WU enables the wake-up function; it is cleared by eleven consecutive ones or during reset. WU will not be set if the line is idle. Refer to **WAKE-UP FEATURE**.

Bit 1 **Transmit Enable** — When set, P24 DDR bit is set, cannot be changed, and will remain set if TE is

Table 6. SCI Bit Times and Rates

EBE	SS1:SS0		$4 f_o \rightarrow$ E	2.4576 MHz 614.4 kHz		4.0 MHz 1.0 MHz		4.9152 MHz 1.2288 MHz	
				Baud	Time	Baud	Time	Baud	Time
0	0	0	÷16	38400.0	26 μs	62500.0	16.0 μs	76800.0	13.0 μs
0	0	1	÷128	4800.0	208.3 μs	7812.5	128.0 μs	9600.0	104.2 μs
0	1	0	÷1024	600.0	1.67 ms	976.6	1.024 ms	1200.0	833.3 μs
0	1	1	÷4096	150.0	6.67 ms	244.1	4.096 ms	300.0	3.33 ms
1	0	0	÷64	9600.0	104.2 μs	15625.0	64 μs	19200.0	52.0 μs
1	0	1	÷256	2400.0	416.6 μs	3906.3	256 μs	4800.0	208.3 μs
1	1	0	÷512	1200.0	833.3 μs	1953.1	512 μs	2400.0	416.6 μs
1	1	1	÷2048	300.0	3.33 ms	488.3	2.05 ms	600.0	1.67 ms
External (P22)*				76800.0	13.0 μs	125000.0	8.0 μs	153600.0	6.5 μs

*Using maximum clock rate

Table 7. SCI Format and Clock Source Control

CC1:CC0	Format	Clock Source	Port 2 Bit 2
00	Biphase	Internal	Not Used
01	NRZ	Internal	Not Used
10	NRZ	Internal	Output
11	NRZ	External	Input

subsequently cleared. When TE is changed from clear to set, the transmitter is connected to P24 and a preamble of nine consecutive ones is transmitted. TE is cleared during reset.

Bit 2 **Transmit Interrupt Enable** — When set, an $\overline{IRQ2}$ is set; when clear, the interrupt is inhibited. TE is cleared during reset.

Bit 3 **Receive Enable** — When set, the P23 DDR bit is cleared, cannot be changed, and will remain clear if RE is subsequently cleared. While RE is set, the SCI receiver is enabled. RE is cleared during reset.

Bit 4 **Receiver Interrupt Enable** — When set, an $\overline{IRQ2}$ interrupt is enabled when RDRF and/or ORFE is set; when clear, the interrupt is inhibited. RIE is cleared during reset.

Bit 5 **Transmit Data Register Empty** — TDRE is set when the transmit data register is transferred to the output serial shift register or during reset. It is cleared by reading the TRCSR (with TDRE set) and then writing to the transmit data register. Additional data will be transmitted only if TDRE has been cleared.

Bit 6 **Overrun Framing Error** — If set, ORFE indicates either an overrun or framing error. An overrun is a new byte ready to transfer to the receiver data register with RDRF still set. A receiver framing error has occurred when the byte boundaries of the bit stream are not synchronized to the bit counter. An overrun can be distinguished from a framing error by the state of RDRF: if RDRF is set, then an overrun has occurred; otherwise, a framing error has been detected. Data are not transferred to the receive data register in an overrun condition. Unframed data causing a framing error are transferred to the receive data register. However, subsequent data transfer is blocked until the framing error flag is cleared. ORFE is cleared by reading the TRCSR (with ORFE set) then the receive data register, or during reset.

Bit 7 **Receive Data Register Full** — RDRF is set when the input serial shift register is transferred to the receive data register, or during reset.

SERIAL OPERATIONS

The SCI is initialized by writing control bytes first to the rate and mode control register and then to the transmit/receive control and status register. When TE is set, the output of the transmit serial shift register is connected to P24, and serial output is initiated by transmitting a 9-bit preamble of ones.

At this point, one of two situations exists: 1) if the transmit data register is empty (TDRE = 1), a continuous string of ones will be sent indicating an idle line; or 2) if a byte has been written to the transmit data register (TDRE = 0), it will be transferred to the output serial shift register (synchronized with the bit rate clock), TDRE will be set, and transmission will begin.

The start bit (0), eight data bits (beginning with bit 0), and a stop bit (1) will be transmitted. If TDRE is still set when the next byte transfer occurs, ones will be sent until more data is provided. In biphase format, the output toggles at the start of each bit and at half-bit time when a one is sent. Receive operation is controlled by RE, which configures P23 as an input and enables the receiver. SCI data formats are illustrated in Figure 23.

INSTRUCTION SET

The MC6801U4/MC6803U4 is directly source compatible with the MC6801 and upward source and object code compatible with the MC6800. Execution times of key instructions have been reduced, and several instructions have been added, including a hardware multiply. A list

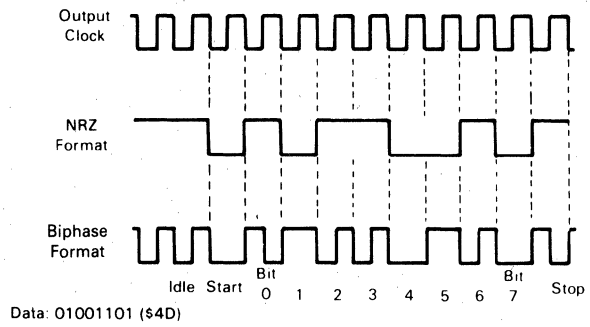


Figure 23. SCI Data Formats

of new operations added to the MC6800 instruction set is shown in Table 1.

In addition, two special opcodes, 4E and 5E, are provided for test purposes. These opcodes force the program counter to increment like a 16-bit counter, causing address lines used in the expanded modes to increment until the device is reset. These opcodes have no mnemonics.

The coding of the first (or only) byte corresponding to an executable instruction is sufficient to identify the instruction and the addressing mode. The hexadecimal equivalents of the binary codes, which result from the translation of the 82 instructions in all valid modes of addressing, are shown in Table 8. There are 220 valid machine codes, 34 unassigned codes, and 2 codes reserved for test purposes.

PROGRAMMING MODEL

A programming model for the MC6801U4/MC6803U4 is shown in Figure 8. Accumulator A can be concatenated with accumulator B and jointly referred to as accumulator D where A is the most-significant byte. Any operation that modifies the double accumulator will also modify accumulators A and/or B. Other registers are defined as follows:

Program Counter

The program counter is a 16-bit register which always points to the next instruction.

Stack Pointer

The stack pointer is a 16-bit register which contains the address of the next available location in a pushdown/pullup (LIFO) queue. The stack resides in random-access memory at a location defined by the programmer.

Index Register

The index register is a 16-bit register that can be used to store data or provide an address for the indexed mode of addressing.

Accumulators

The MPU contains two 8-bit accumulators, A and B, which are used to store operands and results from the arithmetic logic unit (ALU). They can also be concatenated and referred to as the D (double) accumulator.

Condition Code Register

The condition code register indicates the results of an instruction and includes the following five condition bits: negative (N), zero (Z), overflow (V), carry/borrow from MSB (C), and half carry from bit 3 (H). These bits are testable by the conditional branch instructions. Bit 4 is the interrupt mask (I bit) and inhibits all maskable interrupts when set. The two unused bits, B6 and B7, are read as ones.

ADDRESSING MODES

Six addressing modes can be used to reference memory. A summary of addressing modes for all instructions is presented in Table 9, 10, 11, and 12; execution times are provided in E cycles. Instruction execution times are summarized in Table 13. With an input frequency of 4 MHz, one E cycle is equivalent to one microsecond. A cycle-by-cycle description of bus activity for each instruction is provided in Table 14; descriptions of selected instructions are shown in Figure 24.

Immediate Addressing

The operand or "immediate byte(s)" is contained in the following byte(s) of the instruction where the number of bytes matches the size of the register. These are two- or three-byte instructions.

Direct Address

The least-significant byte of the operand address is contained in the second byte of the instruction, and the most-significant byte is assumed to be $00. Direct addressing allows the user to access $00 through $FF, using two-byte instructions, and execution time is reduced by eliminating the additional memory access. In most applications, the 256-byte area is reserved for frequently referenced data.

Extended Addressing

The second and third bytes of the instruction contain the absolute address of the operand. These are three-byte instructions.

Table 8. CPU Instruction Map

OP	MNEM	MODE	~	#	OP	MNEM	MODE	~	#	OP	MNEM	MODE	~	#	OP	MNEM	MODE	~	#	OP	MNEM	MODE	~	#
00	•				34	DES	INHER	3	1	68	ASL	INDXD	6	2	9C	CPX	DIR	5	2	D0	SUBB	DIR	3	2
01	NOP	INHER	2	1	35	TXS		3	1	69	ROL		6	2	9D	JSR		5	2	D1	CMPB		3	2
02	•				36	PSHA		3	1	6A	DEC		6	2	9E	LDS		4	2	D2	SBCB		3	2
03	•				37	PSHB		3	1	6B	•				9F	STS	DIR	4	2	D3	ADDD		5	2
04	LSRD		3	1	38	PULX		5	1	6C	INC		6	2	A0	SUBA	INDXD	4	2	D4	ANDB		3	2
05	ASLD		3	1	39	RTS		5	1	6D	TST		6	2	A1	CMPA		4	2	D5	BITB		3	2
06	TAP		2	1	3A	ABX		3	1	6E	JMP		3	2	A2	SBCA		4	2	D6	LDAB		3	2
07	TPA		2	1	3B	RTI		10	1	6F	CLR	INDXD	6	2	A3	SUBD		6	2	D7	STAB		3	2
08	INX		3	1	3C	PSHX		4	1	70	NEG	EXTND	6	3	A4	ANDA		4	2	D8	EORB		3	2
09	DEX		3	1	3D	MUL		10	1	71	•				A5	BITA		4	2	D9	ADCB		3	2
0A	CLV		2	1	3E	WAI		9	1	72	•				A6	LDAA		4	2	DA	ORAB		3	2
0B	SEV		2	1	3F	SWI		12	1	73	COM		6	3	A7	STAA		4	2	DB	ADDB		3	2
0C	CLC		2	1	40	NEGA		2	1	74	LSR		6	3	A8	EORA		4	2	DC	LDD		4	2
0D	SEC		2	1	41	•				75	•				A9	ADCA		4	2	DD	STD		4	2
0E	CLI		2	1	42	•				76	ROR		6	3	AA	ORAA		4	2	DE	LDX		4	2
0F	SEI		2	1	43	COMA		2	1	77	ASR		6	3	AB	ADDA		4	2	DF	STX	DIR	4	2
10	SBA		2	1	44	LSRA		2	1	78	ASL		6	3	AC	CPX		6	2	E0	SUBB	INDXD	4	2
11	CBA		2	1	45	•				79	ROL		6	3	AD	JSR		6	2	E1	CMPB		4	2
12	•				46	RORA		2	1	7A	DEC		6	3	AE	LDS		6	2	E2	SBCB		4	2
13	•				47	ASRA		2	1	7B	•				AF	STS	INDXD	5	2	E3	ADDD		6	2
14	•				48	ASLA		2	1	7C	INC		6	3	B0	SUBA	EXTND	4	3	E4	ANDB		4	2
15	•				49	ROLA		2	1	7D	TST		6	3	B1	CMPA		4	3	E5	BITB		4	2
16	TAB		2	1	4A	DECA		2	1	7E	JMP		3	3	B2	SBCA		4	3	E6	LDAB		4	2
17	TBA		2	1	4B	•				7F	CLR	EXTND	6	3	B3	SUBD		6	3	E7	STAB		4	2
18	•				4C	INCA		2	1	80	SUBA	IMMED	2	2	B4	ANDA		4	3	E8	EORB		4	2
19	DAA	INHER	2	1	4D	TSTA		2	1	81	CMPA		2	2	B5	BITA		4	3	E9	ADCB		4	2
1A	•				4E	T				82	SBCA		2	2	B6	LDAA		4	3	EA	ORAB		4	2
1B	ABA	INHER	2	1	4F	CLRA		2	1	83	SUBD		4	3	B7	STAA		4	3	EB	ADDB		4	2
1C	•				50	NEGB		2	1	84	ANDA		2	2	B8	EORA		4	3	EC	LDD		5	2
1D	•				51	•				85	BITA		2	2	B9	ADCA		4	3	ED	STD		5	2
1E	•				52	•				86	LDAA		2	2	BA	ORAA		4	3	EE	LDX		5	2
1F	•				53	COMB		2	1	87	•				BB	ADDA		4	3	EF	STX	INDXD	5	2
20	BRA	REL	3	2	54	LSRB		2	1	88	EORA		2	2	BC	CPX		6	3	F0	SUBB	EXTND	4	3
21	BRN		3	2	55	•				89	ADCA		2	2	BD	JSR		6	3	F1	CMPB		4	3
22	BHI		3	2	56	RORB		2	1	8A	ORAA		2	2	BE	LDS		5	3	F2	SBCB		4	3
23	BLS		3	2	57	ASRB		2	1	8B	ADDA		2	2	BF	STS	EXTND	5	3	F3	ADDD		6	3
24	BCC		3	2	58	ASLB		2	1	8C	CPX	IMMED	4	3	C0	SUBB	IMMED	2	2	F4	ANDB		4	3
25	BCS		3	2	59	ROLB		2	1	8D	BSR	REL	6	2	C1	CMPB		2	2	F5	BITB		4	3
26	BNE		3	2	5A	DECB		2	1	8E	LDS	IMMED	3	3	C2	SBCB		2	2	F6	LDAB		4	3
27	BEQ		3	2	5B	•				8F	•				C3	ADDD		4	3	F7	STAB		4	3
28	BVC		3	2	5C	INCB		2	1	90	SUBA	DIR	3	2	C4	ANDB		2	2	F8	EORB		4	3
29	BVS		3	2	5D	TSTB		2	1	91	CMPA		3	2	C5	BITB		2	2	F9	ADCB		4	3
2A	BPL		3	2	5E	T				92	SBCA		3	2	C6	LDAB		2	2	FA	ORAB		4	3
2B	BMI		3	2	5F	CLRB	INHER	2	1	93	SUBD		5	2	C7	•				FB	ADDB		4	3
2C	BGE		3	2	60	NEG	INDXD	6	2	94	ANDA		3	2	C8	EORB		2	2	FC	LDD		5	3
2D	BLT		3	2	61	•				95	BITA		3	2	C9	ADCB		2	2	FD	STD		5	3
2E	BGT		3	2	62	•				96	LDAA		3	2	CA	ORAB		2	2	FE	LDS		5	3
2F	BLE	REL	3	2	63	COM		6	2	97	STAA		3	2	CB	ADDB		2	2	FF		EXTND	5	3
30	TSX	INHER	3	1	64	LSR		6	2	98	EORA		3	2	CC	LDD		3	3					
31	INS		3	1	65	•				99	ADCA		3	2	CD	•				*	UEFINED OP CODE			
32	PULA		4	1	66	ROR		6	2	9A	ORAA		3	2	CE	LDX	IMMED	3	3					
33	PULB		4	1	67	ASR	INDXD	6	2	9B	ADDA		3	2	CF	•								

NOTES:

1. Addressing Modes:
   INHER = Inherent INDXD = Indexed IMMED = Immediate
   REL = Relative EXTND = Extended DIR = Direct
2. Unassigned opcodes are indicated by "•" and should not be executed.
3. Codes marked by "T" force the PC to function as a 16-bit counter.

Indexed Addressing

The unsigned offset contained in the second byte of the instruction is added with carry to the index register and is used to reference memory without changing the index register. These are two-byte instructions.

Inherent Addressing

The operand(s) is a register, and no memory reference is required. These are single-byte instructions.

Relative Addressing

Relative addressing is used only for branch instructions. If the branch condition is true, the program counter is overwritten with the sum of signed single-byte displacement in the second byte of the instruction and the current program counter. This provides a branch range of −126 to +129 bytes from the first byte of the instruction. These are two-byte instructions.

SUMMARY OF CYCLE-BY-CYCLE OPERATION

Table 14 provides a detailed description of the information present on the address bus, data bus, and the read/write (R/$\overline{W}$) line during each cycle of instruction.

The information is useful in comparing actual results with expected results during debug of both software and

Table 9. Index Register and Stack Manipulation Instructions

Pointer Operations	MNEM	Immed Op	Immed ~	Immed #	Direct Op	Direct ~	Direct #	Index Op	Index ~	Index #	Extnd Op	Extnd ~	Extnd #	Inherent Op	Inherent ~	Inherent #	Boolean/ Arithmetic Operation	Condition Codes 5 H	4 I	3 N	2 Z	1 V	0 C
Compare Index Register	CPX	8C	4	3	9C	5	2	AC	6	2	BC	6	3				X − M:M + 1	•	•	↕	↕	↕	↕
Decrement Index Register	DEX													09	3	1	X − 1 → X	•	•	•	↕	•	•
Decrement Stack Pointer	DES													34	3	1	SP − 1 → SP	•	•	•	•	•	•
Increment Index Register	INX													08	3	1	X + 1 → X	•	•	•	↕	•	•
Increment Stack Pointer	INS													31	3	1	1 SP + 1 → SP	•	•	•	•	•	•
Load Index Register	LDX	CE	3	3	DE	4	2	EE	5	2	FE	5	3				M → X_H, (M + 1) → X_L	•	•	↕	↕	R	•
Load Stack Pointer	LDS	8E	3	3	9E	4	2	AE	5	2	BE	5	3				M → SP_H, (M + 1) → SP_L	•	•	↕	↕	R	•
Store Index Register	STX				DF	4	2	EF	5	2	FF	5	3				X_H → M, X_L → (M + 1)	•	•	↕	↕	R	•
Store Stack Pointer	STS				9F	4	2	AF	5	2	BF	5	3				SP_H → M, SP_L → (M + 1)	•	•	↕	↕	R	•
Index Reg → Stack Pointer	TXS													35	3	1	X − 1 → SP	•	•	•	•	•	•
Stack Pntr → Index Register	TSX													30	3	1	SP + 1 → X	•	•	•	•	•	•
Add	ABX													3A	3	1	B + X → X	•	•	•	•	•	•
Push Data	PSHX													3C	4	1	X_L → M_{SP}, SP − 1 → SP X_H → M_{SP}, SP − 1 → SP	•	•	•	•	•	•
Pull Data	PULX													38	5	1	SP + 1 → SP, M_{SP} → X_H SP + 1 → SP, M_{SP} → X_L	•	•	•	•	•	•

3

Table 10. Accumulator and Memory Instructions (Sheet 1 of 2)

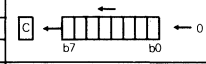
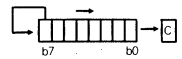
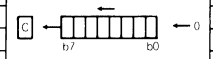
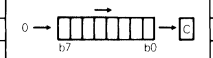
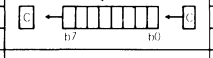
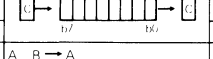
Accumulator and Memory Operations	MNEM	Immed Op	Immed ~	Immed #	Direct Op	Direct ~	Direct #	Index Op	Index ~	Index #	Extend Op	Extend ~	Extend #	Inher Op	Inher ~	Inher #	Boolean Expression	Condition Codes 5 H	4 I	3 N	2 Z	1 V	0 C
Add Accumulators	ABA													1B	2	1	A + B → A	↕	•	↕	↕	↕	↕
Add B to X	ABX													3A	3	1	00:B + X → X	•	•	•	•	•	•
Add with Carry	ADCA	89	2	2	99	3	2	A9	4	2	B9	4	3				A + M + C → A	↕	•	↕	↕	↕	↕
	ADCB	C9	2	2	D9	3	2	E9	4	2	F9	4	3				B + M + C → B	↕	•	↕	↕	↕	↕
Add	ADDA	8B	2	2	9B	3	2	AB	4	2	BB	4	3				A + M → A	↕	•	↕	↕	↕	↕
	ADDB	CB	2	2	DB	3	2	EB	4	2	FB	4	3				B + M → A	↕	•	↕	↕	↕	↕
Add Double	ADDD	C3	4	3	D3	5	2	E3	6	2	F3	6	3				D + M:M + 1 → D	•	•	↕	↕	↕	↕
And	ANDA	84	2	2	94	3	2	A4	4	2	B4	4	3				A•M → A	•	•	↕	↕	R	•
	ANDB	C4	2	2	D4	3	2	E4	4	2	F4	4	3				B•M → B	•	•	↕	↕	R	•
Shift Left, Arithmetic	ASL							68	6	2	78	6	3					•	•	↕	↕	↕	↕
	ASLA													48	2	1		•	•	↕	↕	↕	↕
	ASLB													58	2	1		•	•	↕	↕	↕	↕
Shift Left Double	ASLD													05	3	1		•	•	↕	↕	↕	↕
Shift Right, Arithmetic	ASR							67	6	2	77	6	3					•	•	↕	↕	↕	↕
	ASRA													47	2	1		•	•	↕	↕	↕	↕
	ASRB													57	2	1		•	•	↕	↕	↕	↕
Bit Test	BITA	85	2	2	95	3	2	A5	4	2	B5	4	3				A•M	•	•	↕	↕	R	•
	BITB	C5	2	2	D5	3	2	E5	4	2	F5	4	3				B•M	•	•	↕	↕	R	•
Compare Accumulators	CBA													11	2	1	A − B	•	•	↕	↕	↕	↕
Clear	CLR							6F	6	2	7F	6	3				00 → M	•	•	R	S	R	R
	CLRA													4F	2	1	00 → A	•	•	R	S	R	R
	CLRB													5F	2	1	00 → B	•	•	R	S	R	R
Compare	CMPA	81	2	2	91	3	2	A1	4	2	B1	4	3				A − M	•	•	↕	↕	↕	↕
	CMPB	C1	2	2	D1	3	2	E1	4	2	F1	4	3				B − M	•	•	↕	↕	↕	↕
1's Complement	COM							63	6	2	73	6	3				M → M	•	•	↕	↕	R	S
	COMA													43	2	1	A → A	•	•	↕	↕	R	S
	COMB													53	2	1	B → B	•	•	↕	↕	R	S

Table 10. Accumulator and Memory Instructions (Sheet 2 of 2)

Accumulator and Memory Operations		Immed			Direct			Index			Extend			Inher			Boolean Expression	Condition Codes					
																		5	4	3	2	1	0
	MNEM	Op	~	#	Op	~	#	Op	~	#	Op	~	#	Op	~	#		H	I	N	Z	V	C
Decimal Adjust, A	DAA													19	2	1	Adj binary sum to BCD	•	•	↕	↕	↕	↕
Decrement	DEC							6A	6	2	7A	6	3				M – 1 → M	•	•	↕	↕	↕	•
	DECA													4A	2	1	A – 1 → A	•	•	↕	↕	↕	•
	DECB													5A	2	1	B – 1 → B	•	•	↕	↕	↕	•
Exclusive OR	EORA	88	2	2	98	3	2	A8	4	2	B8	4	3				A ⊕ M → A	•	•	↕	↕	R	•
	EORB	C8	2	2	D8	3	2	E8	4	2	F8	4	3				B ⊕ M → B	•	•	↕	↕	R	•
Increment	INC							6C	6	2	7C	6	3				M + 1 → M	•	•	↕	↕	↕	•
	INCA													4C	2	1	A + 1 → A	•	•	↕	↕	↕	•
	INCB													5C	2	1	B + 1 → B	•	•	↕	↕	↕	•
Load Accumulators	LDAA	86	2	2	96	3	2	A6	4	2	B6	4	3				M → A	•	•	↕	↕	R	•
	LDAB	C6	2	2	D6	3	2	E6	4	2	F6	4	3				M → B	•	•	↕	↕	R	•
Load Double	LDD	CC	3	3	DC	4	2	EC	5	2	FC	5	3				M:M + 1 → D	•	•	↕	↕	R	•
Logical Shift, Left	LSL							68	6	2	78	6	3					•	•	↕	↕	↕	↕
	LSLA													48	2	1		•	•	↕	↕	↕	↕
	LSLB													58	2	1		•	•	↕	↕	↕	↕
	LSLD													05	3	2		•	•	↕	↕	↕	↕
Shift Right, Logical	LSR							64	6	2	74	6	3					•	•	R	↕	↕	↕
	LSRA													44	2	1		•	•	R	↕	↕	↕
	LSRB													54	2	1		•	•	R	↕	↕	↕
	LSRD													04	3	1		•	•	R	↕	↕	↕
Multiply	MUL													3D	10	1	A × B → D	•	•	•	•	•	↕
2's Complement (Negate)	NEG							60	6	2	70	6	3				00 – M → M	•	•	↕	↕	↕	↕
	NEGA													40	2	1	00 – A → A	•	•	↕	↕	↕	↕
	NEGB													50	2	1	00 – B → B	•	•	↕	↕	↕	↕
No Operation	NOP													01	2	1	PC + 1 → PC	•	•	•	•	•	•
Inclusive OR	ORAA	8A	2	2	9A	3	2	AA	4	2	BA	4	3				A + M → A	•	•	↕	↕	R	•
	ORAB	CA	2	2	DA	3	2	EA	4	2	FA	4	3				B + M → B	•	•	↕	↕	R	•
Push Data	PSHA													36	3	1	A → Stack	•	•	•	•	•	•
	PSHB													37	3	1	B → Stack	•	•	•	•	•	•
Pull Data	PULA													32	4	1	Stack → A	•	•	•	•	•	•
	PULB													33	4	1	Stack → B	•	•	•	•	•	•
Rotate Left	ROL							69	6	2	79	6	3					•	•	↕	↕	↕	↕
	ROLA													49	2	1		•	•	↕	↕	↕	↕
	ROLB													59	2	1		•	•	↕	↕	↕	↕
Rotate Right	ROR							66	6	2	76	6	3					•	•	↕	↕	↕	↕
	RORA													46	2	1		•	•	↕	↕	↕	↕
	RORB													56	2	1		•	•	↕	↕	↕	↕
Subtract Accumulator	SBA													10	2	1	A B → A	•	•	↕	↕	↕	↕
Subtract with Carry	SBCA	82	2	2	92	3	2	A2	4	2	B2	4	3				A M C → A	•	•	↕	↕	↕	↕
	SBCB	C2	2	2	D2	3	2	E2	4	2	F2	4	3				B – M – C → B	•	•	↕	↕	↕	↕
Store Accumulators	STAA				97	3	2	A7	4	2	B7	4	3				A → M	•	•	↕	↕	R	•
	STAB				D7	3	2	E7	4	2	F7	4	3				B → M	•	•	↕	↕	R	•
	STD				DD	4	2	ED	5	2	FD	5	3				D → M M + 1	•	•	↕	↕	R	•
Subtract	SUBA	80	2	2	90	3	2	A0	4	2	B0	4	3				A – M → A	•	•	↕	↕	↕	↕
	SUBB	C0	2	2	D0	3	2	E0	4	2	F0	4	3				B – M → B	•	•	↕	↕	↕	↕
Subtract Double	SUBD	83	4	3	93	5	2	A3	6	2	B3	6	3				D – M M + 1 → D	•	•	↕	↕	↕	↕
Transfer Accumulator	TAB													16	2	1	A → B	•	•	↕	↕	R	•
	TBA													17	2	1	B → A	•	•	↕	↕	R	•
Test, Zero or Minus	TST							6D	6	2	7D	6	3				M – 00	•	•	↕	↕	R	R
	TSTA													4D	2	1	A – 00	•	•	↕	↕	R	R
	TSTB													5D	2	1	B – 00	•	•	↕	↕	R	R

The condition code register notes are listed after Table 12.

Table 11. Jump and Branch Instructions

Operations	MNEM	Direct			Relative			Index			Extend			Inherent			Branch Test	Condition Code Reg. 5 H	4 I	3 N	2 Z	1 V	0 C
		Op	~	#	Op	~	#	Op	~	#	Op	~	#	Op	~	#							
Branch Always	BRA				20	3	2										None	•	•	•	•	•	•
Branch Never	BRN				21	3	2										None	•	•	•	•	•	•
Branch If Carry Clear	BCC				24	3	2										C = 0	•	•	•	•	•	•
Branch If Carry Set	BCS				25	3	2										C = 1	•	•	•	•	•	•
Branch If = Zero	BEQ				27	3	2										Z = 1	•	•	•	•	•	•
Branch If ≥ Zero	BGE				2C	3	2										N ⊕ V = 0	•	•	•	•	•	•
Branch If > Zero	BGT				2E	3	2										Z + (N ⊕ V) = 0	•	•	•	•	•	•
Branch If Higher	BHI				22	3	2										C + Z = 0	•	•	•	•	•	•
Branch If Higher or Same	BHS				24	3	2										C = 0	•	•	•	•	•	•
Branch If ≤ Zero	BLE				2F	3	2										Z + (N ⊕ V) = 1	•	•	•	•	•	•
Branch If Carry Set	BLO				25	3	2										C = 1	•	•	•	•	•	•
Branch If Lower Or Same	BLS				23	3	2										C + Z = 1	•	•	•	•	•	•
Branch If < Zero	BLT				2D	3	2										N ⊕ V = 1	•	•	•	•	•	•
Branch If Minus	BMI				2B	3	2										N = 1	•	•	•	•	•	•
Branch If Not Equal Zero	BNE				26	3	2										Z = 0	•	•	•	•	•	•
Branch If Overflow Clear	BVC				28	3	2										V = 0	•	•	•	•	•	•
Branch If Overflow Set	BVS				29	3	2										V = 1	•	•	•	•	•	•
Branch If Plus	BPL				2A	3	2										N = 0	•	•	•	•	•	•
Branch To Subroutine	BSR				8D	6	2											•	•	•	•	•	•
Jump	JMP							6E	3	2	7E	3	3				See Special Operations-Figure 24	•	•	•	•	•	•
Jump To Subroutine	JSR	9D	5	2				AD	6	2	BD	6	3					•	•	•	•	•	•
No Operation	NOP													01	2	1		•	•	•	•	•	•
Return From Interrupt	RTI													3B	10	1		↕	↕	↕	↕	↕	↕
Return From Subroutine	RTS													39	5	1	See Special Operations-Figure 24	•	•	•	•	•	•
Software Interrupt	SWI													3F	12	1		•	S	•	•	•	•
Wait For Interrupt	WAI													3E	9	1		•	•	•	•	•	•

Table 12. Condition Code Register Manipulation Instructions

Operations	Inherent MNEM	Op	~	#	Boolean Operation	Condition Code Register 5 H	4 I	3 N	2 Z	1 V	0 C
Clear Carry	CLC	0C	2	1	0 → C	•	•	•	•	•	R
Clear Interrupt Mask	CLI	0E	2	1	0 → I	•	R	•	•	•	•
Clear Overflow	CLV	0A	2	1	0 → V	•	•	•	•	R	•
Set Carry	SEC	0D	2	1	1 → C	•	•	•	•	•	S
Set Interrupt Mask	SEI	0F	2	1	1 → I	•	S	•	•	•	•
Set Overflow	SEV	0B	2	1	1 → V	•	•	•	•	S	•
Accumulator A → CCR	TAP	06	2	1	A → CCR	↕	↕	↕	↕	↕	↕
CCR → Accumulator A	TPA	07	2	1	CCR → A	•	•	•	•	•	•

LEGEND

- Op Operation Code (Hexadecimal)
- ~ Number of MPU Cycles
- M_{SP} Contents of memory location pointed to by Stack Pointer
- # Number of Program Bytes
- + Arithmetic Plus
- − Arithmetic Minus
- • Boolean AND
- X Arithmetic Multiply
- + Boolean Inclusive OR
- ⊕ Boolean Exclusive OR
- M Complement of M
- → Transfer Into
- 0 Bit = Zero
- 00 Byte = Zero

CONDITION CODE SYMBOLS

- H Half-carry from bit 3
- I Interrupt mask
- N Negative (sign bit)
- Z Zero (byte)
- V Overflow, 2's complement
- C Carry/Borrow from MSB
- R Reset Always
- S Set Always
- ↕ Affected
- • Not Affected

Table 13. Instruction Execution Times in E Cycles

	ADDRESSING MODE					
	Immediate	Direct	Extended	Indexed	Inherent	Relative
ABA	●	●	●	●	2	●
ABX	●	●	●	●	3	●
ADC	2	3	4	4	●	●
ADD	2	3	4	4	●	●
ADDD	4	5	6	6	●	●
AND	2	3	4	4	●	●
ASL	●	●	6	6	2	●
ASLD	●	●	●	●	3	●
ASR	●	●	6	6	2	●
BCC	●	●	●	●	●	3
BCS	●	●	●	●	●	3
BEQ	●	●	●	●	●	3
BGE	●	●	●	●	●	3
BGT	●	●	●	●	●	3
BHI	●	●	●	●	●	3
BHS	●	●	●	●	●	3
BIT	2	3	4	4	●	●
BLE	●	●	●	●	●	3
BLO	●	●	●	●	●	3
BLS	●	●	●	●	●	3
BLT	●	●	●	●	●	3
BMI	●	●	●	●	●	3
BNE	●	●	●	●	●	3
BPL	●	●	●	●	●	3
BRA	●	●	●	●	●	3
BRN	●	●	●	●	●	3
BSR	●	●	●	●	●	6
BVC	●	●	●	●	●	3
BVS	●	●	●	●	●	3
CBA	●	●	●	●	2	●
CLC	●	●	●	●	2	●
CLI	●	●	●	●	2	●
CLR	●	●	6	6	2	●
CLV	●	●	●	●	2	●
CMP	2	3	4	4	●	●
COM	●	●	6	6	2	●
CPX	4	5	6	6	●	●
DAA	●	●	●	●	2	●
DEC	●	●	6	6	2	●
DES	●	●	●	●	3	●
DEX	●	●	●	●	3	●
EOR	2	3	4	4	●	●
INC	●	●	6	6	●	●
INS	●	●	●	●	3	●

	ADDRESSING MODE					
	Immediate	Direct	Extended	Indexed	Inherent	Relative
INX	●	●	●	●	3	●
JMP	●	●	3	3	●	●
JSR	●	5	6	6	●	●
LDA	2	3	4	4	●	●
LDD	3	4	5	5	●	●
LDS	3	4	5	5	●	●
LDX	3	4	5	5	●	●
LSL	●	●	6	6	2	●
LSLD	●	●	●	●	3	●
LSR	●	●	6	6	2	●
LSRD	●	●	●	●	3	●
MUL	●	●	●	●	10	●
NEG	●	●	6	6	2	●
NOP	●	●	●	●	2	●
ORA	2	3	4	4	●	●
PSH	●	●	●	●	3	●
PSHX	●	●	●	●	4	●
PUL	●	●	●	●	4	●
PULX	●	●	●	●	5	●
ROL	●	●	6	6	2	●
ROR	●	●	6	6	2	●
RTI	●	●	●	●	10	●
RTS	●	●	●	●	5	●
SBA	●	●	●	●	2	●
SBC	2	3	4	4	●	●
SEC	●	●	●	●	2	●
SEI	●	●	●	●	2	●
SEV	●	●	●	●	2	●
STA	●	3	4	4	●	●
STD	●	4	5	5	●	●
STS	●	4	5	5	●	●
STX	●	4	5	5	●	●
SUB	2	3	4	4	●	●
SUBD	4	5	6	6	●	●
SWI	●	●	●	●	12	●
TAB	●	●	●	●	2	●
TAP	●	●	●	●	2	●
TBA	●	●	●	●	2	●
TPA	●	●	●	●	2	●
TST	●	●	6	6	2	●
TSX	●	●	●	●	3	●
TXS	●	●	●	●	3	●
WAI	●	●	●	●	9	●

Table 14. Cycle-By-Cycle Operation (Sheet 1 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
IMMEDIATE					
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	2	1 2	Opcode Address Opcode Address + 1	1 1	Opcode Operand Data
LDS LDX LDD	3	1 2 3	Opcode Address Opcode Address + 1 Opcode Address + 2	1 1 1	Opcode Operand Data (High Order Byte) Operand Data (Low Order Byte)
CPX SUBD ADDD	4	1 2 3 4	Opcode Address Opcode Address + 1 Opcode Address + 2 Address Bus FFFF	1 1 1 1	Opcode Operand Data (High Order Byte) Operand Data (Low Order Byte) Low Byte of Restart Vector
DIRECT					
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	3	1 2 3	Opcode Address Opcode Address + 1 Address of Operand	1 1 1	Opcode Address of Operand Operand Data
STA	3	1 2 3	Opcode Address Opcode Address + 1 Destination Address	1 1 0	Opcode Destination Address Data from Accumulator
LDS LDX LDD	4	1 2 3 4	Opcode Address Opcode Address + 1 Address of Operand Operand Address + 1	1 1 1 1	Opcode Address of Operand Operand Data (High Order Byte) Operand Data (Low Order Byte)
STS STX STD	4	1 2 3 4	Opcode Address Opcode Address + 1 Address of Operand Address of Operand + 1	1 1 0 0	Opcode Address of Operand Register Data (High Order Byte) Register Data (Low Order Byte)
CPX SUBD ADDD	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Operand Address Operand Address + 1 Address Bus FFFF	1 1 1 1 1	Opcode Address of Operand Operand Data (High Order Byte) Operand Data (Low Order Byte) Low Byte of Restart Vector
JSR	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Subroutine Address Stack Pointer Stack Pointer – 1	1 1 1 0 0	Opcode Irrelevant Data First Subroutine Opcode Return Address (Low Order Byte) Return Address (High Order Byte)

Table 14. Cycle-By-Cycle Operation (Sheet 2 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
EXTENDED					
JMP	3	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Jump Address (High Order Byte)
		3	Opcode Address + 2	1	Jump Address (Low Order Byte)
ADC EOR	4	1	Opcode Address	1	Opcode
ADD LDA		2	Opcode Address + 1	1	Address of Operand
AND ORA		3	Opcode Address + 2	1	Address of Operand (Low Order Byte)
BIT SBC		4	Address of Operand	1	Operand Data
CMP SUB					
STA	4	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Destination Address (High Order Byte)
		3	Opcode Address + 2	1	Destination Address (Low Order Byte)
		4	Operand Destination Address	0	Data from Accumulator
LDS	5	1	Opcode Address	1	Opcode
LDX		2	Opcode Address + 1	1	Address of Operand (High Order Byte)
LDD		3	Opcode Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	1	Operand Data (High Order Byte)
		5	Address of Operand + 1	1	Operand Data (Low Order Byte)
STS	5	1	Opcode Address	1	Opcode
STX		2	Opcode Address + 1	1	Address of Operand (High Order Byte)
STD		3	Opcode Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	0	Operand Data (High Order Byte)
		5	Address of Operand + 1	0	Operand Data (Low Order Byte)
ASL LSR	6	1	Opcode Address	1	Opcode
ASR NEG		2	Opcode Address + 1	1	Address of Operand (High Order Byte)
CLR ROL		3	Opcode Address + 2	1	Address of Operand (Low Order Byte)
COM ROR		4	Address of Operand	1	Current Operand Data
DEC TST*		5	Address Bus FFFF	1	Low Byte of Restart Vector
INC		6	Address of Operand	0	New Operand Data
CPX	6	1	Opcode Address	1	Opcode
SUBD		2	Opcode Address + 1	1	Operand Address (High Order Byte)
ADDD		3	Opcode Address + 2	1	Operand Address (Low Order Byte)
		4	Operand Address	1	Operand Data (High Order Byte)
		5	Operand Address + 1	1	Operand Data (Low Order Byte)
		6	Address Bus FFFF	1	Low Byte of Restart Vector
JSR	6	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Address of Subroutine (High Order Byte)
		3	Opcode Address + 2	1	Address of Subroutine (Low Order Byte)
		4	Subroutine Starting Address	1	Opcode of Next Instruction
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer − 1	0	Return Address (High Order Byte)

*TST does not perform the write cycle during the sixth cycle. The sixth cycle is another address bus = $FFFF

Table 14. Cycle-By-Cycle Operation (Sheet 3 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
INDEXED					
JMP	3	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
ADC EOR	4	1	Opcode Address	1	Opcode
ADD LDA		2	Opcode Address + 1	1	Offset
AND ORA		3	Address Bus FFFF	1	Low Byte of Restart Vector
BIT SBC		4	Index Register Plus Offset	1	Operand Data
CMP SUB					
STA	4	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	0	Operand Data
LDS	5	1	Opcode Address	1	Opcode
LDX		2	Opcode Address + 1	1	Offset
LDD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	1	Operand Data (High Order Byte)
		5	Index Register Plus Offset + 1	1	Operand Data (Low Order Byte)
STS	5	1	Opcode Address	1	Opcode
STX		2	Opcode Address + 1	1	Offset
STD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	0	Operand Data (High Order Byte)
		5	Index Register Plus Offset + 1	0	Operand Data (Low Order Byte)
ASL LSR	6	1	Opcode Address	1	Opcode
ASR NEG		2	Opcode Address + 1	1	Offset
CLR ROL		3	Address Bus FFFF	1	Low Byte of Restart Vector
COM ROR		4	Index Register Plus Offset	1	Current Operand Data
DEC TST*		5	Address Bus FFFF	1	Low Byte of Restart Vector
INC		6	Index Register Plus Offset	0	New Operand Data
CPX	6	1	Opcode Address	1	Opcode
SUBD		2	Opcode Address + 1	1	Offset
ADDD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register + Offset	1	Operand Data (High Order Byte)
		5	Index Register + Offset + 1	1	Operand Data (Low Order Byte)
		6	Address Bus FFFF		Low Byte of Restart Vector
JSR	6	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register + Offset	1	First Subroutine Opcode
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer − 1	0	Return Address (High Order Byte)

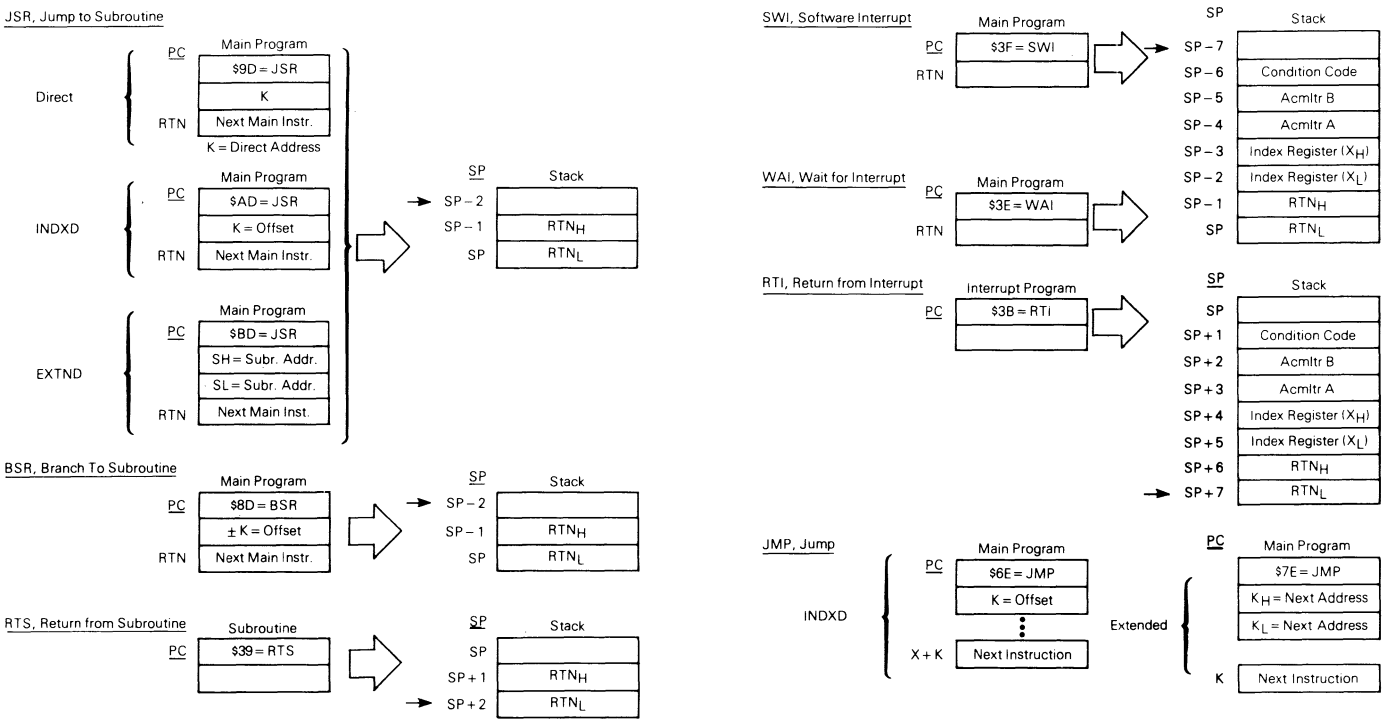
*TST does not perform the write cycle during the sixth cycle. The sixth cycle is another address bus = $FFFF.

Table 14. Cycle-By-Cycle Operation (Sheet 4 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
INHERENT					
ABA DAA SEC ASL DEC SEI ASR INC SEV CBA LSR TAB CLC NEG TAP CLI NOP TBA CLR ROL TPA CLV ROR TST COM SBA	2	1 2	Opcode Address Opcode Address + 1	1 1	Opcode Opcode of Next Instruction
ABX	3	1 2 3	Opcode Address Opcode Address + 1 Address Bus FFFF	1 1 1	Opcode Irrelevant Data Low Byte of Restart Vector
ASLD LSRD	3	1 2 3	Opcode Address Opcode Address + 1 Address Bus FFFF	1 1 1	Opcode Irrelevant Data Low Byte of Restart Vector
DES INS	3	1 2 3	Opcode Address Opcode Address + 1 Previous Stack Pointer Contents	1 1 1	Opcode Opcode of Next Instruction Irrelevant Data
INX DEX	3	1 2 3	Opcode Address Opcode Address + 1 Address Bus FFFF	1 1 1	Opcode Opcode of Next Instruction Low Byte of Restart Vector
PSHA PSHB	3	1 2 3	Opcode Address Opcode Address + 1 Stack Pointer	1 1 0	Opcode Opcode of Next Instruction Accumulator Data
TSX	3	1 2 3	Opcode Address Opcode Address + 1 Stack Pointer	1 1 1	Opcode Opcode of Next Instruction Irrelevant Data
TXS	3	1 2 3	Opcode Address Opcode Address + 1 Address Bus FFFF	1 1 1	Opcode Opcode of Next Instruction Low Byte of Restart Vector
PULA PULB	4	1 2 3 4	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer + 1	1 1 1 1	Opcode Opcode of Next Instruction Irrelevant Data Operand Data from Stack
PSHX	4	1 2 3 4	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer − 1	1 1 0 0	Opcode Irrelevant Data Index Register (Low Order Byte) Index Register (High Order Byte)
PULX	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer + 1 Stack Pointer + 2	1 1 1 1 1	Opcode Irrelevant Data Irrelevant Data Index Register (High Order Byte) Index Register (Low Order Byte)
RTS	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer + 1 Stack Pointer + 2	1 1 1 1 1	Opcode Irrelevant Data Irrelevant Data Address of Next Instruction (High Order Byte) Address of Next Instruction (Low Order Byte)
WAI	9	1 2 3 4 5 6 7 8 9	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer − 1 Stack Pointer − 2 Stack Pointer − 3 Stack Pointer − 4 Stack Pointer − 5 Stack Pointer − 6	1 1 0 0 0 0 0 0 0	Opcode Opcode of Next Instruction Return Address (Low Order Byte) Return Address (High Order Byte) Index Register (Low Order Byte) Index Register (High Order Byte) Contents of Accumulator A Contents of Accumulator B Contents of Condition Code Register

TABLE 14 — CYCLE-BY-CYCLE OPERATION (Sheet 5 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/W Line	Data Bus
INHERENT					
MUL	10	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Irrelevant Data
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Address Bus FFFF	1	Low Byte of Restart Vector
		5	Address Bus FFFF	1	Low Byte of Restart Vector
		6	Address Bus FFFF	1	Low Byte of Restart Vector
		7	Address Bus FFFF	1	Low Byte of Restart Vector
		8	Address Bus FFFF	1	Low Byte of Restart Vector
		9	Address Bus FFFF	1	Low Byte of Restart Vector
		10	Address Bus FFFF	1	Low Byte of Restart Vector
RTI	10	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Irrelevant Data
		3	Stack Pointer	1	Irrelevant Data
		4	Stack Pointer + 1	1	Contents of Condition Code Register from Stack
		5	Stack Pointer + 2	1	Contents of Accumulator B from Stack
		6	Stack Pointer + 3	1	Contents of Accumulator A from Stack
		7	Stack Pointer + 4	1	Index Register from Stack (High Order Byte)
		8	Stack Pointer + 5	1	Index Register from Stack (Low Order Byte)
		9	Stack Pointer + 6	1	Next Instruction Address from Stack (High Order Byte)
		10	Stack Pointer + 7	1	Next Instruction Address from Stack (Low Order Byte)
SWI	12	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Irrelevant Data
		3	Stack Pointer	0	Return Address (Low Order Byte)
		4	Stack Pointer − 1	0	Return Address (High Order Byte)
		5	Stack Pointer − 2	0	Index Register (Low Order Byte)
		6	Stack Pointer − 3	0	Index Register (High Order Byte)
		7	Stack Pointer − 4	0	Contents of Accumulator A
		8	Stack Pointer − 5	0	Contents of Accumulator B
		9	Stack Pointer − 6	0	Contents of Condition Code Register
		10	Stack Pointer − 7	1	Irrelevant Data
		11	Vector Address FFFA (Hex)	1	Address of Subroutine (High Order Byte)
		12	Vector Address FFFB (Hex)	1	Address of Subroutine (Low Order Byte)
RELATIVE					
BCC BHT BNE BLO BCS BLE BPL BHS BEQ BLS BRA BRN BGE BLT BVC BGT BMI BVS	3	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Branch Offset
		3	Address Buss FFFF	1	Low Byte of Restart Vector
BSR	6	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Branch Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Subroutine Starting Address	1	Opcode of Next Instruction
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer − 1	0	Return Address (High Order Byte)



Legend:
RTN = Address of next instruction in Main Program to be executed upon return from subroutine
RTN$_H$ = Most significant byte of Return Address
RTN$_L$ = Least significant byte of Return Address
→ = Stack Pointer After Execution
K = 8-bit Unsigned Value

Figure 24. Special Operations

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola using the following media:

MDOS, disk file
PC-DOS disk file (360K)
EPROM(s) Two 2516 or 2716,or a single 2532, 2732, or MC68701U4

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field-service office, sales person, or Motorola representative.

FLEXIBLE DISKS

Several types of flexible disks (MDOS™ or PC-DOS disk file) may be submitted for pattern generation. They should be programmed with the customer's program, using positive logic sense for address and data. The diskette should be clearly labeled with the customer's name, date, project or product name, and the filename containing the pattern.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to expedite the process in case of any difficulty with the pattern file.

MDOS Disk File

MDOS is Motorola's disk operating system available on the EXORciser development system. The disk media submitted must be a single-sided, single-density, 8-inch, MDOS-compatible floppy diskette. The diskette must contain the minimum set of MDOS system files in addition to the pattern file.

The .L0 output of the M6801 cross assembler should be furnished. In addition, the file must be produced using the ROLLOUT command, so that it contains the absolute image of the M6801 memory. The entire memory image of both program and data space must be included. All unused bytes, including those in the user space, must be set to logic zero.

PC-DOS Disk File

PC-DOS is IBM® personal computer disk operating system. Submitted disk media must be standard-density (360K), double-sided, 5-1/4-inch-compatible floppy diskette. The diskette must contain the object file code in Motorola's S-record format. The S-record format is a chracter-based object file format generated by M6801 cross assemblers and linkers on IBM PC-style machines.

EPROMS

Two K of EPROM are necessary to contain the entire MC6801U4 program. Two 2516 or 2716 type EPROMS, a single 2532 or 2732 type EPROM, or an MC68701U4 can be submitted for pattern generation. The EPROM is programmed with the customer program, using positive logic sense for address and data. Submissions on two EPROMs must be clearly marked. All unused bytes, including the user's space, must be set to zero.

Whether the MC6801U4 MCU ROM pattern is submitted on a single 2532 or 2732 type EPROM, an MC68701U4, or on two 2516 or 2716 type EPROMs, memory map addressing is one-for-one. When using a single 2532 or 2732 EPROM, the ROM pattern to be copied runs from EPROM address $000 to$FFF. If an MC68701U4 is used, the ROM map runs from $F000 to $FFFF. If a pair of 2516 or 2716 type EPROMs is used, then they must be clearly marked; the data-space ROM runs from EPROM address $000 to $7FF, and the program-space ROM from $7FF to $FFF.

For shipment to Motorola, EPROMs should be placed in a conductive IC carrier and packed securely. Styrofoam is not acceptable for shipment.

VERIFICATION MEDIA

All original pattern media, EPROMs or floppy disks, are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked and the verification form should be completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for the creation of the customer mask. To aid in the verification process, Motorola will program customer-supplied blank EPROM(s) or DOS disks from the data file used to create the custom mask.

ROM VERIFICATION UNITS (RVUs)

Ten MCUs containing the customer's ROM pattern will be sent for program verification. These units will have been made using the custom mask, but are for the purpose of ROM verification only. For expediency, the MCUs are unmarked, packaged in ceramic, and tested with five volts at room temperature. These RVUs are free with the minimum-order quantity, but are not production parts. These RVUs are not guaranteed by Motorola Quality Assurance.

ORDERING INFORMATION

The following table provides generic information pertaining to the package type and temperatue for the MC6801 and MC6803. These MCU devices are available in 40-pin CERDIP and plastic dual-in-line (DIP) packages.

MDOS is a trademark of Motorola Inc.
IBM is a registered trademark of International Business Machines Corporation.

MECHANICAL DATA AND ORDERING INFORMATION

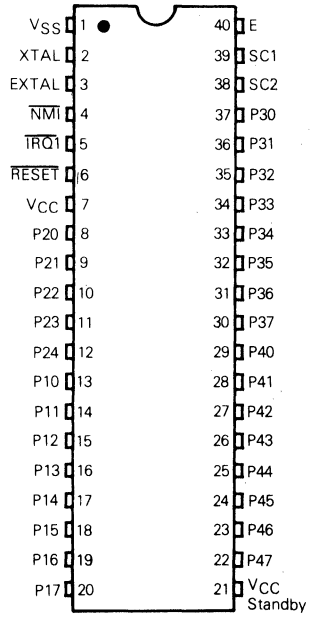
The following table provides generic information pertaining to the package type and temperatue for the MC6801 and MC6803. These MCU devices are available in 40-pin CERDIP and plastic dual-in-line (DIP) packages.

GENERIC INFORMATION

Package Type	Frequency (MHz)	Temperature	Part Number
Cerdip (S Suffix)	1.0	0° to 70°C	MC6801U4S1
	1.0	−40° to 85°C	MC6801U4CS1
	1.25	0° to 70°C	MC6801U4S1-1
	1.25	−40° to 85°C	MC6801U4CS1-1
	1.0	0° to 70°C	MC6803U4S
	1.0	−40° to 85°C	MC6803U4CS
	1.25	0° to 70°C	MC6803U4S-1
	1.25	−40° to 85°C	MC6803U4CS-1
Plastic (P Suffix)	1.0	0° to 70°C	MC6801U4P1
	1.0	−40° to 85°C	MC6801U4CP1
	1.25	0° to 70°C	MC6801U4P1-1
	1.25	−40° to 85°C	MC6801U4CP1-1
	1.0	0° to 70°C	MC6803U4P
	1.0	−40° to 85°C	MC6803U4CP
	1.25	0° to 70°C	MC6803U4P-1
	1.25	−40° to 85°C	MC6803U4CP-1

3

PIN ASSIGNMENT



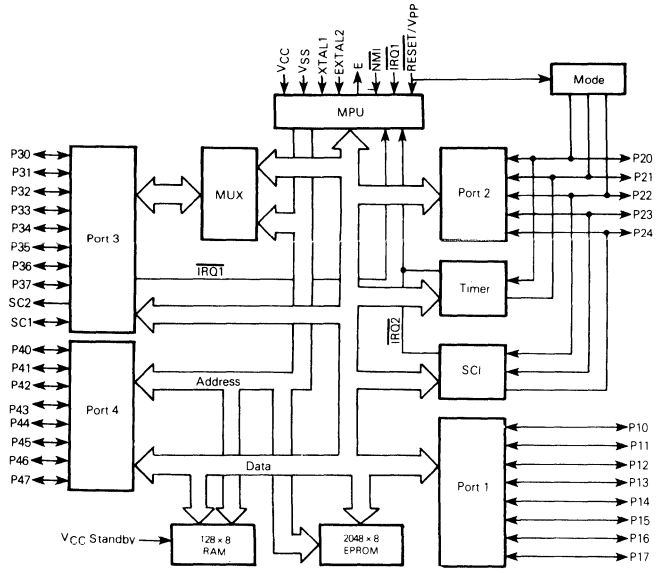
Advance Information

MC68701 Microcontroller Unit (MCU)

The MC68701 is an 8-bit single-chip EPROM microcontroller unit (MCU) which significantly enhances the capabilities of the M6800 Family of parts. It can be used in production systems to allow for easy firmware changes with minimum delay or it can be used to emulate the MC6801/MC6803 for software development. It includes an upgraded M6800 microprocessor unit (MPU) with upward source and object code compatibility. Execution times of key instructions have been improved and several new instructions have been added including an unsigned multiply. The MCU can function as a monolithic microcomputer or can be expanded to a 64K byte address space. It is TTL compatible and requires one +5 volt power supply for nonprogramming operation. An additional V_{PP} power supply is needed for EPROM programming. On-chip resoources include 2048 bytes of EPROM, 128 byte of RAM, Serial Communications Interface (SCI), parallel I/O, and a three function Programmable Timer. A summary of MCU features includes:

- Enhanced MC6800 Instruction Set
- 8×8 Multiply Instruction
- Serial Communications Interface (SCI)
- Upward Source and Object Code Compatibility with the MC6800
- 16-Bit Three-Function Programmable Timer
- Single-Chip or Expanded Operation to 64K Byte Address Space
- Bus Compatibility with the M6800 Family
- 2048 Bytes of UV Erasable, User Programmable ROM (EPROM)
- 128 Bytes of RAM (64 Bytes Retainable on Powerdown)
- 29 Parallel I/O and Two Handshake Control Lines
- Internal Clock Generator with Divide-by-Four Output
- −40 to 85°C Temperature Range

MC68701 MICROCOMPUTER BLOCK DIAGRAM



3

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range MC68701 MC68701C	T_A	T_L to T_H 0 to 70 −40 to 85	°C
Storage Temperature Range	T_{stg}	0 to 85	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high-impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \le (V_{in}$ or $V_{out}) \le V_{CC}$. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Rating
Thermal Resistance Ceramic Package Cerdip Package	θ_{JA}	 50 50	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A

CONTROL TIMING (V_{CC} = 5.0 V ±5%, V_{SS} = 0 to 70°C)

Characteristic	Symbol	MC68701		MC68701-1		MC68B701		Unit
		Min	Max	Min	Max	Min	Max	
Frequency of Operation	f_o	0.5	1.0	0.5	1.25	0.5	2.0	MHz
Crystal Frequency	f_{XTAL}	2.0	4.0	2.0	5.0	2.0	8.0	MHz
External Oscillator Frequency	$4f_o$	2.0	4.0	2.0	5.0	2.0	8.0	MHz
Crystal Oscillator Start Up Time	t_{rc}	—	100	—	100	—	100	ms
Processor Control Setup Time	t_{PCS}	200	—	170	—	110	—	ns

DC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 Vdc ±5%, V_{SS} = 0, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic		Symbol	MC68701			MC68701C			Unit
			Min	Typ	Max	Min	Typ	Max	
Input High Voltage	$\overline{RESET}$	V_{IH}	$V_{SS}+4.0$	—	V_{CC}	$V_{SS}+4.0$	—	V_{CC}	V
	Other Inputs*		$V_{SS}+2.0$	—	V_{CC}	$V_{SS}+2.2$	—	V_{CC}	
Input Low Voltage	$\overline{RESET}$	V_{IL}	$V_{SS}-0.3$	—	$V_{SS}+0.4$	$V_{SS}-0.3$	—	$V_{SS}+0.4$	V
	Other Inputs*		$V_{SS}-0.3$	—	$V_{SS}+0.8$	$V_{SS}-0.3$	—	$V_{SS}+0.8$	
Input Current, See Note (V_{in} = 0 to 2.4 V)	Port 4	I_{in}	—	—	0.6	—	—	1.0	mA
	SC1		—	—	1.0	—	—	1.6	
Input Current (V_{in} = 0 to 5.25 V)	$\overline{NMI}$, $\overline{IRQ1}$	I_{in}	—	1.5	2.5	—	1.5	5	μA
Input Current (V_{in} = 0 to 0.4 V)	$\overline{RESET}/V_{PP}$	I_{in}	—	−2.0	—	—	−2.0	—	mA
(V_{in} = 4.0 V to V_{CC})			—	—	8.0	—	—	8.0	
Hi-Z (Off State) Input Current (V_{in} = 0.5 to 2.4 V)	Ports 1, 2, and 3	I_{TSI}	—	2	10	—	2	20	μA
Output High Voltage (I_{Load} = −65 μA, V_{CC} = Min)	Port 4, SC1, SC2	V_{OH}	$V_{SS}+2.4$	—	—	$V_{SS}+2.4$	—	—	V
(I_{Load} = −100 μA, V_{CC} = Min)	Other Outputs		$V_{SS}+2.4$	—	—	$V_{SS}+2.4$	—	—	
Output Low Voltage (I_{Load} = 2.0 mA, V_{CC} = Min)	All Outputs	V_{OL}	—	—	$V_{SS}+0.5$	—	—	$V_{SS}+0.6$	V
Darlington Drive Current (V_O = 1.5 V)	Port 1	I_{OH}	1.0	2.5	10.0	1.0	2.5	10.0	mA
Internal Power Dissipation (Measured at $T_A = T_L$ in Steady-State Operation)		P_{INT}	—	—	1500	—	—	1500	mW
Input Capacitance (V_{in} = 0, T_A = 25°C, f_o = 1 MHz)	Port 3, Port 4, SC1	C_{in}	—	—	12.5	—	—	12.5	pF
	Other Inputs		—	—	10.0	—	—	10.0	
V_{CC} Standby	Powerdown	V_{SBB}	4.0	—	5.25	4.0	—	5.25	V
	Powerup	V_{SB}	4.75	—	5.25	4.75	—	5.25	
Standby Current	Powerdown	I_{SBB}	—	—	6.0	—	—	8.0	mA
Programming Time Per Byte (T_A = 25°C)		t_{PP}	25	—	50	25	—	50	ms
Programming Voltage (T_A = 25°C)		V_{PP}	20.0	21.0	22.0	20.0	21.0	22.0	V
Programming Current ($V_{RESET} = V_{PP}$, T_A = 25°C)		I_{PP}	—	30	50	—	30	50	mA

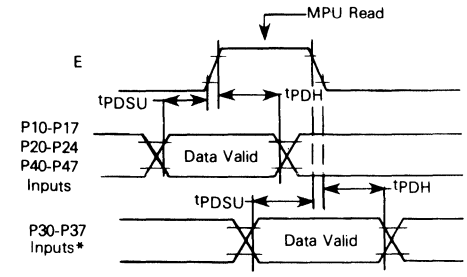
*Except mode programming levels; see Figure 15.

NOTE: $\overline{RESET}/V_{PP}$ I_{in} differs from MC6801 and MC6803 values.

PERIPHERAL PORT TIMING (Refer to Figures 3–6)

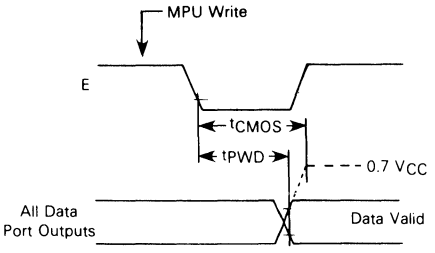
Characteristic	Symbol	MC68701		MC68701-1		MC68B701		Unit
		Min	Max	Min	Max	Min	Max	
Peripheral Data Setup Time	t_{PDSU}	200	—	200	—	100	—	ns
Peripheral Data Hold Time	t_{PDH}	200	—	200	—	100	—	ns
Delay Time, Enable Positive Transition to $\overline{OS3}$ Negative Transition	t_{OSD1}	—	350	—	350	—	250	ns
Delay Time, Enable Positive Transition to $\overline{OS3}$ Positive Transition	t_{OSD2}	—	350	—	350	—	250	ns
Delay Time, Enable Negative Transition to Peripheral Data Valid	t_{PWD}	—	350	—	350	—	250	ns
Delay Time, Enable Negative Transition to Peripheral CMOS Data Valid	t_{CMOS}	—	2.0	—	2.0	—	2.0	μs
Input Strobe Pulse Width	t_{PWIS}	200	—	200	—	100	—	ns
Input Data Hold Time	t_{IH}	50	—	50	—	30	—	ns
Input Data Setup Time	t_{IS}	20	—	20	—	20	—	ns

FIGURE 1 — DATA SETUP AND HOLD TIMES
(MPU READ)



*Port 3 Non-Latched Operation (LATCHE ENABLE = 0)

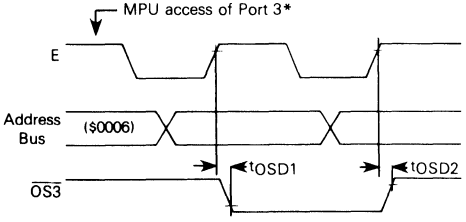
FIGURE 2 — DATA SETUP AND HOLD TIMES
(MPU WRITE)



NOTES:
1. 10 k Pullup resistor required for Port 2 to reach 0.7 V_{CC}
2. Not applicable to P21
3. Port 4 cannot be pulled above V_{CC}

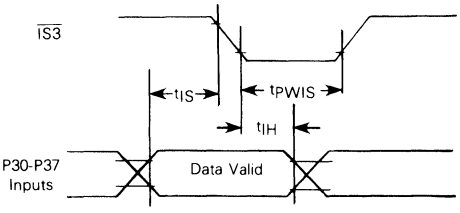
3

FIGURE 3 — PORT 3 OUTPUT STROBE TIMING
(SINGLE-CHIP MODE)



*Access matches Output Strobe Select (OSS = 0, a read; OSS = 1, a write)

FIGURE 4 — PORT 3 LATCH TIMING
(SINGLE-CHIP MODE)



NOTE: Timing measurements are referenced to a low voltage of 0.8 volts and a high voltage of 2.0 volts unless otherwise noted.

FIGURE 5 — CMOS LOAD

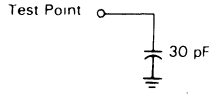
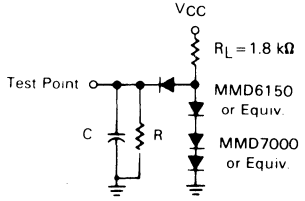


FIGURE 6 — TIMING TEST LOAD PORTS 1, 2, 3, 4



C = 90 pF for P30-P37, P40-P47, E, SC1, SC2
= 30 pF for P10-P17, P20-P24
R = 37 kΩ for P40-P47, SC1, SC2,
= 24 kΩ for P10-P17, P20-P24, P30-P37, E

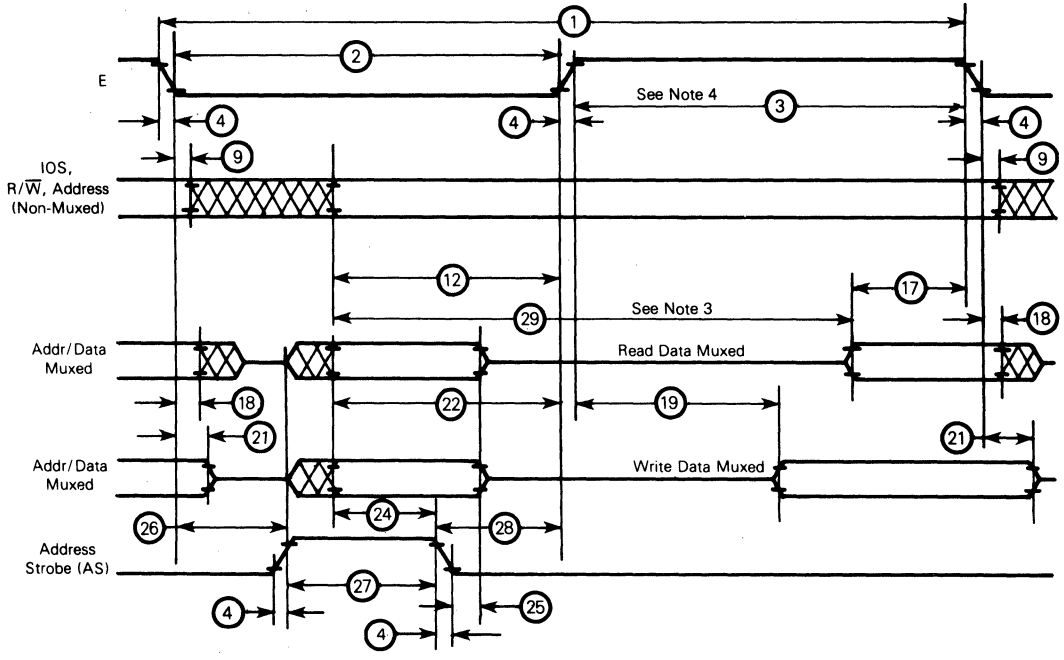
BUS TIMING (See Notes 2 and 3)

Ident Number	Characteristic	Symbol	MC68701		MC68701-1		MC68B701		Unit
			Min	Max	Min	Max	Min	Max	
1	Cycle Time	t_{cyc}	1.0	2.0	0.8	2.0	0.5	2.0	μs
2	Pulse Width, E Low	PW_{EL}	430	1000	360	1000	210	1000	ns
3	Pulse Width, E High	PW_{EH}	450	1000	360	1000	220	1000	ns
4	Clock Rise and Fall Time	t_r, t_f	—	25	—	25	—	20	ns
9	Address Hold Time	t_{AH}	20	—	20	—	10	—	ns
12	Non-Muxed Address Valid Time to E*	t_{AV}	200	—	150	—	70	—	ns
17	Read Data Setup Time	t_{DSR}	80	—	70	—	40	—	ns
18	Read Data Hold Time	t_{DHR}	10	—	10	—	10	—	ns
19	Write Data Delay Time	t_{DDW}	—	225	—	200	—	120	ns
21	Write Data Hold Time	t_{DHW}	20	—	20	—	10	—	ns
22	Multiplexed Address Valid Time to E Rise*	t_{AVM}	200	—	150	—	80	—	ns
24	Multiplexed Address Valid Time to AS Fall*	t_{ASL}	60	—	50	—	20	—	ns
25	Multiplexed Address Hold Time	t_{AHL}	20	—	20	—	10	—	ns
26	Delay Time, E to AS Rise*	t_{ASD}	90**	—	70**	—	45**	—	ns
27	Pulse Width, AS High*	PW_{ASH}	220	—	170	—	110	—	ns
28	Delay Time, AS to E Rise*	t_{ASED}	90	—	70	—	45	—	ns
29	Usable Access Time*	t_{ACC}	595	—	465	—	270	—	ns

*At specified cycle time.

**t_{ASD} parameters listed assume external TTL clock drive with 50% ±5% duty cycle. Devices driven by an external TTL clock with 50% ±1% duty cycle or which use a crystal have the following t_{ASD} specification: 100 nanoseconds minimum (1.0 MHz devices), 80 nanoseconds minimum (1.25 MHz devices), 50 nanoseconds minimum (2.0 MHz devices).

FIGURE 7 — BUS TIMING



NOTES:
1. Voltage levels shown are $V_L \leq 0.5$ V, $V_H \geq 2.4$ V, unless otherwise specified.
2. Measurement points shown are 0.8 V and 2.0 V, unless otherwise specified.
3. Usable access time is computed by 12+3−17+4.
4. Memory devices should be enabled only during E high to avoid port 3 bus contention.

INTRODUCTION

The MC68701 is an 8-bit monolithic microcomputer which can be configured to function in a wide variety of applications. The facility which provides this extraordinary flexibility is its ability to be hardware programmed into eight different operating modes. The operating mode controls the configuration of 18 of the 40 MCU pins, available on-chip resources, memory map, location (internal or external) of interrupt vectors, and type of external bus. The configuration of the remaining 22 pins is not dependent on the operating mode.

Twenty-nine pins are organized as three 8-bit ports and one 5-bit port. Each port consists of at least a Data Register and a write-only Data Direction Register. The Data Direction Register is used to define whether corresponding bits in the Data Register are configured as an input (clear) or output (set).

The term "port," by itself, refers to all of the hardware associated with the port. When the port is used as a "data port" or "I/O port," it is controlled by the port Data Direction Register and the programmer has direct access to the port pins using the port Data Register. Port pins are labled as Pij where i identifies one of four ports and j indicates the particular bit.

The Microprocessor Unit (MPU) is an enhanced MC6800 MPU with additional capabilities and greater throughput. It is upward source and object code compatible with the MC6800. The programming model is depicted in Figure 8 where Accumulator D is a concatenation of Accumulators A and B. A list of new operations added to the M6800 instruction set are shown in Table 1.

The basic difference between the MC6801 and the MC68701 is that the MC6801 has an onboard ROM while the MC68701 has an onboard EPROM. The MC68701 is pin and code compatible with the MC6801 and can be used to emulate the MC6801, allowing easy software development using the onboard EPROM. Software developed using the MC68701 can then be masked into the MC6801 ROM.

In order to support the onboard EPROM, the MC68701 differs from the MC6801 as follows:

(1) Mode 0 in the MC6801 is a test mode only, while in the MC68701 Mode 0 is also used to program the onboard EPROM and has interrupt vectors at $BFF0-$BFFF rather than $FFF0-$FFFF.
(2) The MC68701 RAM/EPROM Control Register has two bits used to control the EPROM in Mode 0 that are not defined in the MC6801 RAM Control Register.
(3) The $\overline{\text{RESET}}$/V_{PP} pin in the MC68701 is dual purpose, used to supply EPROM power as well as to reset the device; while in the MC6801 the pin is called $\overline{\text{RESET}}$ and is used only to reset the device.

3

FIGURE 8 — MC68701/6801/6803 PROGRAMMING MODEL

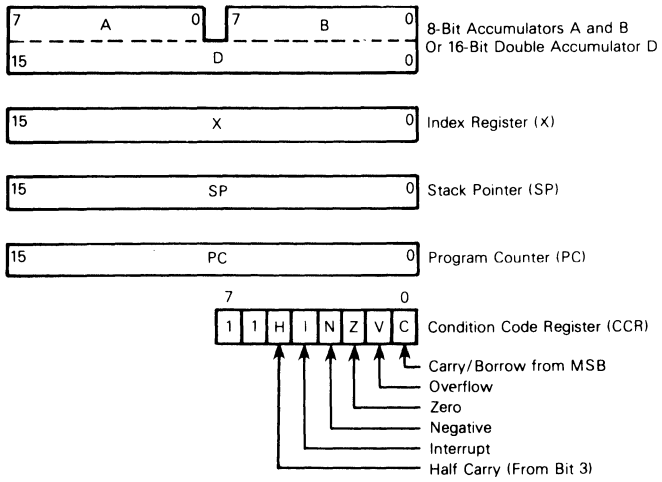


TABLE 1 — NEW INSTRUCTIONS

Instruction	Description
ABX	Unsigned addition of Accumulator B to Index Register
ADDD	Adds (without carry) the double accumulator to memory and leaves the sum in the double accumulator
ASLD or LSLD	Shifts the double accumulator left (towards MSB) one bit; the LSB is cleared and the MSB is shifted into the C-bit
BHS	Branch if Higher or Same; unsigned conditional branch (same as BCC)
BLO	Branch if Lower; Unsigned conditional branch (same as BCS)
BRN	Branch Never
JSR	Additional addressing mode: direct
LDD	Loads double accumulator from memory
LSL	Shifts memory or accumulator left (towards MSB) one bit; the LSB is cleared and the MSB is shifted into the C-bit (same as ASL)
LSRD	Shifts the double accumulator right (towards LSB) one bit; the MSB is cleared and the LSB is shifted into the C-bit
MUL	Unsigned multiply; multiplies the two accumulators and leaves the product in the double accumulator
PSHX	Pushes the Index Register to stack
PULX	Pulls the Index Register from stack
STD	Stores the double accumulator to memory
SUBD	Subtracts memory from the double accumulator and leaves the difference in the double accumulator
CPX	Internal processing modified to permit its use with any conditional branch instruction

3

OPERATING MODES

The MCU provides eight different operating modes which are selectable by hardware programming and referred to as Mode 0 through Mode 7. The operating mode controls the memory map, configuration of Port 3, Port 4, SC1, SC2, and the physical location of interrupt vectors.

FUNDAMENTAL MODES

The eight MCU modes can be grouped into three fundamental modes which refer to the type of bus it supports: Single Chip, Expanded Non-Multiplexed, and Expanded Multiplexed. Modes 4 and 7 are single chip modes. Mode 5 is the expanded non-multiplexed mode, and the remaining modes are expanded multiplexed modes. Table 2 summarizes the characteristics of the operating modes.

Single-Chip Modes (4, 7)

In the Single-Chip Mode, the four MCU ports are configured as parallel input/output data ports, as shown in Figure 9. The MCU functions as a monolithic microcomputer in these two modes without external address or data buses. A maximum of 29 I/O lines and two Port 3 control lines are provided. Peripherals or another MCU can be interfaced to Port 3 in a loosely coupled dual processor configuration, as shown in Figure 10.

In Single-Chip Test Mode (4), the RAM responds to $XX80 through $XXFF and the EPROM is removed from the internal address map. A test program must first be loaded into the RAM using modes 0, 1, 2, or 6. If the MCU is reset and then programmed into Mode 4, execution will begin at $XXFE:XXFF. Mode 5 can be irreversibly entered from Mode 4 without asserting $\overline{\text{RESET}}$ by setting bit 5 of the Port 2 Data Register. This mode is used primarily to test Ports 3 and 4 in the Single-Chip and Non-Multiplexed Modes.

TABLE 2 — SUMMARY OF MC68701 OPERATING MODES

Common to all Modes:
- Reserved Register Area
- Port 1
- Port 2
- Programmable Timer
- Serial Communications Interface

Single Chip Mode 7
- 128 bytes of RAM; 2048 bytes of EPROM
- Port 3 is a parallel I/O port with two control lines
- Port 4 is a parallel I/O port
- SC1 is Input Strobe 3 (IS3)
- SC2 is Output Strobe 3 (OS3)

Expanded Non-Multiplexed Mode 5
- 128 bytes of RAM; 2048 bytes of EPROM
- 256 bytes of external memory space
- Port 3 is an 8-bit data bus
- Port 4 is an input port/address bus
- SC1 is Input/Output Select (IOS)
- SC2 is Read/Write (R/W)

Expanded Multiplexed Modes 1, 2, 3, 6
- Four memory space options (64K address space):
  - (1) No internal RAM or EPROM (Mode 3)
  - (2) Internal RAM, no EPROM (Mode 2)
  - (3) Internal RAM and EPROM (Mode 1)
  - (4) Internal RAM, EPROM with partial address bus (Mode 6)
- Port 3 is a multiplexed address/data bus
- Port 4 is an address bus (inputs/address in Mode 6)
- SC1 is Address Strobe (AS)
- SC2 is Read/Write (R/$\overline{\text{W}}$)

Test Mode 4
- (1) May be changed to Mode 5 without going through Reset
- (2) May be used to test Ports 3 and 4 as I/O ports

Expanded Multiplexed Mode 0
- (1) Internal RAM and EPROM
- (2) External interrupt vectors located at $BFF0-$BFFF
- (3) Used to program EPROM

FIGURE 9 — SINGLE-CHIP MODE

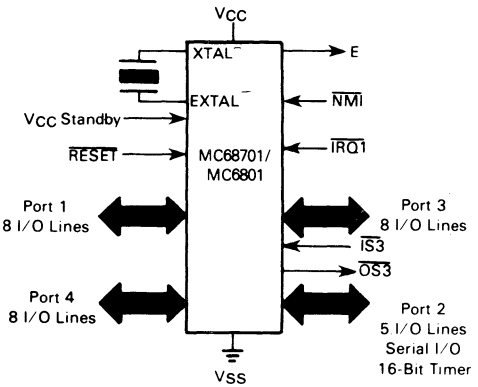


FIGURE 10 — SINGLE-CHIP DUAL PROCESSOR CONFIGURATION

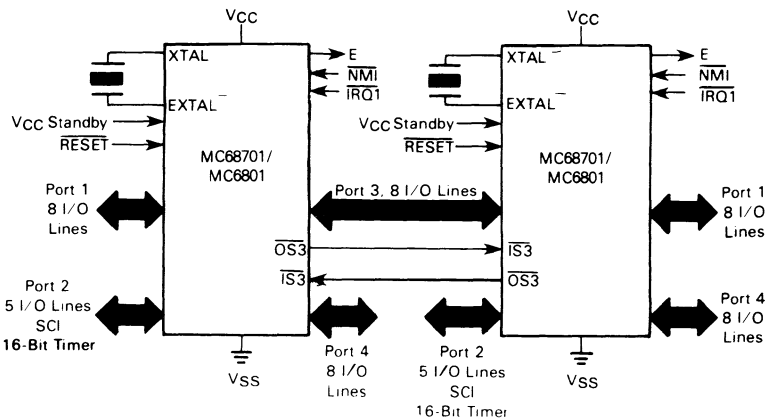
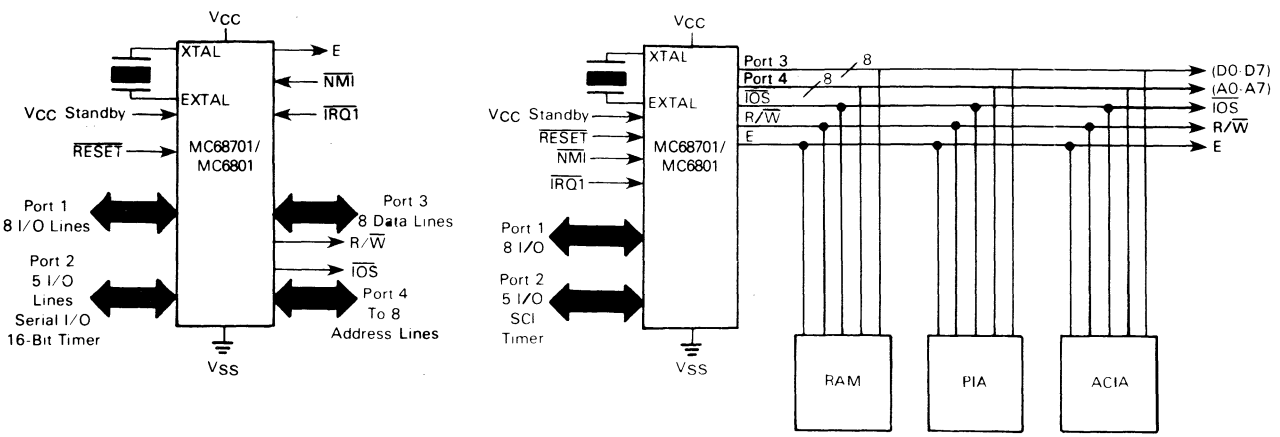


FIGURE 11 — EXPANDED NON-MULTIPLEXED CONFIGURATION



3

Expanded Non-Multiplexed Mode (5)

A modest amount of external memory spce is provided in the Expanded Non-Multiplexed Mode while significat on-chip resources are retained. Port 3 functions as an 8-bit bidirectional data bus and Port 4 is configured initially as an input data port. Any combination of the eight least-significant address lines may be obtained by writing to the Port 4 Data Direction Register. Stated alternatively, any combination of A0 to A7 may be provided while retaining the remainder as input data lines. Internal pullup resistors are intended to pull the Port 4 lines high until the port is configured.

Figure 11 illustrates a typical system configuration in the Expanded Non-Multiplexed Mode. The MCU interfaces directy with M6800 Family parts and can access 256 bytes of external address space at $100 through $1FF. IOS provides an address decode of external memory ($100-$1FF) and can be used as a memory page select or chip select line.

Expanded-Multiplexed Modes (0, 1, 2, 3, 6)

In the Expanded-Multiplexed Modes, the MCU has the ability to access a 64K bytes memory space. Port 3 functions as a time multiplexed address/data bus with address valid on the negative edge of Address Strobe (AS), and data valid while E is high. In Modes 0 to 3, Port 4 provides address lines A8 to A15. In Mode 6, however, Port 4 is initially configured at RESET as an input data port. The Port 4 Data Direction Register can then be changed to provide any combination of address lines, A8 to A15. Stated alternatively, any subset of A8 to A15 can be provided while retaining the remaining Port 4 lines as input data lines. Internal pullup resistors pull the Port 4 lines high until software configures the port.

Figure 12 depicts a typical configuration for the Expanded-Multiplexed Modes. Address Strobe can be used to control a transparent D-type latch to capture addresses A0 to A7, as shown in Figure 13. This allows Port 3 to function as a Data Bus when E is high.

In Mode 0, the internal and external data buses are connected; there must therefore be no memory map overlap in order to avoid potential bus conflicts. Mode 0 is used to program the onboard EPROM. All interrupt vectors are external in this mode and are located at $BFF0-$BFFF.

PROGRAMMING THE MODE

The operating mode is determined at RESET by the levels asserted on P22, P21, and P20. These levels are latched into PC2, PC1, and PC0 of the program control register on the positive edge of RESET. The operating mode may be read from the Port 2 Data Register as shown below, and programming levels and timing must be met as shown in Figure 14. A brief outline of the operating modes is shown in Table 3.

PORT 2 DATA REGISTER

7	6	5	4	3	2	1	0	
PC2	PC1	PC0	P24	P23	P22	P21	P20	$0003

Circuitry to provide the programming levels is dependent primarily on the normal system usage of the three pins. If configured as outputs, the circuits shown in Figure 15 may be used; otherwise, three-state buffers can be used to provide isolation while programming the mode. Note that if diodes are used to program the mode, the diode forward voltage drop must not exceed the V_{MPDD} minimum.

MEMORY MAPS

The MCU can provide up to 64K byte address space depending on the operating mode. A memory map for each operating mode is shown in Figure 16. The first 32 locations of each map are reserved for the MCU internal registers as shown in Table 4, with exceptions as indicated.

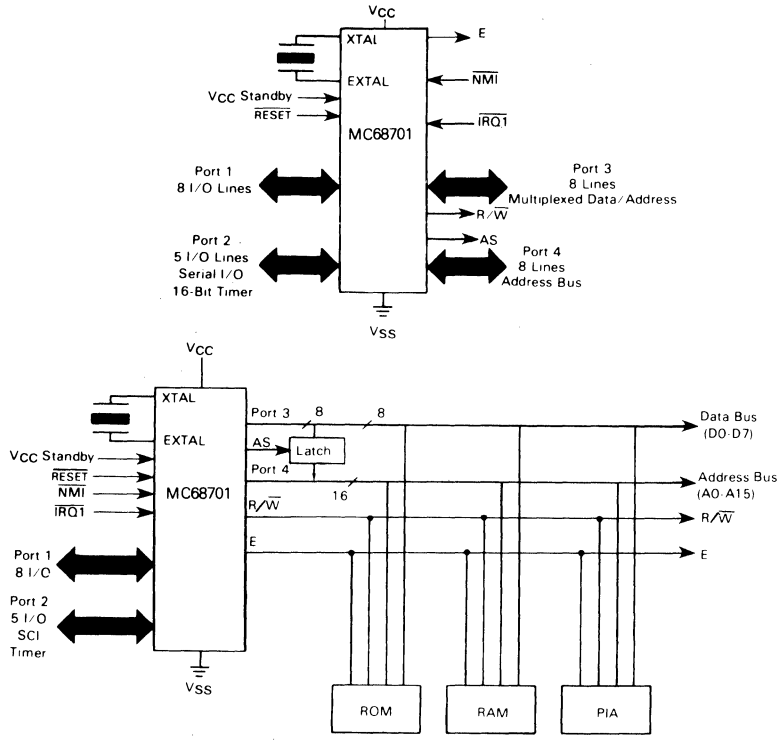
TABLE 3 — MODE SELECTION SUMMARY

Mode	P22 PC2	P21 PC1	P20 PC0	EPROM	RAM	Interrupt Vectors	Bus Mode	Operating Mode
7	H	H	H	I	I	I	I	Single Chip
6	H	H	L	I	I	I	MUX(5, 6)	Multiplexed/Partial Decode
5	H	L	H	I	I	I	NMUX(5, 6)	Non-Multiplexed/Partial Decode
4	H	L	L	I(2)	I(1)	I	I	Single Chip Test
3	L	H	H	E	E	E	MUX(4)	Multiplexed/No RAM or EPROM
2	L	H	L	E	I	E	MUX(4)	Multiplexed/RAM
1	L	L	H	I	I	E	MUX(4)	Multiplexed/RAM and EPROM
0	L	L	L	I	I	I(3)	MUX(4)	Multiplexed/Programming

Legend:
I — Internal
E — External
MUX — Multiplexed
NMUX — Non-Multiplexed
L — Logic "0"
H — Logic "1"

Notes:
(1) Internal RAM is addressed at $XX80
(2) Internal EPROM is disabled
(3) Interrupt vectors located at $BFF0-$BFFF
(4) Addresses associated with Ports 3 and 4 are considered external in Modes 0, 1, 2, and 3
(5) Addresses associated with Port 3 are considered external in Modes 5 and 6
(6) Port 4 default is user data input; address output is optional by writing to Port 4 Data Direction Register

FIGURE 12 — EXPANDED MULTIPLEXED CONFIGURATION



NOTE: To avoid data bus (Port 3) contention in the expanded multiplexed modes, memory devices should be enabled only during E high time.

FIGURE 13 — TYPICAL LATCH ARRANGEMENT

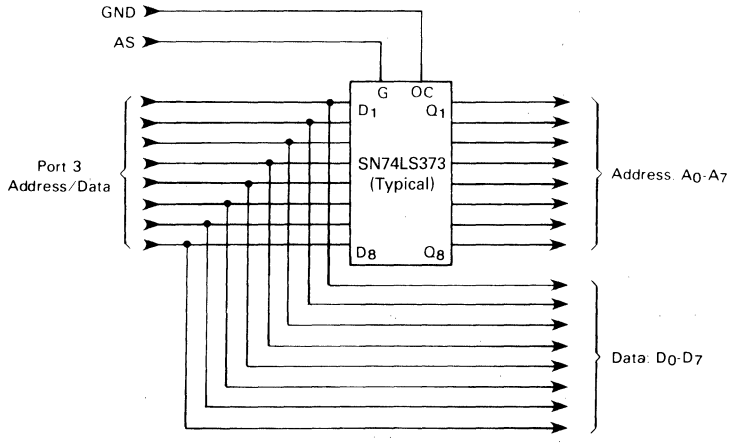
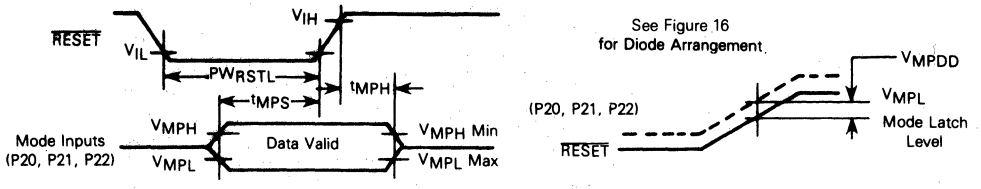


FIGURE 14 — MODE PROGRAMMING TIMING

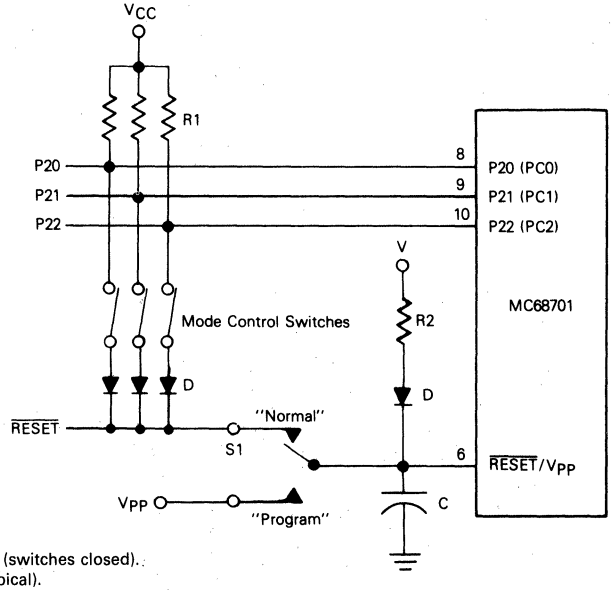


MODE PROGRAMMING (Refer to Figure 14)

Characteristic	Symbol	Min	Typ	Max	Unit
Mode Programming Input Voltage Low for T_A=0 to 70°C	V_{MPL}	—	—	1.8	V
Mode Programming Input Voltage High	V_{MPH}	4.0	—	—	V
Mode Programming Diode Differential for T_A=0 to 70°C	V_{MPDD}	0.6	—	—	V
$\overline{RESET}$ Low Pulse Width	PW_{RSTL}	3.0	—	—	E-Cycles
Mode Programming Set-Up Time	t_{MPS}	2.0	—	—	E-Cycles
Mode Programming Hold Time $\overline{RESET}$ Rise Time≥1 μs $\overline{RESET}$ Rise Time<1 μs	t_{MPH}	0 100	— —	— —	ns

Note: For T_A= −40 to 85°C, Maximum V_{MPL}=1.7, and Minimum V_{MPDD}=0.4.

FIGURE 15 — TYPICAL MODE PROGRAMMING CIRCUIT

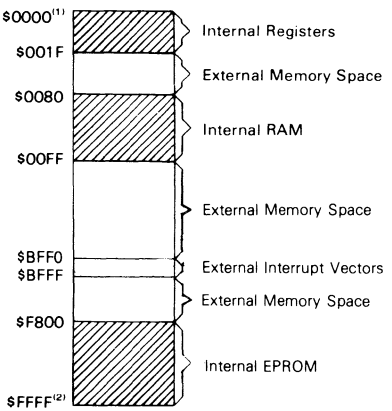


Notes:
1. Mode 0 as shown (switches closed).
2. R1 = 10k ohms (typical).
3. The $\overline{RESET}$ time constant is equal to RC where R is the equivalent parallel resistance of R2 and the number of resistors (R1) placed in the circuit by closed mode control switches.
4. D = 1N914, 1N4001 in the 0 to 70°C range
   D = 1N270, MBD201 in the −40 to 85°C range
5. If V = V_{CC}, the R2 = 50 ohms (typical) to meet V_{IH} for the $\overline{RESET}$/V_{PP} pin. V = V_{CC} is also compatible with MC6801. The $\overline{RESET}$ time constant in this case is approximately R2*C.
6. Switch S1 allows selection of normal ($\overline{RESET}$) or programming (V_{PP}) as the input to the $\overline{RESET}$/V_{PP} pin. During switching, the input level is held at a value determined by a diode (D), resistor (R2) and input voltage (V).
7. While S1 is in the "Program" position, $\overline{RESET}$ should not be asserted.
8. From powerup, $\overline{RESET}$ must be held low for at least t_{RC}. The capacitor, C, is shown for conceptual purposes only and is on the order of 1000 μF for the circuit shown. Typically, a buffer with an RC input will be used to drive $\overline{RESET}$, eliminating the need for the larger capacitor.
9. Diode V_f should not exceed V_{MPDD} min.

FIGURE 16 — MC68701 MEMORY MAPS

MC68701 Mode 0

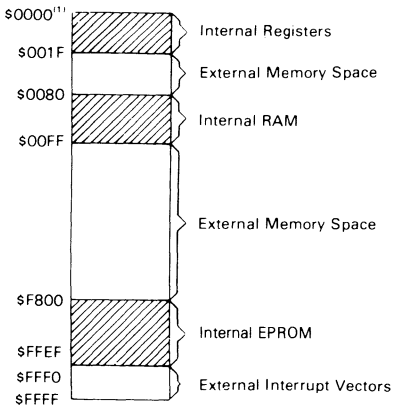
Multiplexed Test mode



Notes:
1) Excludes the following addresses which may be used externally: $04, $05, $06, $07 and $0F.
2) There must be no overlapping of internal and external memory spaces to avoid driving the data bus with more than one device.
3) This mode is used to program the onboard EPROM.

MC68701 Mode 1

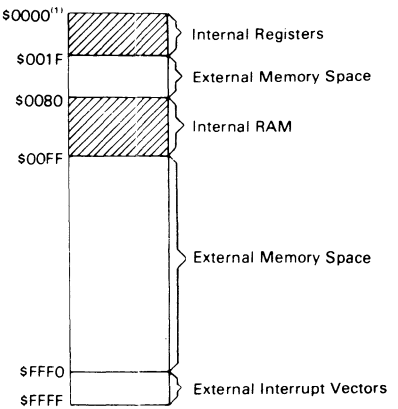
Multiplexed/RAM & EPROM



Notes:
1) Excludes the following addresses which may be used externally: $04, $05, $06, $07 and $0F
2) Internal EPROM addresses $FFF0 to $FFFF are not usable

MC68701 Mode 2

Multiplexed/RAM



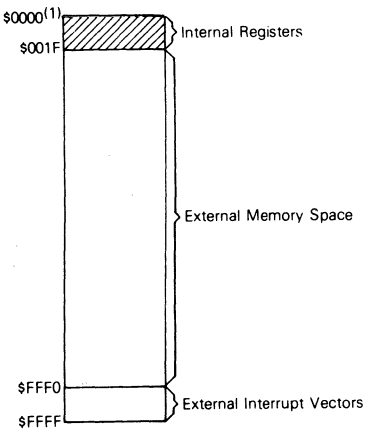
Notes:
1) Excludes the following addresses which may be used externally: $04, $05, $06, $07, and $0F.

3

FIGURE 16 – MC68701 MEMORY MAPS (CONTINUED)

MC68701 Mode 3

Multiplexed/No RAM or EPROM

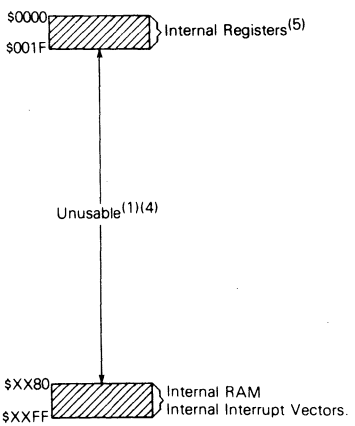


Notes:

1) Excludes the following addresses which may be used externally: $04, $05, $06, $07 and $0F.

MC68701 Mode 4

Single Chip Test

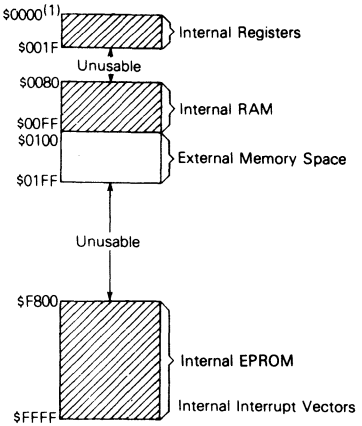


Notes:

1) The internal EPROM is disabled.
2) Mode 4 may be changed to Mode 5 without having to assert $\overline{\text{RESET}}$ by writing a "1" into the PCO bit of Port 2 Data Register.
3) Addresses A8 to A15 are treated as "don't cares" to decode internal RAM.
4) Internal RAM will appear at $XX80 to $XXFF.
5) MCU read of the Port 3 Data Direction Register will access the Port 3 Data Register.

MC68701 Mode 5

Non-Multiplexed/Partial Decode



Notes:

1) Excludes the following addresses which may **NOT** be used externally: $04, $06, and $0F (No IOS).
2) This mode may be entered without going through $\overline{\text{RESET}}$ by using Mode 4 and subsequently writing a "1" into the PCO bit of Port 2 Data Register.
3) Address lines A0 to A7 will not contain addresses until the Data Direction Register for Port 4 has been written with "1's" in the appropriate bits. These address lines will assert "1's" until made outputs by writing the Data Direction Register.

FIGURE 16 — MC68701 MEMORY MAPS (CONCLUDED)

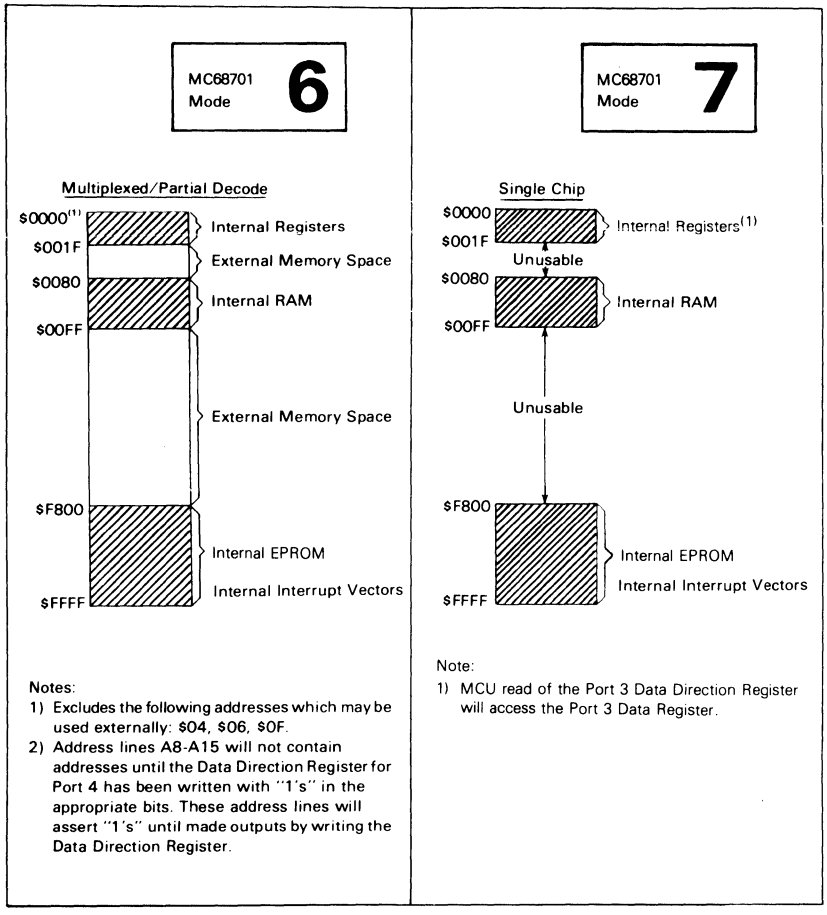


TABLE 4 — INTERNAL REGISTER AREA

Register	Address
Port 1 Data Direction Register***	00
Port 2 Data Direction Register***	01
Port 1 Data Register	02
Port 2 Data Register	03
Port 3 Data Direction Register***	04*
Port 4 Data Direction Register***	05**
Port 3 Data Register	06*
Port 4 Data Register	07**
Timer Control and Status Register	08
Counter (High Byte)	09
Counter (Low Byte)	0A
Output Compare Register (High Byte)	0B
Output Compare Register (Low Byte)	0C
Input Capture Register (High Byte)	0D
Input Capture Register (Low Byte)	0E
Port 3 Control and Status Register	0F*
Rate and Mode Control Register	10
Transmit/Receive Control and Status Register	11
Receive Data Register	12
Transmit Data Register	13
RAM/EPROM Control Register	14
Reserved	15-1F

*External addresses in Modes 0, 1, 2, 3, 5, 6; cannot be accessed in Mode 5 (No $\overline{IOS}$)
**External addresses in Modes 0, 1, 2, 3
***1 = output, 0 = Input

MC68701 INTERRUPTS

The MCU supports two types of interrupt requests: maskable and non-maskable. A Non-Maskable Interrupt ($\overline{NMI}$) is always recognized and acted upon at the completion of the current instruction. Maskable interrupts are controlled by the Condition Code Register's I-bit and by individual enable bits. The I-bit controls all maskable interrupts. Of the maskable interrupts, there are two types: $\overline{IRQ1}$ and $\overline{IRQ2}$. The Programmable Timer and Serial Communications Interface use an internal $\overline{IRQ2}$ interrupt line. External devices (and $\overline{IS3}$) use $\overline{IRQ1}$. An $\overline{IRQ1}$ interrupt is serviced before $\overline{IRQ2}$ if both are pending.

All $\overline{IRQ2}$ interrupts use hardware prioritized vectors. The single SCI interrupt and three timer interrupts are serviced in a prioritized order and each is vectored to a separate location. All MCU interrupt vector locations are shown in Table 5.

TABLE 5 — MCU INTERRUPT VECTOR LOCATIONS

Mode 0		Modes 1-7		
MSB	LSB	MSB	LSB	Interrupt
BFFE	BFFF	FFFE	FFFF	$\overline{RESET}$
BFFC	BFFD	FFFC	FFFD	$\overline{NMI}$
BFFA	BFFB	FFFA	FFFB	Software Interrupt (SWI)
BFF8	BFF9	FFF8	FFF9	$\overline{IRQ1}$ (or $\overline{IS3}$)
BFF6	BFF7	FFF6	FFF7	ICF (Input Capture)*
BFF4	BFF5	FFF4	FFF5	OCF (Output Compare)*
BFF2	BFF3	FFF2	FFF3	TOF (Timer Overflow)*
BFF0	BFF1	FFF0	FFF1	SCI(RDRF+ORFE+TDRE)*

*IRQ2 Interrupt

The Interrupt flowchart is depicted in Figure 17 and is common to every MCU interrupt excluding reset. During interrupt servicing the Program Counter, Index Register, A Accumulator, B Accumulator, and Condition Code Register are pushed to the stack. The I-bit is set to inhibit maskable interrupts and a vector is fetched corresponding to the current highest priority interrupt. The vector is transferred to the Program Counter and instruction execution is resumed. Interrupt and $\overline{RESET}$ timing are illustrated in Figures 18 and 19.

FUNCTIONAL PIN DESCRIPTIONS

V_{CC} AND V_{SS}

V_{CC} and V_{SS} provide power to a large portion of the MCU. The power supply should provide +5 volts (±5%) to V_{CC}, and V_{SS} should be tied to ground. Total power dissipation (including V_{CC} Standby), will not exceed P_D milliwatts.

V_{CC} STANDBY

V_{CC} Standby provides power to the standby portion ($80 through $BF) of the RAM and the STBY PWR and RAME bits of the RAM Control Register. Voltage requirements depend on whether the MCU is in a powerup or powerdown state. In the powerup state, the power supply should provide +5 volts (±5%) and must reach V_{SB} volts before $\overline{RESET}$ reaches 4.0 volts. During powerdown, V_{CC} Standby must remain above V_{SBB} (min) to sustain the standby RAM and STBY PWR bit. While in powerdown operation, the standby current will not exceed I_{SBB}.

It is typical to power both V_{CC} and V_{CC} Standby from the same source during normal operation. A diode must be used between them to prevent supplying power to V_{CC} during powerdown operation. V_{CC} Standby should be tied to ground in Mode 3.

XTAL1 AND EXTAL2

These two input pins interface either a crystal or TTL compatible clock to the MCU internal clock generator. Divide-by-four circuitry is included which allows use of the inexpensive 3.58 MHz or 4.4336 MHz Color Burst TV crystals. A 20 pF capacitor should be tied from each crystal pin to ground to ensure reliable startup and operation. Alternatively, EXTAL2 may be driven by an external TTL compatible clock at $4f_O$ with a duty cycle of 50% (±5%) with XTAL1 connected to ground.

The internal oscillator is designed to interface with an AT-cut quartz crystal resonator operated in parallel resonance mode in the frequency range specified for f_{XTAL}. The crystal should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time.** The MCU is compatible with most commercially available crystals. Nominal crystal parameters are shown in Figure 20.

$\overline{RESET}$/V_{PP}

This input is used to reset the MCU internal state and provide an orderly startup procedure. During powerup, $\overline{RESET}$ must be held below 0.4 volts: (1) at least t_{RC} after V_{CC} reaches 4.75 volts in order to provide sufficient time for the clock generator to stabilize, and (2) until V_{CC} Standby reaches V_{SB} volts. $\overline{RESET}$ must be held low at least three E-cycles if asserted during powerup operation.

This pin is also used to supply V_{PP} in Mode 0 for programming the EPROM, and supplies operating power to the EPROM during powerup operation.

E (ENABLE)

This is an output clock used primarily for bus synchronization. It is TTL compatible and is the slightly skewed divide-by-four result of the MCU input clock frequency. It will drive one Schottky TTL load and 90 pF, and all data given in cycles is referenced to this clock unless otherwise noted.

$\overline{NMI}$ (NON-MASKABLE INTERRUPT)

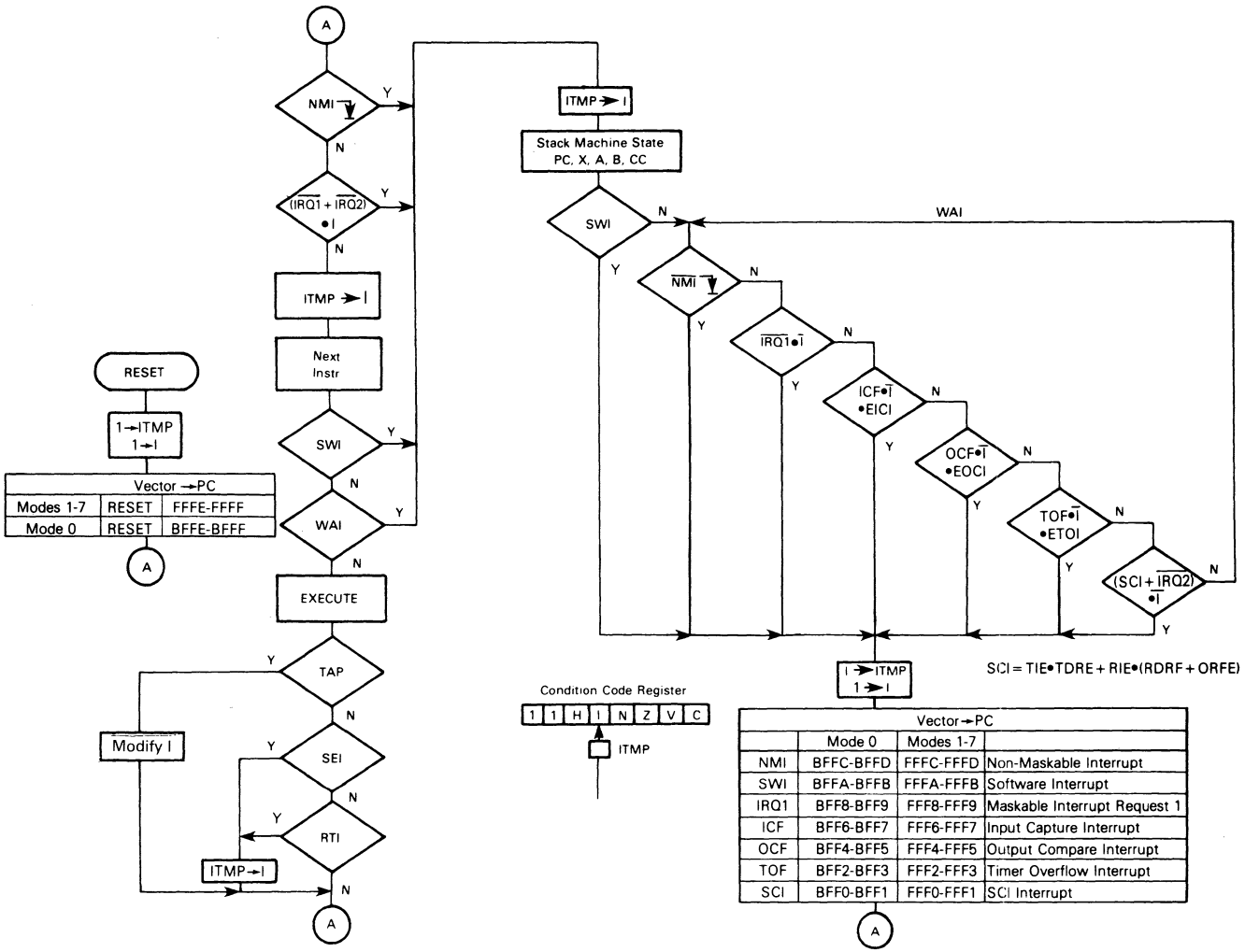
An $\overline{NMI}$ negative edge requests an MCU interrupt sequence, but the current instruction will be completed before it responds to the request. The MCU will then begin an interrupt sequence. Finally, a vector is fetched from $FFFC and $FFFD (or $BFFC and $BFFD in Mode 0), transferred to the Program Counter and instruction execution is resumed. $\overline{NMI}$ typically requires a 3.3 kΩ (nominal) resistor to V_{CC}. There is no internal $\overline{NMI}$ pullup resistor. $\overline{NMI}$ must be held low for at least one E-cycle to be recognized under all conditions.

$\overline{IRQ1}$ (MASKABLE INTERRUPT REQUEST 1)

$\overline{IRQ1}$ is a level-sensitive input which can be used to request an interrupt sequence. The MPU will complete the current instruction before it responds to the request. If the inter-

** Devices made with masks subsequent to T7A and CB4 incorporate an advanced clock with improved startup characteritics.

FIGURE 17 — INTERRUPT FLOWCHART



Vector→PC			
	Mode 0	Modes 1-7	
NMI	BFFC-BFFD	FFFC-FFFD	Non-Maskable Interrupt
SWI	BFFA-BFFB	FFFA-FFFB	Software Interrupt
IRQ1	BFF8-BFF9	FFF8-FFF9	Maskable Interrupt Request 1
ICF	BFF6-BFF7	FFF6-FFF7	Input Capture Interrupt
OCF	BFF4-BFF5	FFF4-FFF5	Output Compare Interrupt
TOF	BFF2-BFF3	FFF2-FFF3	Timer Overflow Interrupt
SCI	BFF0-BFF1	FFF0-FFF1	SCI Interrupt

FIGURE 18 — INTERRUPT SEQUENCE

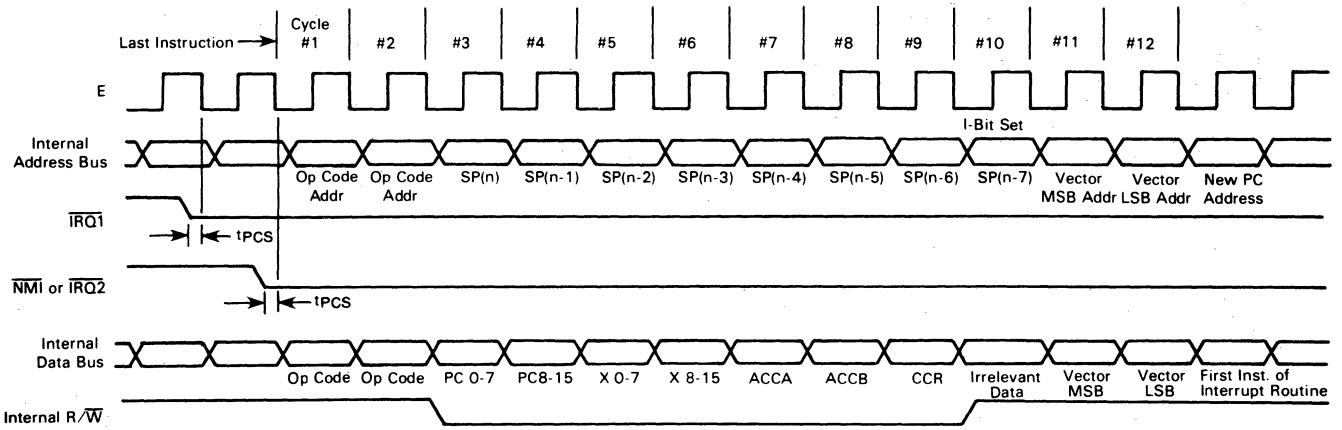
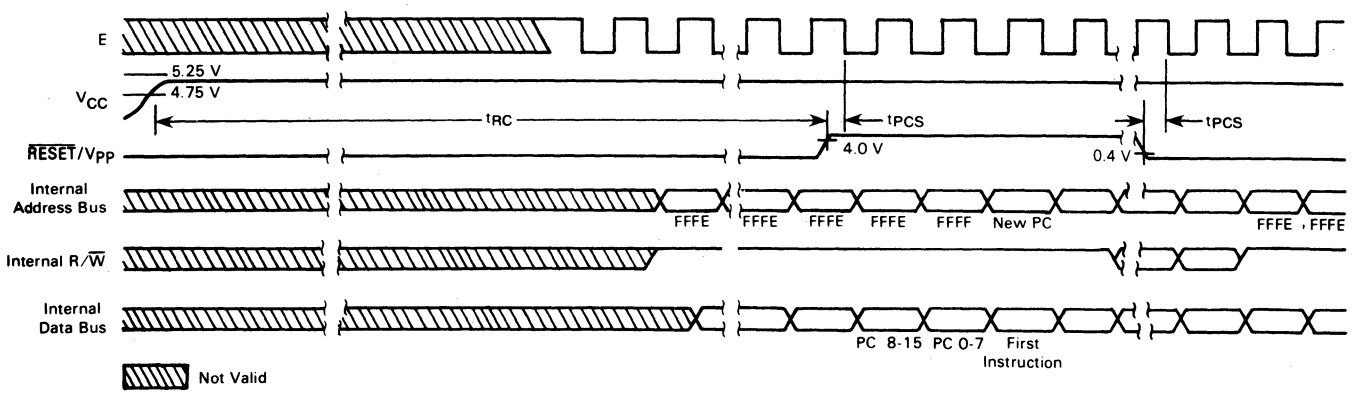


FIGURE 19 — RESET TIMING



rupt mask bit (I-bit) in the Condition Code Register is clear, the MCU will begin an interrupt sequence. A vector is fetched from $FFF8 and $FFF9 (or $BFF8 and $BFF9 in Mode 0), transferred to the Program Counter, and instruction execution is resumed.

$\overline{IRQ1}$ typically requires an external 3.3 kΩ (nominal) resistor to V_{CC} for wire-OR applications. $\overline{IRQ1}$ has no internal pullup resistor.

SC1 AND SC2 (STROBE CONTROL 1 AND 2)

The function of SC1 and SC2 depends on the operating mode. SC1 is configured as an output in all modes except single chip mode, whereas SC2 is always an output. SC1 and SC2 can drive one Schottky load and 90 pF.

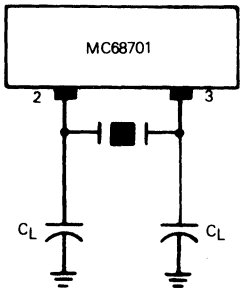
SC1 and SC2 In Single Chip Mode

In Single Chip Mode, SC1 and SC2 are configured as an input and output, respectively, and both function as Port 3 control lines. SC1 functions as $\overline{IS3}$ and can be used to indicate that Port 3 input data is ready or output data has been accepted. Three options associated with $\overline{IS3}$ are controlled by the Port 3 Control and Status Register and are discussed in the Port 3 description. If unused, $\overline{IS3}$ can remain unconnected.

SC2 is configured as $\overline{OS3}$ and can be used to strobe output data or acknowledge input data. It is controlled by Output Strobe Select (OSS) in the Port 3 Control and Status Register. The strobe is generated by a read (OSS = 0) or write (OSS = 1) to the Port 3 Data Register. $\overline{OS3}$ timing is shown in Figure 5.

FIGURE 20 — MC68701 OSCILLATOR CHARACTERISTICS

(a) Nominal Recommended Crystal Parameters



C_L = 20 pF (typical)

NOTE

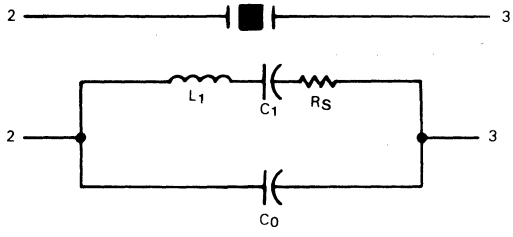
TTL-compatible oscillators may be obtained from:

Motorola Component Products
Attn: Data Clock Sales
2553 N. Edginton St.
Franklin Park, IL 60131
Tel: 312-451-1000
Telex: 433-0067

MC68701 Nominal Crystal Parameters

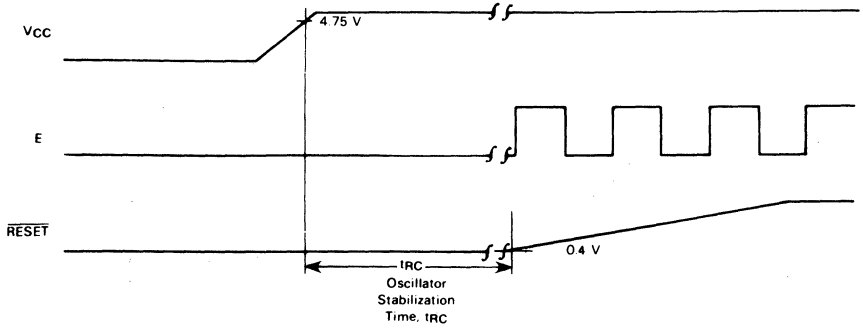
	3.58 MHz	4.00 MHz	5.0 MHz	6.0 MHz	8.0 MHz
RS	60 Ω	50 Ω	30-50 Ω	30-50 Ω	20-40 Ω
C_0	3.5 pF	6.5 pF	4.6 pF	4-6 pF	4.6 pF
C_1	0.015 pF	0.025 pF	0.01-0.02 pF	0.01-0.02 pF	0.01-0.02 pF
Q	>40 k	>30 k	>20 k	>20 k	>20 k

*Note: These are representative AT-cut crystal parameters only. Crystals of other types of cuts may also be used.



Equivalent Circuit

(b) Oscillator Stabilization Time (t_{RC})



SC1 And SC2 In Expanded Non-Multiplexed Mode

In the Expanded Non-Multiplexed Mode, both SC1 and SC2 are configured as outputs. SC1 functions as Input/Output Select ($\overline{IOS}$) and is asserted only when $0100 through $01FF is sensed on the internal address bus.

SC2 is configured as Read/Write and is used to control the direction of data bus transfers. An MPU read is enabled when Read/Write and E are high.

SC1 And SC2 In Expanded Multiplexed Mode

In the Expanded Multiplexed Modes, both SC1 and SC2 are configured as outputs. SC1 functions as Address Strobe and can be used to demultiplex the eight least significant addresses and the data bus. A latch controlled by Address Strobe captures address on the negative edge, as shown in Figure 15.

SC2 is configured as Read/Write and is used to control the direction of data bus transfers. An MPU read is enabled when Read/Write and E are high.

3

P10-P17 (PORT 1)

Port 1 is a mode independent 8-bit I/O port with each line an input or output as defined by the Port 1 Data Direction Register. The TTL compatible three-state output buffers can drive one Schottky TTL load and 30 pF, Darlington transistors, or CMOS devices using external pullup resistors. It is configured as a data input port by $\overline{RESET}$. Unused lines can remain unconnected.

P20-P24 (PORT 2)

Port 2 is a mode-independent, 5-bit, multipurpose I/O port. The voltage levels present on P20, P21, and P22 on the rising edge of $\overline{RESET}$ determine the operating mode of the MCU. The entire port is then configured as a data input port. The Port 2 lines can be selectively configured as data output lines by setting the appropriate bits in the Port 2 Data Direction Register. The Port 2 Data Register is used to move data through the port. However, if P21 is configured as an output, it will be tied to the timer Output Compare function and cannot be used to provide output from the Port 2 Data Register.

Port 2 can also be used to provide an interface for the Serial Communications Interface and the timer Input Edge function. These configurations are described in the appropriate SCI and Timer sections of this publication.

The Port 2 high-impedance, TTL compatible output buffers are capable of driving one Schottky TTL load and 30 pF or CMOS devices using external pullup resistors.

PORT 2 DATA REGISTER

7	6	5	4	3	2	1	0	
PC2	PC1	PC0	P24	P23	P22	P21	P20	$0003

P30-P37 (PORT 3)

Port 3 can be configured as an I/O port, a bidirectional 8-bit data bus, or a multiplexed address/data bus depending on the operating mode. The TTL compatible three-state output buffers can drive one Schottky TTL load and 90 pF. Unused lines can remain unconnected.

Port 3 In Single-Chip Mode

Port 3 is an 8-bit I/O port in the Single-Chip Mode, with each line configured by the Port 3 Data Direction Register. There are also two lines, $\overline{IS3}$ and $\overline{OS3}$, which can be used to control Port 3 data transfers.

Three Port 3 options are controlled by the Port 3 Control and Status Register and are available only in Single-Chip Mode: (1) Port 3 input data can be latched using IS3 as a control signal, (2) $\overline{OS3}$ can be generated by either an MPU read or write to the Port 3 Data Register, and (3) an $\overline{IRQ1}$ interrupt can be enabled by an $\overline{IS3}$ negative edge. Port 3 latch timing is shown in Figure 4.

PORT 3 CONTROL AND STATUS REGISTER

7	6	5	4	3	2	1	0	
IS3 Flag	IS3 IRQ1 Enable	X	OSS	Latch Enable	X	X	X	$000F

Bit 0-2 Not used.

Bit 3 LATCH ENABLE. This bit controls the input latch for Port 3. If set, input data is latched by an $\overline{IS3}$ negative edge. The latch is transparent after a read of Port 3 Data Register. LATCH ENABLE is cleared during reset.

Bit 4 OSS (Output Strobe Select). This bit determines whether $\overline{OS3}$ will be generated by a read or write of the Port 3 Data Register. When clear, the strobe is generated by a read; when set, it is generated by a write. OSS is cleared during reset.

Bit 5 Not used.

Bit 6 IS3 $\overline{IRQ1}$ ENABLE. When set, an $\overline{IRQ1}$ interrupt will be enabled whenever IS3 FLAG is set; when clear, the interrupt is inhibited. This bit is cleared during reset.

Bit 7 IS3 FLAG. This read-only status bit is set by an IS3 negative edge. It is cleared by a read of the Port 3 Control and Status Register (with IS3 FLAG set) followed by a read or write to the Port 3 Data Register or during reset.

Port 3 In Expanded Non-Multiplexed Mode

Port 3 is configured as a bidirectional data bus (D7-D0) in the Expanded Non-Multiplexed Mode. The direction of data transfers is controlled by Read/Write (SC2). Data is clocked by E (Enable).

Port 3 In Expanded Multiplexed Mode

Port 3 is configured as a time multiplexed address (A0-A7) and data bus (D7-D0) in the Expanded Multiplexed Modes where Address Strobe (AS) can be used to demultiplex the two buses. Port 3 is held in a high impedance state between valid address and data to prevent potentional bus conflicts.

P40-P47 (PORT 4)

Port 4 is configured as an 8-bit I/O port, as address outputs, or as data inputs depending on the operating mode. Port 4 can drive one Schottky TTL load and 90 pF and is the only port with internal pullup resistors. Unused lines can remain unconnected.

Port 4 In Single Chip Mode

In Single Chip Mode, Port 4 functions as an 8-bit I/O port with each line configured by the Port 4 Data Direction Register. Internal pullup resistors allow the port to directly interface with CMOS at 5 volt levels. External pullup resistors to more than 5 volts, however, cannot be used.

Port 4 In Expanded Non-Multiplexed Mode

Port 4 is configured during reset as an 8-bit input port, where the Port 4 Data Direction Register can be written to provide any or all of eight address lines A0 to A7. Internal pullup resistors pull the lines high until the Port 4 Data Direction Register is configured.

Port 4 In Expanded Multiplexed Mode

In all Expanded Multiplexed modes except Mode 6, Port 4 functions as half of the address bus and provides A8 to A15. In Mode 6, the port is configured during reset as an 8-bit parallel input port, where the Port 4 Data Direction Register can be written to provide any or all of upper address lines A8 to A15. Internal pullup resistors pull the lines high until the Port 4 Data Direction Register is configured, where bit 0 controls A8.

RESIDENT MEMORY

The MC68701 has 128 bytes of onboard RAM and 2048 bytes of onboard UV erasable EPROM. This memory is controlled by four bits in the RAM/EPROM Control Register.

One half of the RAM is powered through the V_{CC} standby pin and is maintainable during V_{CC} powerdown. This standby portion of the RAM consists of 64 bytes located from $80 through $BF.

Power must be supplied to V_{CC} standby if the internal RAM is to be used, regardless of whether standby power operation is anticipated. In Mode 3, V_{CC} standby should be tied to ground.

The RAM is controlled by the RAM/EPROM Control Register.

RAM/EPROM CONTROL REGISTER ($14)

The RAM/EPROM Control Register includes four bits: STBY PWR, RAME, PPC, and PLC. Two of these bits, STBY PWR and RAME, are used to control RAM access and determine the adequacy of the standby power source during power-down operation. It is intended that RAME be cleared and STBY PWR be set as part of a power-down procedure. RAME and STBY PWR are Read/Write bits.

The remaining two bits, PLC and PPC, control the operation of the EPROM. PLC and PPC are readable in all modes but can be changed only in Mode 0. The PLC bit can be written without restriction in Mode 0, but operation of the PPC bit is controlled by the state of PLC.

Associated with the EPROM are an 8-bit data latch and a 16-bit address latch. The data latch is enabled at all times, latching each data byte written to the EPROM. The address latch is controlled by the PLC bit.

A description of the RAM/EPROM Control Register follows.

MC68701 RAM/EPROM CONTROL REGISTER

7	6	5	4	3	2	1	0	
STBY PWR	RAME	X	X	X	X	PPC	PLC	$14

Bit 0 — PLC. Programming Latch Control. This bit controls (a) a latch which captures the EPROM address to be programmed and (b) whether the PPC bit can be cleared. The latch is triggered by an MPU write to a location in the EPROM. This bit is set during reset and can be cleared only in Mode 0. The PLC bit is defined as follows:

PLC = 0 EPROM address latch enabled; EPROM address is latched during MPU writes to the EPROM.

PLC = 1 EPROM address latch is transparent.

Bit 1 — PPC. Programming Power Control. This bit gates power from the $\overline{RESET}$/V_{PP} pin to the EPROM programming circuit. PPC is set during reset and whenever the PLC bit is set. It can be cleared only if (a) operating in Mode 0, and (b) if PLC has been previously cleared. The PPC bit is defined as follows:

PPC = 0 EPROM programming power (V_{PP}) applied.

PPC = 1 EPROM programming power (V_{PP}) is not applied.

Bit 2-5 — Unused.

Bit 6 RAME — RAM Enable. This Read/Write bit can be used to remove the entire RAM from the internal memory map. RAME is set (enabled) during reset provided standby power is available on the positive edge of reset. If RAME is clear, any access to a RAM address is external. If RAME is set and not in Mode 3, the RAM is included in the internal map.

Bit 7 STBY PWR — Standby Power. This bit is a read/write status bit which, when once set, remains set as long as V_{CC} standby remains above V_{SBB} (minimum). As long as this bit is set following a period of standby operation, the standby power supply has adequately preserved the data in the standby RAM. If this bit is cleared during a period of standby operation, it indicates that V_{CC} standby had fallen to a level sufficiently below V_{SBB} (minimum) to suspect that data in the standby RAM is not valid. This bit can be set only by software and is not affected during reset.

Note that if PPC and PLC are set, they cannot be simultaneously cleared with a single MPU write. The PLC bit must be cleared prior to attempting to clear PPC. If both PPC and PLC are clear, setting PLC will also set PPC. In addition,

3

it is assumed that Vpp is applied to the RESET/Vpp pin whenever PPC is clear. If this is not the case, the result is undefined.

ERASING THE MC68701 EPROM

Ultraviolet erasure will clear all bits of the EPROM to the "0" state. Note that this erased state differs from that of some other widely used EPROMs (such as the MCM68708) where the erased state is a "1". The MC68701 EPROM is programmed by erasing it to "0's" and entering "1's" into the desired bit locations.

The MC68701 EPROM can be erased by exposure to high intensity ultraviolet light with a wave length of 2537A for a minimum of 30 minutes. The recommended integrated dose (UV intensity X exposure time) is 15 Ws/cm. The lamps should be used without shortwave filters and the MC68701 should be positioned about one inch away from the UV tubes.

The MC68701 transparent lid should always be covered after erasing. This protects both the EPROM and light-sensitive nodes from accidental exposure to ultraviolet light.

3

PROGRAMMING THE MC68701 EPROM

When the MC68701 is released from Reset in Mode 0, a vector is fetched from location $BFFE:BFFF. This provides a method for an external program to obtain control of the microcomputer with access to every location in the EPROM.

To program the EPROM, it is necessary to operate the MC68701 in Mode 0 under the control of a program resident in external memory which can facilitate loading and programming of the EPROM. After the pattern has been loaded into external memory, the EPROM can be programmed as follows:

a. Apply programming power (Vpp) to the $\overline{\text{RESET}}$/Vpp pin.
b. Clear the PLC control bit and set the PPC bit by writing $FE to the RAM/EPROM Control Register.
c. Write data to the next EPROM location to be programmed. Triggered by an MPU write to the EPROM, internal latches capture both the EPROM address and the data byte.
d. Clear the PPC bit for programming time, t_{pp}, by writing $FC to the RAM/EPROM Control Register and waiting for time, t_{pp}. This step gates the programming power (Vpp) from the $\overline{\text{RESET}}$/Vpp pin to the EPROM which programs the location.
e. Repeat steps b through d for each byte to be programmed.
f. Set the PLC and PPC bits by writing $FF to the RAM/EPROM control register.
g. Remove the programming power (Vpp) from the $\overline{\text{RESET}}$/Vpp pin. The EPROM can now be read and verified.

Because of the erased state of an EPROM byte is $00, it is not necessary to program a location which is to contain $00. Finally, it should be noted that the result of inadvertently programming a location more than once is the logical OR of the data patterns.

A routine which can be used to program the MC68701 EPROM is provided at the end of this publication. This non-reentrant routine requires four double byte variables named IMBEQ, IMEND, PNTR, and WAIT to be initialized prior to entry to the routine. These variables indicate (a) the first and last memory locations which bound the data to be programmed into the EPROM, (b) the first EPROM location to be programmed, and (c) a number which is used to generate the programming time delay. The last variable, WAIT, takes into account the MCU input crystal (or TTL-compatible clock) frequency to insure the programming time, t_{pp}, is met. WAIT is defined as the number of MPU E-cycles that will occur in the real-time EPROM programming interval, t_{pp}. For example, if $t_{pp} = 50$ milliseconds and the MC68701 is being driven with a 4.00 MHz TTL-compatible clock:

$$\text{WAIT (MPU E-cycles)} = .t_{pp} * (\text{MCU INPUT FREQ/})4 * 10^6$$
$$= 50000(4 * 10^6)/4 * 10^6$$
$$= 50000$$

NOTE

A monitor program called PRObug is available from Motorola Microsystems. PRObug contains a user option for programming the on-board MC68701 EPROM.

PROGRAMMABLE TIMER

The Programmable Timer can be used to perform input waveform measurements while independently generating an output waveform. Pulse widths can vary from several microseconds to many seconds. A block diagram of the Timer is shown in Figure 21.

COUNTER ($09:0A)

The key timer element is a 16-bit free-running counter which is incremented by E (Enable). It is cleared during reset and is read-only with one exception: a write to the counter ($09) will preset it to $FFF8. This feature, intended for testing, can disturb serial operations because the counter provides the SCI internal bit rate clock. TOF is set whenever the counter contains all 1's.

OUTPUT COMPARE REGISTER ($0B:0C)

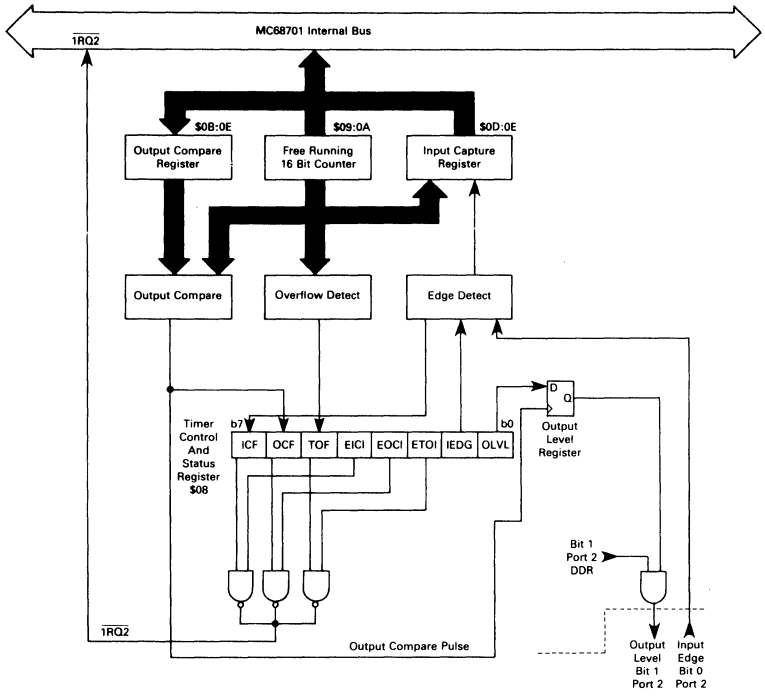
The Output Compare Register is a 16-bit Read/Write register used to control an output waveform or provide an arbitrary timeout flag. It is compared with the free-running counter on each E-cycle. When a match occurs, OCF is set and OLVL is clocked to an output level register. If Port 2, bit 1, is configured as an output, OLVL will appear at P21 and the Output Compare Register and OLVL can then be changed for the next compare. The function is inhibited for one cycle after a write to the high byte of the Compare Register ($0B) to ensure a valid compare. The Output Compare Register is set to $FFFF during reset.

INPUT CAPTURE REGISTER ($0D:0E)

The Input Capture Register is a 16-bit read-only register used to store the free-running counter when a "proper" input transition occurs as defined by IEDG. Port 2, bit 0 should be configured as an input, but the edge detect circuit always

PRObug is a trademark of Motorola Inc.

FIGURE 21 — BLOCK DIAGRAM OF PROGRAMMABLE TIMER



senses P20 even when configured as an output. An input capture can occur independently of ICF: the register always contains the most current value. Counter transfer is inhibited, however, between accesses of a double byte MPU read. The input pulse width must be at least two E-cycles to ensure an input capture under all conditions.

TIMER CONTROL AND STATUS REGISTER ($08)

The Timer Control and Status Register (TCSR) is an 8-bit register of which all bits are readable while bits 0-4 can be written. The three most significant bits provide the timer status and indicate if:

- a proper level transition has been detected,
- a match has occurred between the free-running counter and the output compare register, and
- the free-running counter has overflowed.

Each of the three events can generate an $\overline{IRQ2}$ interrupt and is controlled by an individual enable bit in the TCSR.

TIMER CONTROL AND STATUS REGISTER (TCSR)

7	6	5	4	3	2	1	0	
ICF	OCF	TOF	EICI	EOCI	ETOI	IEDG	OLVL	$0008

Bit 0 OLVL — Output level. OLVL is clocked to the output level register by a successful output compare and will appear at P21 if Bit 1 of the Port 2 Data Direction Register is set. It is cleared during reset.

Bit 1 EIDG — Input Edge. IEDG is cleared during reset and controls which level transition will trigger a counter transfer to the Input Capture Register:

IEDG = 0 Transfer on a negative-edge

IEDG = 1 Transfer on a positive-edge.

Bit 2 ETOI — Enable Timer Overflow Interrupt. When set, an $\overline{IRQ2}$ interrupt is enabled for a timer overflow; when clear, the interrupt is inhibited. It is cleared during reset.

Bit 3 EOCI — Enable Output Compare Interrupt. When set, an $\overline{IRQ2}$ interrupt is enabled for an output compare; when clear, the interrupt is inhibited. It is cleared during reset.

Bit 4 EICI — Enable Input Capture Interrupt. When set, an $\overline{IRQ2}$ interrupt is enabled for an input capture; when clear, the interrupt is inhibited. It is cleared during reset.

Bit 5 TOF — Timer Overflow Flag. TOF is set when the counter contains all 1's. It is cleared by reading the TCSR (with TOF set) then reading the counter high byte ($09), or by RESET.

Bit 6 OCF — Output Compare Flag. OCF is set when the Output Compare Register matches the free-running counter. It is cleared by reading the TCSR (with OCF set) and then writing to the Output Compare Register ($0B or $0C), or by RESET.

Bit 7 ICF — Input Capture Flag. ICF is set to indicate a proper level transition; it is cleared by reading the TCSR (with ICF set) and then the Input Capture Register High Byte ($0D), or by RESET.

3

SERIAL COMMUNICATIONS INTERFACE (SCI)

A full-duplex asynchronous Serial Communications Interface (SCI) is provided with two data formats and a variety of rates. The SCI transmitter and receiver are functionally independent, but use the same data format and bit rate. Serial data formats include standard mark/space (NRZ) and Bi-phase and both provide one start bit, eight data bits, and one stop bit. "Baud" and "bit rate" are used synonymously in the following description.

WAKE-UP FEATURE

In a typical serial loop multi-processor configuration, the software protocol will usually identify the addressee(s) at the beginning of the message. In order to permit uninterested MPU's to ignore the remainder of the message, a wake-up feature is included whereby all further SCI receiver flag (and interrupt) processing can be inhibited until the data line goes idle. An SCI receiver is re-enabled by an idle string of 11 consecutive 1's or during reset. Software must provide for the required idle string between consecutive messages and prevent it within messages.

PROGRAMMABLE OPTIONS

The following features of the SCI are programmable:

- format: standard mark/space (NRZ) or Bi-phase
- clock: external or internal bit rate clock
- Baud : one of 4 per E-clock frequency, or external clock (X8 desired baud)
- wake-up feature: enabled or disabled
- interrupt requests: enabled individually for transmitter and receiver
- clock output: internal bit rate clock enabled or disabled to P22

SERIAL COMMUNICATIONS REGISTERS

The Serial Communications Interface includes four addressable registers as depicted in Figure 22. It is controlled by the Rate and Mode Control Register and the Transmit/Receive Control and Status Register. Data is transmitted and received utilizing a write-only Transmit Register and a read-only Receive Register. The shift registers are not accessible to software.

Rate and Mode Control Register (RMCR) ($10)

The Rate and Mode Control Register controls the SCI bit rate, format, clock source, and under certain conditions, the configuration of P22. The register consists of four write-only bits which are cleared during reset. The two least significant bits control the bit rate of the internal clock and the remaining two bits control the format and clock source.

RATE AND MODE CONTROL REGISTER (RMCR)

7	6	5	4	3	2	1	0	
X	X	X	X	CC1	CC0	SS1	SS0	$0010

Bit 1:Bit 0 — SS1:SS0 Speed Select. These two bits select the Baud rate when using the internal clock. Four rates may be selected which are a function of the MCU input frequency. Table 6 lists bit time and rates for three selected MCU frequencies.

Bit 3:Bit 2 — CC1:CC0 Clock Control and Format Select. These two bits control the format and select the serial clock source. If CC1 is set, the DDR value for P22 is forced to the complement of CC0 and cannot be altered until CC1 is cleared. If CC1 is cleared after having been set, its DDR value is unchanged. Table 7 defines the formats, clock source, and use of P22.

If both CC1 and CC0 are set, an external TTL compatible clock must be connected to P22 at eight times (8X) the desired bit rate, but not greater than E, with a duty cycle of 50% (±10%). If CC1:CC0=10, the internal bit rate clock is provided at P22 regardless of the values for TE or RE.

NOTE: The source of SCI internal bit rate clock is the timer free running counter. An MPU write to the counter can disturb serial operations.

Transmit/Receive Control And Status Register (TRCSR) ($11)

The Transmit/Receive Control and Status Register controls the transmitter, receiver, wake-up feature, and two individual interrupts and monitors the status of serial operations. All eight bits are readable while bits 0 to 4 are also writable. The register is initialized to $20 by $\overline{\text{RESET}}$.

TRANSMIT/RECEIVE CONTROL AND STATUS REGISTER (TRCSR)

7	6	5	4	3	2	1	0	
RDRF	ORFE	TDRE	RIE	RE	TIE	TE	WU	$0011

TABLE 6 — SCI BIT TIMES AND RATES

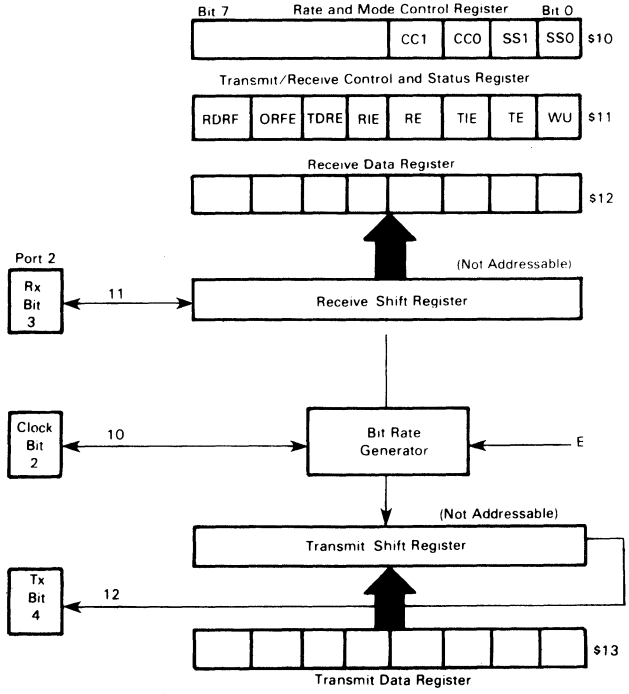
SS1:SS0		$4f_O \rightarrow$ E	2.4576 MHz 614.4 kHz	4.0 MHz 1.0 MHz	4.9152 MHz 1.2288 MHz
0	0	÷ 16	26 μs/38,400 Baud	16 μs/62,500 Baud	13.0 μs/76,800 Baud
0	1	÷ 128	208 μs/4,800 Baud	128 μs/7812.5 Baud	104.2 μs/9,600 Baud
1	0	÷ 1024	1.67 ms/600 Baud	1.024 ms/976.6 Baud	833.3 μs/1,200 Baud
1	1	÷ 4096	6.67 ms/150 Baud	4.096 ms/244.1 Baud	3.33 ms/300 Baud
External (P22)			Up to 76,800 Baud	Up to 125,000 Baud	Up to 153,600 Baud

TABLE 7 — SCI FORMAT AND CLOCK SOURCE CONTROL

CC1:CC0		Format	Clock Source	Port 2, Bit 2
0	0	Bi-Phase	Internal	Not Used
0	1	NRZ	Internal	Not Used
1	0	NRZ	Internal	Output
1	1	NRZ	External	Input

3

FIGURE 22 — SCI REGISTERS



Bit 0 WU — "Wake-up" on Idle Line. When set, WU enables the wake-up function; it is cleared by 11 consecutive 1's or during reset. WU will not set if the line is idle.

Bit 1 TE — Transmit Enable. When set, P24 DDR bit is set, cannot be changed, and will remain set if TE is subsequently cleared. When TE is changed from clear to set, the transmitter is connected to P24 and a preamble of nine consecutive 1's is transmitted. TE is cleared during reset.

Bit 2 TIE — Transmit Interrupt Enable. When set, an $\overline{IRQ2}$ interrupt is enabled when TDRE is set; when clear, the interrupt is inhibited. TE is cleared during reset.

Bit 3 RE — Receive Enable. When set, the P23 DDR bit is cleared, cannot be changed, and will remain clear if RE is subsequently cleared. While RE is set, the SCI receiver is enabled. RE is cleared during reset.

Bit 4 RIE — Receiver Interrupt Enable. When set, an IRQ2 interrupt is enabled when RDRF and/or ORFE is set; when clear, the interrupt is inhibited. RIE is cleared during reset.

Bit 5 TDRE — Transmit Data Register Empty. TDRE is set when the Transmit Data Register is transferred to the output serial shift register or during reset. It is cleared by reading the TRCSR (with TDRE set) and then writing to the Transmit Data Register. Additional data will be transmitted only if TDRE has been cleared.

Bit 6 ORFE — Overrun Framing Error. If set, ORFE indicates either an overrun or framing error. An overrun is a new byte ready to transfer to the Receiver Data Register with RDRF still set. A receiver framing error has occurred when the byte boundaries of the bit stream are not synchronized to the bit counter. An overrun can be distinguished from a framing error by the state of RDRF: if RDRF is set, then an overrun has occurred; otherwise a framing error has been detected. Data is not transferred to the Receive Data Register in an overrun condition. Unframed data causing a framed error is transferred to the Receive Data Register. However, subsequent data transfer is blocked until the framing error flag is cleared.* ORFE is cleared by reading the TRCSR (with ORFE set) then the Receive Data Register, or during reset.

Bit 7 RDRF — Receive Data Register Full. RDRF is set when the input serial shift register is transferred to the Receive Data Register. It is cleared by reading the TRCSR (with RDRF set), and then the Receive Data Register, or during reset.

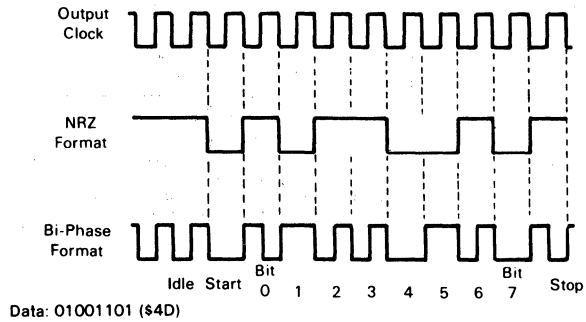
SERIAL OPERATIONS

The SCI is initialized by writing control bytes first to the Rate and Mode Control Register and then to the Transmit/Receive Control and Status Register. When TE is set, the output of the transmit serial shift register is connected to P24 and serial output is initiated by transmitting to 9-bit preamble of 1's.

At this point one of two situations exist: 1) if the Transmit Data Register is empty (TDRE = 1), a continuous string of 1's will be sent indicating an idle line, or 2) if a byte has been written to the Transmit-Data Register (TDRE = 0), it will be transferred to the output serial shift register (synchronized with the bit rate clock), TDRE will be set, and transmission will begin.

The start bit (0), eight data bits (beginning with bit 0) and a stop bit (1), will be transmitted. If TDRE is still set when the next byte transfer should occur, 1's will be sent until more data is provided. In Bi-phase format, the output toggles at the start of each bit and at half-bit time when a "1" is sent. Receive operation is controlled by RE which configures P23 as an input and enables the receiver. SCI data formats are illustrated in Figure 23.

FIGURE 23 — SCI DATA FORMATS



* Devices made with mask numbers T7A and CB4 do not transfer unframed data to the Receive Data Register.

INSTRUCTION SET

The MC68701 is upward source and object code compatible with the MC6800. Execution times of key instructions have been reduced and several new instructions have been added, including a hardware multiply. A list of new operations added to the MC6800 instruction set is shown in Table 1. In addition, two new special opcodes, 4E and 5E, are provided for test purposes. These opcodes force the program counter to increment like a 16-bit counter, causing address lines used in the expanded modes to increment until the device is reset. These opcodes have no mnemonics.

The coding of the first (or only) byte corresponding to an executable instruction is sufficient to identify the instruction and the addressing mode. The hexadecimal equivalents of the binary codes, which result from the translation of the 82 instructions in all valid modes of addressing, are shown in Table 8. There are 220 valid machine codes, 34 unassigned codes, and 2 reserved for test purposes.

PROGRAMMING MODEL

A programming model for the MC68701 is shown in Figure 9. Accumulator A can be concatenated with accumulator B and jointly referred to as accumulator D where A is the most significant byte. Any operation which modifies the double accumulator will also modify accumulator A and/or B. Other registers are defined as follows:

Program Counter — The program counter is a 16-bit register which always points to the next instruction.

Stack Pointer — The stack pointer is a 16-bit register which contains the address of the next available location in a pushdown/pullup (LIFO) queue. The stack resides in random access memory at a location defined by the programmer.

Index Register — The Index Register is a 16-bit register which can be used to store data or provide an address for the indexed mode of addressing.

Accumulators — The MCU contains two 8-bit accumulators, A and B, which are used to store operands and results from the arithmetic logic unit (ALU). They can also be concatenated and referred to as the D (double) accumulator.

Condition Code Registers — The condition code register indicates the results of an instruction and includes the Overflow (V), Carry/Borrow from MSB (C), and Half Carry following five condition bits: Negative (N), Zero (Z), from bit 3 (H). These bits are testable by the conditional branch instructions. Bit 4 is the interrupt mask (I-bit) and inhibits all maskable interrupts when set. The two unused bits, B6 and B7 are read as ones.

ADDRESSING MODES

The MC68701 provides six addressing modes which can be used to reference memory. A summary of addressing modes for all instructions is presented in Tables 9, 10, 11, and 12 where execution times are provided in E cycles. Instruction execution times are summarized in Table 13. With an input frequency of 4 MHz, E cycles are equivalent to microseconds. A cycle-by-cycle description of bus activity for each instruction is provided in Table 14 and a description of selected instructions is shown in Figure 24.

Immediate Addressing — The operand or "immediate byte(s)" is contained in the following byte(s) of the instruction where the number of bytes matches the size of the register. These are two or three byte instructions.

Direct Addressing — The least significant byte of the operand address is contained in the second byte of the instruction and the most significant byte is assumed to be $00. Direct addressing allows the user to access $00 through $FF using two byte instructions and execution time is reduced by eliminating the additional memory access. In most applications, the 256-byte area is reserved for frequently referenced data.

Extended Addressing — The second and third bytes of the instruction contain the absolute address of the operand. These are three byte instrutions.

Indexed Addressing — The unsigned offset contained in the second byte of the instruction is added with carry to the Index Register and used to reference memory without changing the Index Register. These are two byte instructions.

Inherent Addressing — The operand(s) are registers and no memory reference is required. These are single byte instructions.

Relative Addressing — Relative addressing is used only for branch instructions. If the branch condition is true, the Program Counter is overwritten with the sum of a signed single byte displacement in the second byte of the instruction and the current Program Counter. This provides a branch range of −126 to 129 bytes from the first byte of the instruction. These are two byte instructions.

TABLE 8 — CPU INSTRUCTION MAP

OP	MNEM	MODE	~	#	OP	MNEM	MODE	~	#	OP	MNEM	MODE	~	#	OP	MNEM	MODE	~	#	OP	MNEM	MODE	~	#
00	•				34	DES	INHER	3	1	68	ASL	INDXD	6	2	9C	CPX	DIR	5	2	D0	SUBB	DIR	3	2
01	NOP	INHER	2	1	35	TXS		3	1	69	ROL		6	2	9D	JSR		5	2	D1	CMPB		3	2
02	•				36	PSHA		3	1	6A	DEC		6	2	9E	LDS		4	2	D2	SBCB		3	2
03	•				37	PSHB		3	1	6B	•				9F	STS	DIR	4	2	D3	ADDD		5	2
04	LSRD		3	1	38	PULX		5	1	6C	INC		6	2	A0	SUBA	INDXD	4	2	D4	ANDB		3	2
05	ASLD		3	1	39	RTS		5	1	6D	TST		6	2	A1	CMPA		4	2	D5	BITB		3	2
06	TAP		2	1	3A	ABX		3	1	6E	JMP		3	2	A2	SBCA		4	2	D6	LDAB		3	2
07	TPA		2	1	3B	RTI		10	1	6F	CLR	INDXD	6	2	A3	SUBD		6	2	D7	STAB		3	2
08	INX		3	1	3C	PSHX		4	1	70	NEG	EXTND	6	3	A4	ANDA		4	2	D8	EORB		3	2
09	DEX		3	1	3D	MUL		10	1	71	•				A5	BITA		4	2	D9	ADCB		3	2
0A	CLV		2	1	3E	WAI		9	1	72	•				A6	LDAA		4	2	DA	ORAB		3	2
0B	SEV		2	1	3F	SWI		12	1	73	COM		6	3	A7	STAA		4	2	DB	ADDB		3	2
0C	CLC		2	1	40	NEGA		2	1	74	LSR		6	3	A8	EORA		4	2	DC	LDD		4	2
0D	SEC		2	1	41	•				75	•				A9	ADCA		4	2	DD	STD		4	2
0E	CLI		2	1	42	•				76	ROR		6	3	AA	ORAA		4	2	DE	LDX		4	2
0F	SEI		2	1	43	COMA		2	1	77	ASR		6	3	AB	ADDA		4	2	DF	STX	DIR	4	2
10	SBA		2	1	44	LSRA		2	1	78	ASL		6	3	AC	CPX		6	2	E0	SUBB	INDXD	4	2
11	CBA		2	1	45	•				79	ROL		6	3	AD	JSR		6	2	E1	CMPB		4	2
12	•				46	RORA		2	1	7A	DEC		6	3	AE	LDS		5	2	E2	SBCB		4	2
13	•				47	ASRA		2	1	7B	•				AF	STS	INDXD	5	2	E3	ADDD		6	2
14	•				48	ASLA		2	1	7C	INC		6	3	B0	SUBA	EXTND	4	3	E4	ANDB		4	2
15	•				49	ROLA		2	1	7D	TST		6	3	B1	CMPA		4	3	E5	BITB		4	2
16	TAB		2	1	4A	DECA		2	1	7E	JMP		3	3	B2	SBCA		4	3	E6	LDAB		4	2
17	TBA		2	1	4B	•				7F	CLR	EXTND	6	3	B3	SUBD		6	3	E7	STAB		4	2
18	•				4C	INCA		2	1	80	SUBA	IMMED	2	2	B4	ANDA		4	3	E8	EORB		4	2
19	DAA	INHER	2	1	4D	TSTA		2	1	81	CMPA		2	2	B5	BITA		4	3	E9	ADCB		4	2
1A	•				4E	T				82	SBCA		2	2	B6	LDAA		4	3	EA	ORAB		4	2
1B	ABA	INHER	2	1	4F	CLRA		2	1	83	SUBD		4	3	B7	STAA		4	3	EB	ADDB		4	2
1C	•				50	NEGB		2	1	84	ANDA		2	2	B8	EORA		4	3	EC	LDD		5	2
1D	•				51	•				85	BITA		2	2	B9	ADCA		4	3	ED	STD		5	2
1E	•				52	•				86	LDAA		2	2	BA	ORAA		4	3	EE	LDX		5	2
1F	•				53	COMB		2	1	87	•				BB	ADDA		4	3	EF	STX	INDXD	5	2
20	BRA	REL	3	2	54	LSRB		2	1	88	EORA		2	2	BC	CPX		6	3	F0	SUBB	EXTND	4	3
21	BRN		3	2	55	•				89	ADCA		2	2	BD	JSR		6	3	F1	CMPB		4	3
22	BHI		3	2	56	RORB		2	1	8A	ORAA		2	2	BE	LDS		5	3	F2	SBCB		4	3
23	BLS		3	2	57	ASRB		2	1	8B	ADDA		2	2	BF	STS	EXTND	5	3	F3	ADDD		6	3
24	BCC		3	2	58	ASLB		2	1	8C	CPX	IMMED	4	3	C0	SUBB	IMMED	2	2	F4	ANDB		4	3
25	BCS		3	2	59	ROLB		2	1	8D	BSR	REL	6	2	C1	CMPB		2	2	F5	BITB		4	3
26	BNE		3	2	5A	DECB		2	1	8E	LDS	IMMED	3	3	C2	SBCB		2	2	F6	LDAB		4	3
27	BEQ		3	2	5B	•				8F	•				C3	ADDD		4	3	F7	STAB		4	3
28	BVC		3	2	5C	INCB		2	1	90	SUBA	DIR	3	2	C4	ANDB		2	2	F8	EORB		4	3
29	BVS		3	2	5D	TSTB		2	1	91	CMPA		3	2	C5	BITB		2	2	F9	ADCB		4	3
2A	BPL		3	2	5E	T				92	SBCA		3	2	C6	LDAB		2	2	FA	ORAB		4	3
2B	BMI		3	2	5F	CLRB	INHER	2	1	93	SUBD		5	2	C7	•				FB	ADDB		4	3
2C	BGE		3	2	60	NEG	INDXD	6	2	94	ANDA		3	2	C8	EORB		2	2	FC	LDD		5	3
2D	BLT		3	2	61	•				95	BITA		3	2	C9	ADCB		2	2	FD	STD		5	3
2E	BGT		3	2	62	•				96	LDAA		3	2	CA	ORAB		2	2	FE	LDX		5	3
2F	BLE	REL	3	2	63	COM		6	2	97	STAA		3	2	CB	ADDB		2	2	FF	STX	EXTND	5	3
30	TSX	INHER	3	1	64	LSR		6	2	98	EORA		3	2	CC	LDD		3	3					
31	INS		3	1	65	•				99	ADCA		3	2	CD	•								
32	PULA		4	1	66	ROR		6	2	9A	ORAA		3	2	CE	LDX	IMMED	3	3					
33	PULB		4	1	67	ASR	INDXD	6	2	9B	ADDA		3	2	CF	•								

*UNDEFINED OP CODE

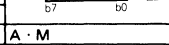
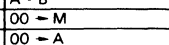
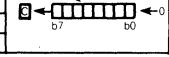
NOTES:

1. Addressing Modes
   INHER = Inherent INDXD = Indexed IMMED = Immediate
   REL = Relative EXTND = Extended DIR = Direct
2. Unassigned opcodes are indicated by "•" and should not be executed.
3. Codes marked by "T" force the PC to function as a 16-bit counter.

TABLE 9 — INDEX REGISTER AND STACK MANIPULATION INSTRUCTIONS

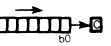
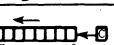
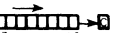
Pointer Operations	MNEM	Immed Op	Immed ~	Immed #	Direct Op	Direct ~	Direct #	Index Op	Index ~	Index #	Extend Op	Extend ~	Extend #	Inherent Op	Inherent ~	Inherent #	Boolean/ Arithmetic Operation	5 H	4 I	3 N	2 Z	1 V	0 C
Compare Index Register	CPX	8C	4	3	9C	5	2	AC	6	2	BC	6	3				X − M:M + 1	•	•	↕	↕	↕	↕
Decrement Index Register	DEX													09	3	1	X − 1 → X	•	•	•	↕	•	•
Decrement Stack Pointer	DES													34	3	1	SP − 1 → SP	•	•	•	•	•	•
Increment Index Register	INX													08	3	1	X + 1 → X	•	•	•	↕	•	•
Increment Stack Pointer	INS													31	3	1	1 SP + 1 → SP	•	•	•	•	•	•
Load Index Register	LDX	CE	3	3	DE	4	2	EE	5	2	FE	5	3				M → X_H, (M + 1) → X_L	•	•	↕	↕	R	•
Load Stack Pointer	LDS	8E	3	3	9E	4	2	AE	5	2	BE	5	3				M → SP_H, (M + 1) → SP_L	•	•	↕	↕	R	•
Store Index Register	STX				DF	4	2	EF	5	2	FF	5	3				X_H → M, X_L → (M + 1)	•	•	↕	↕	R	•
Store Stack Pointer	STS				9F	4	2	AF	5	2	BF	5	3				SP_H → M, SP_L → (M + 1)	•	•	↕	↕	R	•
Index Reg → Stack Pointer	TXS													35	3	1	X − 1 → SP	•	•	•	•	•	•
Stack Pntr → Index Register	TSX													30	3	1	SP + 1 → X	•	•	•	•	•	•
Add	ABX													3A	3	1	B + X → X	•	•	•	•	•	•
Push Data	PSHX													3C	4	1	X_L → M_{SP}, SP − 1 → SP X_H → M_{SP}, SP − 1 → SP	•	•	•	•	•	•
Pull Data	PULX													38	5	1	SP + 1 → SP, M_{SP} → X_H SP + 1 → SP, M_{SP} → X_L	•	•	•	•	•	•

TABLE 10 — ACCUMULATOR AND MEMORY INSTRUCTIONS (Sheet 1 of 2)

Accumulator and Memory Operations	MNE	Immed Op	Immed ~	Immed #	Direct Op	Direct ~	Direct #	Index Op	Index ~	Index #	Extend Op	Extend ~	Extend #	Inher Op	Inher ~	Inher #	Boolean Expression	H	I	N	Z	V	C
Add Acmltrs	ABA													1B	2	1	A + B → A	↕	•	↕	↕	↕	↕
Add B to X	ABX													3A	3	1	00:B + X → X	•	•	•	•	•	•
Add with Carry	ADCA	89	2	2	99	3	2	A9	4	2	B9	4	3				A + M + C → A	↕	•	↕	↕	↕	↕
	ADCB	C9	2	2	D9	3	2	E9	4	2	F9	4	3				B + M + C → B	↕	•	↕	↕	↕	↕
Add	ADDA	8B	2	2	9B	3	2	AB	4	2	BB	4	3				A + M → A	↕	•	↕	↕	↕	↕
	ADDB	CB	2	2	DB	3	2	EB	4	2	FB	4	3				B + M → A	↕	•	↕	↕	↕	↕
Add Double	ADDD	C3	4	3	D3	5	2	E3	6	2	F3	6	3				D + M:M + 1 → D	•	•	↕	↕	↕	↕
And	ANDA	84	2	2	94	3	2	A4	4	2	B4	4	3				A · M → A	•	•	↕	↕	R	•
	ANDB	C4	2	2	D4	3	2	E4	4	2	F4	4	3				B · M → B	•	•	↕	↕	R	•
Shift Left, Arithmetic	ASL							68	6	2	78	6	3					•	•	↕	↕	↕	↕
	ASLA													48	2	1		•	•	↕	↕	↕	↕
	ASLB													58	2	1		•	•	↕	↕	↕	↕
Shift Left Dbl	ASLD													05	3	1		•	•	↕	↕	↕	↕
Shift Right, Arithmetic	ASR							67	6	2	77	6	3					•	•	↕	↕	↕	↕
	ASRA													47	2	1		•	•	↕	↕	↕	↕
	ASRB													57	2	1		•	•	↕	↕	↕	↕
Bit Test	BITA	85	2	2	95	3	2	A5	4	2	B5	4	3				A · M	•	•	↕	↕	R	•
	BITB	C5	2	2	D5	3	2	E5	4	2	F5	4	3				B · M	•	•	↕	↕	R	•
Compare Acmltrs	CBA													11	2	1	A − B	•	•	↕	↕	↕	↕
Clear	CLR							6F	6	2	7F	6	3				00 → M	•	•	R	S	R	R
	CLRA													4F	2	1	00 → A	•	•	R	S	R	R
	CLRB													5F	2	1	00 → B	•	•	R	S	R	R
Compare	CMPA	81	2	2	91	3	2	A1	4	2	B1	4	3				A − M	•	•	↕	↕	↕	↕
	CMPB	C1	2	2	D1	3	2	E1	4	2	F1	4	3				B − M	•	•	↕	↕	↕	↕
1's Complement	COM							63	6	2	73	6	3				$\overline{M}$ → M	•	•	↕	↕	R	S
	COMA													43	2	1	$\overline{A}$ → A	•	•	↕	↕	R	S
	COMB													53	2	1	$\overline{B}$ → B	•	•	↕	↕	R	S
Decimal Adj, A	DAA													19	2	1	Adj binary sum to BCD	•	•	↕	↕	↕	↕
Decrement	DEC							6A	6	2	7A	6	3				M − 1 → M	•	•	↕	↕	↕	•
	DECA													4A	2	1	A − 1 → A	•	•	↕	↕	↕	•
	DECB													5A	2	1	B − 1 → B	•	•	↕	↕	↕	•
Exclusive OR	EORA	88	2	2	98	3	2	A8	4	2	B8	4	3				A ⊕ M → A	•	•	↕	↕	R	•
	EORB	C8	2	2	D8	3	2	E8	4	2	F8	4	3				B ⊕ M → B	•	•	↕	↕	R	•
Increment	INC							6C	6	2	7C	6	3				M + 1 → M	•	•	↕	↕	↕	•
	INCA													4C	2	1	A + 1 → A	•	•	↕	↕	↕	•
	INCB													5C	2	1	B + 1 → B	•	•	↕	↕	↕	•
Load Acmltrs	LDAA	86	2	2	96	3	2	A6	4	2	B6	4	3				M → A	•	•	↕	↕	R	•
	LDAB	C6	2	2	D6	3	2	E6	4	2	F6	4	3				M → B	•	•	↕	↕	R	•
Load Double	LDD	CC	3	3	DC	4	2	EC	5	2	FC	5	3				M:M + 1 → D	•	•	↕	↕	R	•
Logical Shift, Left	LSL							68	6	2	78	6	3					•	•	↕	↕	↕	↕
	LSLA													48	2	1		•	•	↕	↕	↕	↕
	LSLB													58	2	1		•	•	↕	↕	↕	↕
	LSLD													05	3	1		•	•	↕	↕	↕	↕

3

TABLE 10 — ACCUMULATOR AND MEMORY INSTRUCTIONS (Sheet 2 of 2)

Accumulator and Memory Operations	MNE	Immed			Direct			Index			Extend			Inher			Boolean Expression	Condition Codes					
		Op	~	#	Op	~	#	Op	~	#	Op	~	#	Op	~	#		H	I	N	Z	V	C
Shift Right, Logical	LSR							64	6	2	74	6	3					•	•	R	↕	↕	↕
	LSRA													44	2	1		•	•	R	↕	↕	↕
	LSRB													54	2	1		•	•	R	↕	↕	↕
	LSRD													04	3	1		•	•	R	↕	↕	↕
Multiply	MUL													3D	10	1	A X B → D	•	•	•	•	•	↕
2's Complement (Negate)	NEG							60	6	2	70	6	3				00 - M → M	•	•	↕	↕	↕	↕
	NEGA													40	2	1	00 - A → A	•	•	↕	↕	↕	↕
	NEGB													50	2	1	00 - B → B	•	•	↕	↕	↕	↕
No Operation	NOP													01	2	1	PC + 1 → PC	•	•	•	•	•	•
Inclusive OR	ORAA	8A	2	2	9A	3	2	AA	4	2	BA	4	3				A + M → A	•	•	↕	↕	R	•
	ORAB	CA	2	2	DA	3	2	EA	4	2	FA	4	3				B + M → B	•	•	↕	↕	R	•
Push Data	PSHA													36	3	1	A → Stack	•	•	•	•	•	•
	PSHB													37	3	1	B → Stack	•	•	•	•	•	•
Pull Data	PULA													32	4	1	Stack → A	•	•	•	•	•	•
	PULB													33	4	1	Stack → B	•	•	•	•	•	•
Rotate Left	ROL							69	6	2	79	6	3					•	•	↕	↕	↕	↕
	ROLA													49	2	1		•	•	↕	↕	↕	↕
	ROLB													59	2	1		•	•	↕	↕	↕	↕
Rotate Right	ROR							66	6	2	76	6	3					•	•	↕	↕	↕	↕
	RORA													46	2	1		•	•	↕	↕	↕	↕
	RORB													56	2	1		•	•	↕	↕	↕	↕
Subtract Acmltr	SBA													10	2	1	A - B → A	•	•	↕	↕	↕	↕
Subtract with Carry	SBCA	82	2	2	92	3	2	A2	4	2	B2	4	3				A - M - C → A	•	•	↕	↕	↕	↕
	SBCB	C2	2	2	D2	3	2	E2	4	2	F2	4	3				B - M - C → B	•	•	↕	↕	↕	↕
Store Acmltrs	STAA				97	3	2	A7	4	2	B7	4	3				A → M	•	•	↕	↕	R	•
	STAB				D7	3	2	E7	4	2	F7	4	3				B → M	•	•	↕	↕	R	•
	STD				DD	4	2	ED	5	2	FD	5	3				D → M:M + 1	•	•	↕	↕	R	•
Subtract	SUBA	80	2	2	90	3	2	A0	4	2	B0	4	3				A - M → A	•	•	↕	↕	↕	↕
	SUBB	C0	2	2	D0	3	2	E0	4	2	F0	4	3				B - M → B	•	•	↕	↕	↕	↕
Subtract Double	SUBD	83	4	3	93	5	2	A3	6	2	B3	6	3				D - M:M + 1 → D	•	•	↕	↕	↕	↕
Transfer Acmltr	TAB													16	2	1	A → B	•	•	↕	↕	R	•
	TBA													17	2	1	B → A	•	•	↕	↕	R	•
Test, Zero or Minus	TST							6D	6	2	7D	6	3				M - 00	•	•	↕	↕	R	R
	TSTA													4D	2	1	A - 00	•	•	↕	↕	R	R
	TSTB													5D	2	1	B - 00	•	•	↕	↕	R	R

The condition code register notes are listed after Table 12.

TABLE 11 — JUMP AND BRANCH INSTRUCTIONS

Operations	MNEM	Direct			Relative			Index			Extend			Inherent			Branch Test	Condition Code Reg.					
		Op	~	#	Op	~	#	Op	~	#	Op	~	#	Op	~	#		5 H	4 I	3 N	2 Z	1 V	0 C
Branch Always	BRA				20	3	2										None	•	•	•	•	•	•
Branch Never	BRN				21	3	2										None	•	•	•	•	•	•
Branch If Carry Clear	BCC				24	3	2										C = 0	•	•	•	•	•	•
Branch If Carry Set	BCS				25	3	2										C = 1	•	•	•	•	•	•
Branch If = Zero	BEQ				27	3	2										Z = 1	•	•	•	•	•	•
Branch If ≥ Zero	BGE				2C	3	2										N ⊕ V = 0	•	•	•	•	•	•
Branch If > Zero	BGT				2E	3	2										Z + (N ⊕ V) = 0	•	•	•	•	•	•
Branch If Higher	BHI				22	3	2										C + Z = 0	•	•	•	•	•	•
Branch If Higher or Same	BHS				24	3	2										C = 0	•	•	•	•	•	•
Branch If ≤ Zero	BLE				2F	3	2										Z + (N ⊕ V) = 1	•	•	•	•	•	•
Branch If Carry Set	BLO				25	3	2										C = 1	•	•	•	•	•	•
Branch If Lower Or Same	BLS				23	3	2										C + Z = 1	•	•	•	•	•	•
Branch If < Zero	BLT				2D	3	2										N ⊕ V = 1	•	•	•	•	•	•
Branch If Minus	BMI				2B	3	2										N = 1	•	•	•	•	•	•
Branch If Not Equal Zero	BNE				26	3	2										Z = 0	•	•	•	•	•	•
Branch If Overflow Clear	BVC				28	3	2										V = 0	•	•	•	•	•	•
Branch If Overflow Set	BVS				29	3	2										V = 1	•	•	•	•	•	•
Branch If Plus	BPL				2A	3	2										N = 0	•	•	•	•	•	•
Branch To Subroutine	BSR				8D	6	2											•	•	•	•	•	•
Jump	JMP							6E	3	2	7E	3	3				See Special Operations-Figure 24	•	•	•	•	•	•
Jump To Subroutine	JSR	9D	5	2				AD	6	2	BD	6	3					•	•	•	•	•	•
No Operation	NOP													01	2	1		•	•	•	•	•	•
Return From Interrupt	RTI													3B	10	1		↕	↕	↕	↕	↕	↕
Return From Subroutine	RTS													39	5	1	See Special Operations-Figure 24	•	•	•	•	•	•
Software Interrupt	SWI													3F	12	1		•	S	•	•	•	•
Wait For Interrupt	WAI													3E	9	1		•	•	•	•	•	•

TABLE 12 — CONDITION CODE REGISTER MANIPULATION INSTRUCTIONS

Operations	Inherent				Boolean Operation	Condition Code Register					
	MNEM	Op	~	#		5 H	4 I	3 N	2 Z	1 V	0 C
Clear Carry	CLC	0C	2	1	0 → C	•	•	•	•	•	R
Clear Interrupt Mask	CLI	0E	2	1	0 → I	•	R	•	•	•	•
Clear Overflow	CLV	0A	2	1	0 → V	•	•	•	•	R	•
Set Carry	SEC	0D	2	1	1 → C	•	•	•	•	•	S
Set Interrupt Mask	SEI	0F	2	1	1 → I	•	S	•	•	•	•
Set Overflow	SEV	0B	2	1	1 → V	•	•	•	•	S	•
Accumulator A → CCR	TAP	06	2	1	A → CCR	↕	↕	↕	↕	↕	↕
CCR → Accumulator A	TPA	07	2	1	CCR → A	•	•	•	•	•	•

LEGEND

Op Operation Code (Hexadecimal)
~ Number of MPU Cycles
MSP Contents of memory location pointed to by Stack Pointer
Number of Program Bytes
+ Arithmetic Plus
– Arithmetic Minus
• Boolean AND
X Arithmetic Multiply
+ Boolean Inclusive OR
⊕ Boolean Exclusive OR
M Complement of M
→ Transfer Into
0 Bit = Zero
00 Byte = Zero

CONDITION CODE SYMBOLS

H Half-carry from bit 3
I Interrupt mask
N Negative (sign bit)
Z Zero (byte)
V Overflow, 2's complement
C Carry/Borrow from MSB
R Reset Always
S Set Always
↕ Affected
• Not Affected

TABLE 13 — INSTRUCTION EXECUTION TIMES IN E CYCLES

	ADDRESSING MODE					
	Immediate	Direct	Extended	Indexed	Inherent	Relative
ABA	●	●	●	●	2	●
ABX	●	●	●	●	3	●
ADC	2	3	4	4	●	●
ADD	2	3	4	4	●	●
ADDD	4	5	6	6	●	●
AND	2	3	4	4	●	●
ASL	●	●	6	6	2	●
ASLD	●	●	●	●	3	●
ASR	●	●	6	6	2	●
BCC	●	●	●	●	●	3
BCS	●	●	●	●	●	3
BEQ	●	●	●	●	●	3
BGE	●	●	●	●	●	3
BGT	●	●	●	●	●	3
BHI	●	●	●	●	●	3
BHS	●	●	●	●	●	3
BIT	2	3	4	4	●	●
BLE	●	●	●	●	●	3
BLO	●	●	●	●	●	3
BLS	●	●	●	●	●	3
BLT	●	●	●	●	●	3
BMI	●	●	●	●	●	3
BNE	●	●	●	●	●	3
BPL	●	●	●	●	●	3
BRA	●	●	●	●	●	3
BRN	●	●	●	●	●	3
BSR	●	●	●	●	●	6
BVC	●	●	●	●	●	3
BVS	●	●	●	●	●	3
CBA	●	●	●	●	2	●
CLC	●	●	●	●	2	●
CLI	●	●	●	●	2	●
CLR	●	●	6	6	2	●
CLV	●	●	●	●	2	●
CMP	2	3	4	4	●	●
COM	●	●	6	6	2	●
CPX	4	5	6	6	●	●
DAA	●	●	●	●	2	●
DEC	●	●	6	6	2	●
DES	●	●	●	●	3	●
DEX	●	●	●	●	3	●
EOR	2	3	4	4	●	●
INC	●	●	6	6	●	●
INS	●	●	●	●	3	●

	ADDRESSING MODE					
	Immediate	Direct	Extended	Indexed	Inherent	Relative
INX	●	●	●	●	3	●
JMP	●	●	3	3	●	●
JSR	●	5	6	6	●	●
LDA	2	3	4	4	●	●
LDD	3	4	5	5	●	●
LDS	3	4	5	5	●	●
LDX	3	4	5	5	●	●
LSL	●	●	6	6	2	●
LSLD	●	●	●	●	3	●
LSR	●	●	6	6	2	●
LSRD	●	●	●	●	3	●
MUL	●	●	●	●	10	●
NEG	●	●	6	6	2	●
NOP	●	●	●	●	2	●
ORA	2	3	4	4	●	●
PSH	●	●	●	●	3	●
PSHX	●	●	●	●	4	●
PUL	●	●	●	●	4	●
PULX	●	●	●	●	5	●
ROL	●	●	6	6	2	●
ROR	●	●	6	6	2	●
RTI	●	●	●	●	10	●
RTS	●	●	●	●	5	●
SBA	●	●	●	●	2	●
SBC	2	3	4	4	●	●
SEC	●	●	●	●	2	●
SEI	●	●	●	●	2	●
SEV	●	●	●	●	2	●
STA	●	3	4	4	●	●
STD	●	4	5	5	●	●
STS	●	4	5	5	●	●
STX	●	4	5	5	●	●
SUB	2	3	4	4	●	●
SUBD	4	5	6	6	●	●
SWI	●	●	●	●	12	●
TAB	●	●	●	●	2	●
TAP	●	●	●	●	2	●
TBA	●	●	●	●	2	●
TPA	●	●	●	●	2	●
TST	●	●	6	6	2	●
TSX	⊖	⊖	⊖	⊖	3	⊖
TXS	●	●	●	●	3	●
WAI	●	●	●	●	9	●

SUMMARY OF CYCLE-BY-CYCLE OPERATION

Table 14 provides a detailed description of the information present on the Address Bus, Data Bus, and the Read/Write (R/W) line during each cycle of each instruction.

The information is useful in comparing actual with expected results during debug of both software and hardware as the program is executed. The information is categorized in groups according to addressing mode and number of cycles per instruction. In general, instructions with the same addressing mode and number of cycles execute in the same manner. Exceptions are indicated in the table.

Note that during MPU reads of internal locations, the resultant value will not appear on the external Data Bus except in Mode 0. "High order" byte refers to the most significant byte of a 16-bit value.

TABLE 14 — CYCLE-BY-CYCLE OPERATION (Sheet 1 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
IMMEDIATE					
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	2	1 2	Opcode Address Opcode Address + 1	1 1	Opcode Operand Data
LDS LDX LDD	3	1 2 3	Opcode Address Opcode Address + 1 Opcode Address + 2	1 1 1	Opcode Operand Data (High Order Byte) Operand Data (Low Order Byte)
CPX SUBD ADDD	4	1 2 3 4	Opcode Address Opcode Address + 1 Opcode Address + 2 Address Bus FFFF	1 1 1 1	Opcode Operand Data (High Order Byte) Operand Data (Low Order Byte) Low Byte of Restart Vector
DIRECT					
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	3	1 2 3	Opcode Address Opcode Address + 1 Address of Operand	1 1 1	Opcode Address of Operand Operand Data
STA	3	1 2 3	Opcode Address Opcode Address + 1 Destination Address	1 1 0	Opcode Destination Address Data from Accumulator
LDS LDX LDD	4	1 2 3 4	Opcode Address Opcode Address + 1 Address of Operand Operand Address + 1	1 1 1 1	Opcode Address of Operand Operand Data (High Order Byte) Operand Data (Low Order Byte)
STS STX STD	4	1 2 3 4	Opcode Address Opcode Address + 1 Address of Operand Address of Operand + 1	1 1 0 0	Opcode Address of Operand Register Data (High Order Byte) Register Data (Low Order Byte)
CPX SUBD ADDD	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Operand Address Operand Address + 1 Address Bus FFFF	1 1 1 1 1	Opcode Address of Operand Operand Data (High Order Byte) Operand Data (Low Order Byte) Low Byte of Restart Vector
JSR	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Subroutine Address Stack Pointer Stack Pointer − 1	1 1 1 0 0	Opcode Irrelevant Data First Subroutine Opcode Return Address (Low Order Byte) Return Address (High Order Byte)

TABLE 14 — CYCLE-BY-CYCLE OPERATION (Sheet 2 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
EXTENDED					
JMP	3	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Jump Address (High Order Byte)
		3	Opcode Address + 2	1	Jump Address (Low Order Byte)
ADC EOR	4	1	Opcode Address	1	Opcode
ADD LDA		2	Opcode Address + 1	1	Address of Operand
AND ORA		3	Opcode Address + 2	1	Address of Operand (Low Order Byte)
BIT SBC		4	Address of Operand	1	Operand Data
CMP SUB					
STA	4	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Destination Address (High Order Byte)
		3	Opcode Address + 2	1	Destination Address (Low Order Byte)
		4	Operand Destination Address	0	Data from Accumulator
LDS	5	1	Opcode Address	1	Opcode
LDX		2	Opcode Address + 1	1	Address of Operand (High Order Byte)
LDD		3	Opcode Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	1	Operand Data (High Order Byte)
		5	Address of Operand + 1	1	Operand Data (Low Order Byte)
STS	5	1	Opcode Address	1	Opcode
STX		2	Opcode Address + 1	1	Address of Operand (High Order Byte)
STD		3	Opcode Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	0	Operand Data (High Order Byte)
		5	Address of Operand + 1	0	Operand Data (Low Order Byte)
ASL LSR	6	1	Opcode Address	1	Opcode
ASR NEG		2	Opcode Address + 1	1	Address of Operand (High Order Byte)
CLR ROL		3	Opcode Address + 2	1	Address of Operand (Low Order Byte)
COM ROR		4	Address of Operand	1	Current Operand Data
DEC TST*		5	Address Bus FFFF	1	Low Byte of Restart Vector
INC		6	Address of Operand	0	New Operand Data
CPX	6	1	Opcode Address	1	Opcode
SUBD		2	Opcode Address + 1	1	Operand Address (High Order Byte)
ADDD		3	Opcode Address + 2	1	Operand Address (Low Order Byte)
		4	Operand Address	1	Operand Data (High Order Byte)
		5	Operand Address + 1	1	Operand Data (Low Order Byte)
		6	Address Bus FFFF	1	Low Byte of Restart Vector
JSR	6	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Address of Subroutine (High Order Byte)
		3	Opcode Address + 2	1	Address of Subroutine (Low Order Byte)
		4	Subroutine Starting Address	1	Opcode of Next Instruction
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer − 1	0	Return Address (High Order Byte)

*TST does not perform the write cycle during the sixth cycle. The sixth cycle is another address bus = $FFFF.

TABLE 14 — CYCLE-BY-CYCLE OPERATION (Sheet 3 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
INDEXED					
JMP	3	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
ADC EOR	4	1	Opcode Address	1	Opcode
ADD LDA		2	Opcode Address + 1	1	Offset
AND ORA		3	Address Bus FFFF	1	Low Byte of Restart Vector
BIT SBC		4	Index Register Plus Offset	1	Operand Data
CMP SUB					
STA	4	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	0	Operand Data
LDS	5	1	Opcode Address	1	Opcode
LDX		2	Opcode Address + 1	1	Offset
LDD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	1	Operand Data (High Order Byte)
		5	Index Register Plus Offset + 1	1	Operand Data (Low Order Byte)
STS	5	1	Opcode Address	1	Opcode
STX		2	Opcode Address + 1	1	Offset
STD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	0	Operand Data (High Order Byte)
		5	Index Register Plus Offset + 1	0	Operand Data (Low Order Byte)
ASL LSR	6	1	Opcode Address	1	Opcode
ASR NEG		2	Opcode Address + 1	1	Offset
CLR ROL		3	Address Bus FFFF	1	Low Byte of Restart Vector
COM ROR		4	Index Register Plus Offset	1	Current Operand Data
DEC TST*		5	Address Bus FFFF	1	Low Byte of Restart Vector
INC		6	Index Register Plus Offset	0	New Operand Data
CPX	6	1	Opcode Address	1	Opcode
SUBD		2	Opcode Address + 1	1	Offset
ADDD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register + Offset	1	Operand Data (High Order Byte)
		5	Index Register + Offset + 1	1	Operand Data (Low Order Byte)
		6	Address Bus FFFF		Low Byte of Restart Vector
JSR	6	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register + Offset	1	First Subroutine Opcode
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer − 1	0	Return Address (High Order Byte)

*TST does not perform the write cycle during the sixth cycle. The sixth cycle is another address bus = $FFFF.

TABLE 14 — CYCLE-BY-CYCLE OPERATION (Sheet 4 of 5)

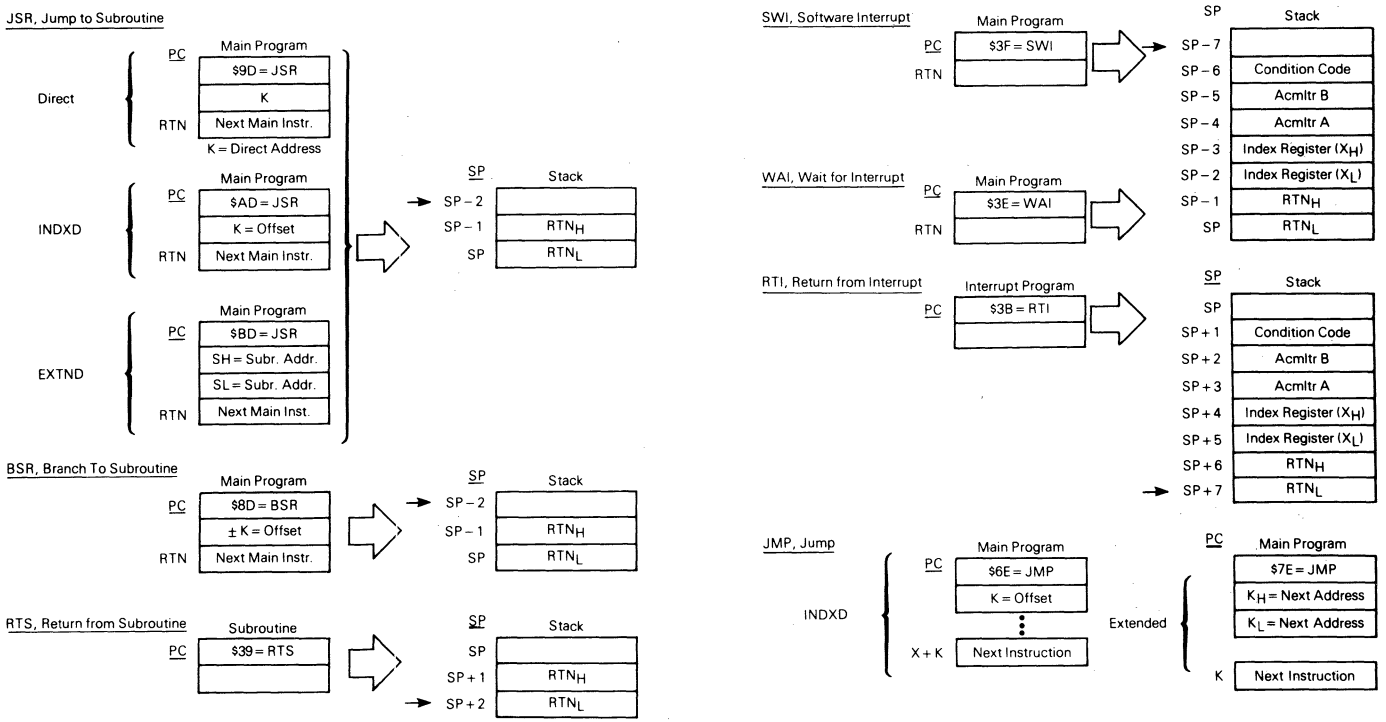
Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
INHERENT					
ABA DAA SEC ASL DEC SEI ASR INC SEV CBA LSR TAB CLC NEG TAP CLI NOP TBA CLR ROL TPA CLV ROR TST COM SBA	2	1 2	Opcode Address Opcode Address + 1	1 1	Opcode Opcode of Next Instruction
ABX	3	1 2 3	Opcode Address Opcode Address + 1 Address Bus FFFF	1 1 1	Opcode Irrelevant Data Low Byte of Restart Vector
ASLD LSRD	3	1 2 3	Opcode Address Opcode Address + 1 Address Bus FFFF	1 1 1	Opcode Irrelevant Data Low Byte of Restart Vector
DES INS	3	1 2 3	Opcode Address Opcode Address + 1 Previous Stack Pointer Contents	1 1 1	Opcode Opcode of Next Instruction Irrelevant Data
INX DEX	3	1 2 3	Opcode Address Opcode Address + 1 Address Bus FFFF	1 1 1	Opcode Opcode of Next Instruction Low Byte of Restart Vector
PSHA PSHB	3	1 2 3	Opcode Address Opcode Address + 1 Stack Pointer	1 1 0	Opcode Opcode of Next Instruction Accumulator Data
TSX	3	1 2 3	Opcode Address Opcode Address + 1 Stack Pointer	1 1 1	Opcode Opcode of Next Instruction Irrelevant Data
TXS	3	1 2 3	Opcode Address Opcode Address + 1 Address Bus FFFF	1 1 1	Opcode Opcode of Next Instruction Low Byte of Restart Vector
PULA PULB	4	1 2 3 4	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer + 1	1 1 1 1	Opcode Opcode of Next Instruction Irrelevant Data Operand Data from Stack
PSHX	4	1 2 3 4	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer − 1	1 1 0 0	Opcode Irrelevant Data Index Register (Low Order Byte) Index Register (High Order Byte)
PULX	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer + 1 Stack Pointer + 2	1 1 1 1 1	Opcode Irrelevant Data Irrelevant Data Index Register (High Order Byte) Index Register (Low Order Byte)
RTS	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer + 1 Stack Pointer + 2	1 1 1 1 1	Opcode Irrelevant Data Irrelevant Data Address of Next Instruction (High Order Byte) Address of Next Instruction (Low Order Byte)
WAI	9	1 2 3 4 5 6 7 8 9	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer − 1 Stack Pointer − 2 Stack Pointer − 3 Stack Pointer − 4 Stack Pointer − 5 Stack Pointer − 6	1 1 0 0 0 0 0 0 0	Opcode Opcode of Next Instruction Return Address (Low Order Byte) Return Address (High Order Byte) Index Register (Low Order Byte) Index Register (High Order Byte) Contents of Accumulator A Contents of Accumulator B Contents of Condition Code Register

TABLE 14 — CYCLE-BY-CYCLE OPERATION (Sheet 5 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
INHERENT					
MUL	10	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Irrelevant Data
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Address Bus FFFF	1	Low Byte of Restart Vector
		5	Address Bus FFFF	1	Low Byte of Restart Vector
		6	Address Bus FFFF	1	Low Byte of Restart Vector
		7	Address Bus FFFF	1	Low Byte of Restart Vector
		8	Address Bus FFFF	1	Low Byte of Restart Vector
		9	Address Bus FFFF	1	Low Byte of Restart Vector
		10	Address Bus FFFF	1	Low Byte of Restart Vector
RTI	10	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Irrelevant Data
		3	Stack Pointer	1	Irrelevant Data
		4	Stack Pointer + 1	1	Contents of Condition Code Register from Stack
		5	Stack Pointer + 2	1	Contents of Accumulator B from Stack
		6	Stack Pointer + 3	1	Contents of Accumulator A from Stack
		7	Stack Pointer + 4	1	Index Register from Stack (High Order Byte)
		8	Stack Pointer + 5	1	Index Register from Stack (Low Order Byte)
		9	Stack Pointer + 6	1	Next Instruction Address from Stack (High Order Byte)
		10	Stack Pointer + 7	1	Next Instruction Address from Stack (Low Order Byte)
SWI	12	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Irrelevant Data
		3	Stack Pointer	0	Return Address (Low Order Byte)
		4	Stack Pointer − 1	0	Return Address (High Order Byte)
		5	Stack Pointer − 2	0	Index Register (Low Order Byte)
		6	Stack Pointer − 3	0	Index Register (High Order Byte)
		7	Stack Pointer − 4	0	Contents of Accumulator A
		8	Stack Pointer − 5	0	Contents of Accumulator B
		9	Stack Pointer − 6	0	Contents of Condition Code Register
		10	Stack Pointer − 7	1	Irrelevant Data
		11	Vector Address FFFA (Hex)	1	Address of Subroutine (High Order Byte)
		12	Vector Address FFFB (Hex)	1	Address of Subroutine (Low Order Byte)
RELATIVE					
BCC BHT BNE BLO BCS BLE BPL BHS BEQ BLS BRA BRN BGE BLT BVC BGT BMT BVS	3	1 2 3	Op Code Address Op Code Address +1 Address Bus FFFF	1 1 1	Op Code Branch Offset Low Byte of Restart Vector
BSR	6	1	Op Code Address	1	Op Code
		2	Op Code Address +1	1	Branch Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Subroutine Starting Address	1	Op Code of Next Instruction
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer -1	0	Return Address (High Order Byte)

3

FIGURE 24 — SPECIAL OPERATIONS



Legend:
RTN = Address of next instruction in Main Program to be executed upon return from subroutine
RTN_H = Most significant byte of Return Address
RTN_L = Least significant byte of Return Address
→ = Stack Pointer After Execution
K = 8-bit Unsigned Value

EPROM PROGRAMMING ROUTINE

```
PAGE  001  EPROM    .SA:1  EPROM  *** ROUTINE TO PROGRAM THE MC68701 EPROM ***

00001                        NAM     EPROM
00002                        OPT     Z01,LLEN=80
00003                        TTL     *** ROUTINE TO PROGRAM THE MC68701 EPROM **
00004
00005                    ************************************************************
00006                    *
00007                    *  E P R O M -- A  NON-REENTRANT  ROUTINE  TO  PROGRAM
00008                    *                THE MC68701 EPROM.
00009                    *
00010                    *                THE ROUTINE PROGRAMS THE MC68701 EPROM
00011                    *                STARTING  AT  ADDRESS  "PNTR"  FROM  A
00012                    *                BLOCK OF MEMORY  STARTING  AT  "IMBEG"
00013                    *                AND ENDING AT "IMEND".
00014                    *
00015                    *  CALLING CONVENTION:
00016                    *
00017                    *     JSR  EPROM
00018                    *
00019                    *  NOTES:
00020                    *
00021                    *     1.  THE ROUTINE EXPECTS FOUR DOUBLE BYTE  VALUES
00022                    *         TO  BE INITIALIZED PRIOR  TO  BEING  CALLED.
00023                    *         THESE VALUES ARE:
00024                    *
00025                    *           IMBEG = A DOUBLE BYTE ADDRESS  WHICH  POINTS
00026                    *                   TO THE FIRST BYTE TO  BE  PROGRAMMED
00027                    *                   INTO THE EPROM.
00028                    *
00029                    *           IMEND = A DOUBLE BYTE ADDRESS  WHICH  POINTS
00030                    *                   TO THE LAST BYTE TO BE PROGRAMED IN-
00031                    *                   INTO THE EPROM.
00032                    *
00033                    *           PNTR  = A DOUBLE BYTE ADDRESS  WHICH  POINTS
00034                    *                   TO THE FIRST BYTE IN THE EPROM TO BE
00035                    *                   PROGRAMMED.
00036                    *
00037                    *           WAIT  = A DOUBLE BYTE COUNTER VALUE WHICH IS
00038                    *                   A FUNCTION OF THE MCU INPUT FREQUEN-
00039                    *                   CY AND IS USED WITH THE OUTPUT  COM-
00040                    *                   PARE FUNCTION TO  GENERATE A 50 MSEC
00041                    *                   TIMEOUT.  IT IS EQUIVALENT TO
00042                    *
00043                    *                   50000 * (MCU INPUT FREQ) / 4 * 10**6
00044                    *
00045                    *                   VALUES FOR TYPICAL INPUT FREQS  ARE:
00046                    *
00047                    *                     WAIT                MCU INPUT FREQ
00048                    *                   -------------         --------------
00049                    *                   30615 ($7797)            2.45 MHZ
00050                    *                   50000 ($C350)            4.00 MHZ
00051                    *                   61375 ($EFBF)            4.91 MHZ
00052                    *
00053                    *     2.  IT IS ASSUMED THAT POWER (VPP) IS  AVAILABLE
00054                    *         TO THE RESET PIN FOR PROGRAMMING.
00055                    *
00056                    *     3.  THIS ROUTINE  PERFORMS  NO  ERROR  CHECKING.
00057                    *
00058                    *  Routine parameter initialization, such as stack pointer, etc., must be done prior to entry.
                         (Use of PROBug will ensure all needed initialization.)
```

3

EPROM PROGRAMMING ROUTINE

```
PAGE  002  EPROM   .SA:1  EPROM  *** ROUTINE TO PROGRAM THE MC68701 EPROM ***

00060
00061                    * E Q U A T E S
00062
00063           0008  A TCSR   EQU    $08       TIMER CONTROL/STAT REGISTER
00064           0009  A TIMER  EQU    $09       COUNTER REGISTER
00065           000B  A OUTCMP EQU    $0B       OUTPUT COMPARE REGISTER
00066           0014  A EPMCNT EQU    $14       RAM/EPROM CONTROL REGISTER
00067
00068                    * L O C A L    V A R I A B L E S
00069
00070A 0080                       ORG    $80
00071A 0080     0002  A IMBEG   RMB    2         START OF MEMORY BLOCK
00072A 0082     0002  A IMEND   RMB    2         LAST BYTE OF MEMORY BLOCK
00073A 0084     0002  A PNTR    RMB    2         FIRST BYTE OF EPROM TO BE PGM'D
00074A 0086     0002  A WAIT    RMB    2         COUNTER VALUE
00075
00076                    * E P R O M    S T A R T S    H E R E
00077
00078A 3000                       ORG    $3000
00079A 3000 DE 84     A EPROM  LDX    PNTR      SAVE CALLING ARGUMENT
00080A 3002 3C                   PSHX            RESTORE WHEN DONE
00081A 3003 DE 80     A        LDX    IMBEG     USE STACK
00082
00083A 3005 3C          EPR002 PSHX              SAVE POINTER ON STACK
00084A 3006 86 FE     A        LDAA   #$FE       REMOVE VPP, SET LATCH
00085A 3008 97 14     A        STAA   EPMCNT     PPC=1, PLC=0
00086A 300A A6 00     A        LDAA   X          MOVE DATA MEMORY-TO-LATCH
00087A 300C DE 84     A        LDX    PNTR       GET WHERE TO PUT IT
00088A 300E A7 00     A        STAA   X          STASH AND LATCH
00089A 3010 08                   INX             NEXT ADDR
00090A 3011 DF 84     A        STX    PNTR       ALL SET FOR NEXT
00091A 3013 86 FC     A        LDAA   #$FC       ENABLE EPROM POWER (VPP)
00092A 3015 97 14     A        STAA   EPMCNT     PPC=0, PLC=0
00093
00094                    * NOW WAIT FOR 50 MSEC TIMEOUT USING OUTPUT COMPARE.
00095
00096A 3017 DC 86     A        LDD    WAIT       GET CYCLE COUNTER
00097A 3019 D3 09     A        ADDD   TIMER      BUMP CURRENT VALUE
00098A 301B 7F 0008   A        CLR    TCSR       CLEAR OCF
00099A 301E DD 0B     A        STD    OUTCMP     SET OUTPUT COMPARE
00100A 3020 86 40     A        LDAA   #$40       NOW WAIT FOR OCF
00101
00102A 3022 95 08     A EPR004 BITA   TCSR
00103A 3024 27 FC 3022         BEQ    EPR004     NOT YET
00104A 3026 38                 PULX              SETUP FOR NEXT ONE
00105A 3027 08                 INX               NEXT
00106A 3028 9C 82     A        CPX    IMEND      MAYBE DONE
00107A 302A 23 D9 3005         BLS    EPR002     NOT YET
00108A 302C 86 FF     A        LDAA   #$FF       REMOVE VPP, INHIBIT LATCH
00109A 302E 97 14     A        STAA   EPMCNT     EPROM CAN NOW BE READ
00110A 3030 38                 PULX              RESTORE PNTR
00111A 3031 DF 84     A        STX    PNTR
00112A 3033 39                 RTS               THAT'S ALL
00113                          END
TOTAL ERRORS 00000--00000
```

ORDERING INFORMATION

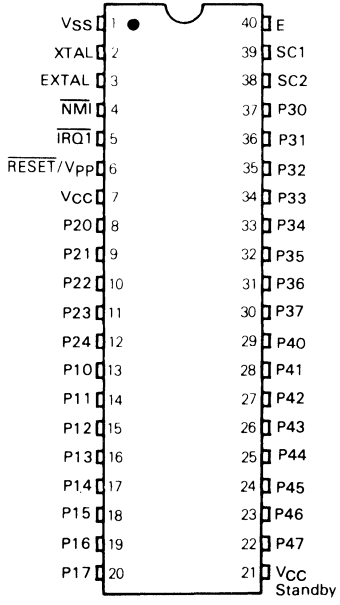
The following table provides generic information pertaining to the package type and temperature for the MC68701. The MCU device is available only in the 40-pin dual-in-line (DIP) package in the Cerdip and Plastic packages.

GENERIC INFORMATION

Frequency (MHz)	Temperature (Degrees C)	Cerdip Package (S Suffix)	Ceramic Package (L Suffix)
1.0	0 to 70	MC68701S	MC68701L
1.0	−40 to +85	MC68701CS	MC68701CL
1.25	0 to 70	MC68701S-1	MC68701L-1
1.25	−40 to +85	MC68701CS-1	MC68701CL-1
2.0	0 to 70	MC68B701S	MC68B701L

3

PIN ASSIGNMENT



MC68701U4

Advance Information

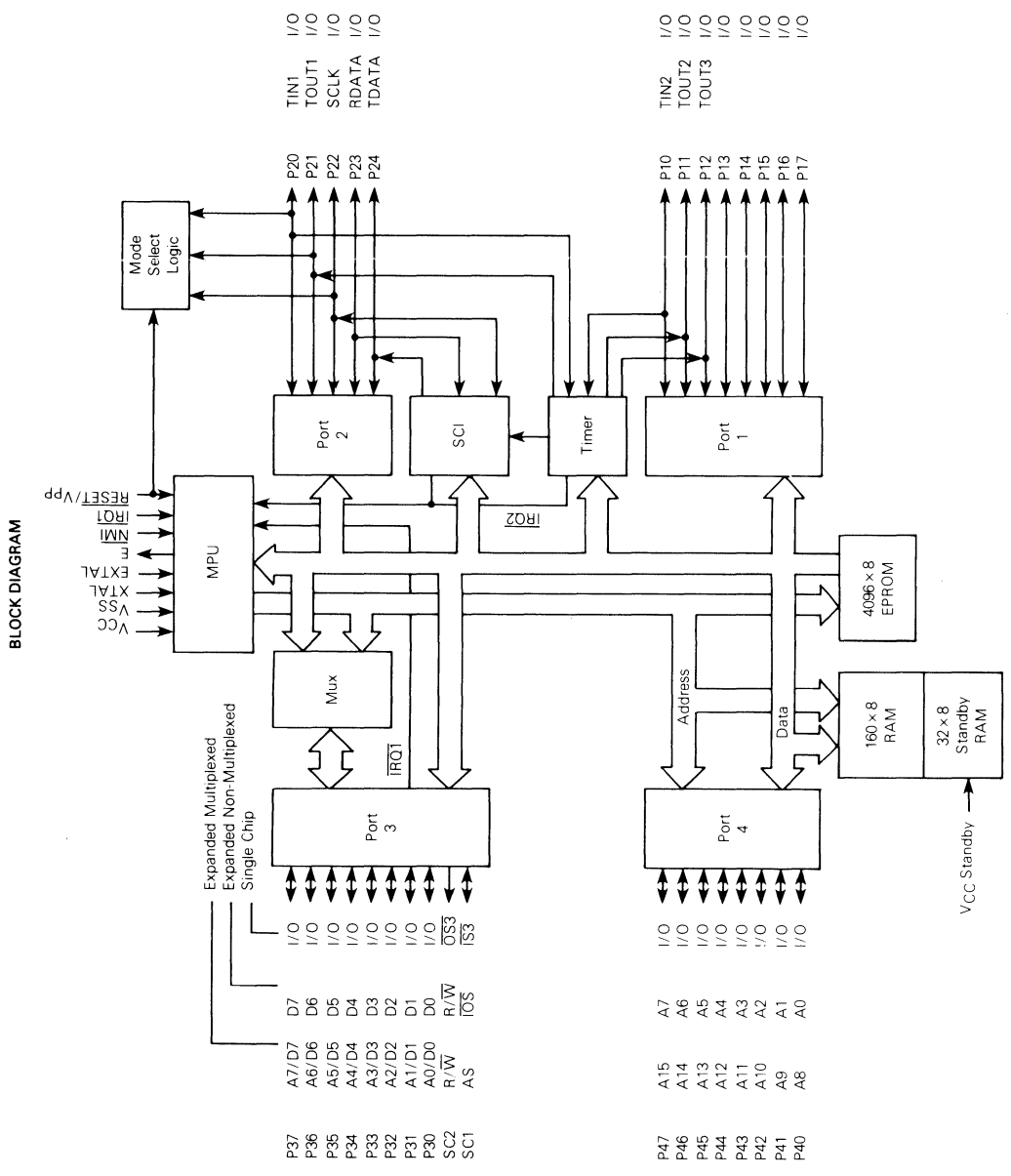
8-Bit EPROM Microcontroller/Microprocessor (MCU/MPU)

The MC68701U4 is an 8-bit single-chip EPROM microcontroller unit (MCU) which enhances the capabilities of the MC6801 and significantly enhances the capabilities of the M6800 Family of parts. It includes an MC6801 microprocessor unit (MPU) with direct object-code compatibility and upward object-code compatibility with the MC6800. Execution times of key instructions have been improved over the MC6800 and the new instructions found on the MC6801 are included. The MCU can function as a monolithic microcontroller or can be expanded to a 64K byte address space. It is TTL compatible and requires one +5-volt power supply for nonprogramming operation. An additional V_{PP} power supply is needed for EPROM programming. On-chip resources include 4096 bytes of EPROM, 192 bytes of RAM, a serial communications interface (SCI), parallel I/O, and a 16-bit six-function programmable timer.

- Enhanced MC6800 Instruction Set
- Upward Source and Object Code Compatibility with the MC6800, MC6801, and MC6801U4
- Bus Compatibility with the M6800 Family
- 8×8 Multiply Instruction
- Single-Chip or Expanded Operation of 64K Byte Address Space
- Internal Clock Generator with Divide-by-Four Output
- Serial Communications Interface (SCI)
- 16-Bit Six-Function Programmable Timer
- Three Output Compare Functions
- Two Input Capture Functions
- Counter Alternate Address
- 4096 Bytes of Use EPROM
- 192 Bytes of RAM
- 32 Bytes of RAM Retainable During Power Down
- 29 Parallel I/O and Two Handshake Control Lines
- $\overline{NMI}$ Inhibited Until Stack Load

This document contains information on a new product. Specifications and information herein are subject to change without notice.

BLOCK DIAGRAM



3

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	T_{in}	−0.3 to +7.0	V
Operating Temperature Range	T_A	0 to 70	°C
Storage Temperature Range Programmed Unprogrammed	T_{stg}	 −40 to +85 −55 to +150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid applications of any voltage higher than maximum rated voltages to this high-impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $GND \leq (V_{in}$ or $V_{out}) \leq V_{CC}$.

Unused inputs must always be tied to an appropriate logic voltage level (e.g., either GND or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Rating
Thermal Resistance Cerdip	θ_{JA}	65	°C/W

3

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A

CONTROL TIMING ($V_{CC} = 5.0$ V ±5%, $V_{SS} = 0$)

Characteristic	Symbol	MC68701U4		MC68701U4-1		Unit
		Min	Max	Min	Max	
Frequency of Operation	f_O	0.5	1.0	0.5	1.25	MHz
Crystal Frequency	f_{XTAL}	2.0	4.0	2.0	5.0	MHz
External Oscillator Frequency	$4\ f_O$	2.0	4.0	2.0	5.0	MHz
Crystal Oscillator Startup Time	t_{rc}	—	100	—	100	ms
Processor Control Setup Time	t_{PCS}	200	—	170	—	ns

DC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 Vdc ±5%, V_{SS} = 0)

Characteristic		Symbol	Min	Typ	Max	Unit
Input High Voltage	$\overline{RESET}$	V_{IH}	V_{SS} + 4.0	–	V_{CC}	V
	Other Inputs*		V_{SS} + 2.0	–	V_{CC}	
Input Low Voltage	$\overline{RESET}$	V_{IL}	V_{SS} – 0.3	–	V_{SS} + 0.4	V
	Other Inputs*		V_{SS} – 0.3	–	V_{SS} + 0.8	
Input Current (V_{in} = 0 to 2.4 V) See Note	Port 4	I_{in}	–	–	0.5	mA
	SC1		–	–	0.8	
Input Current (V_{in} = 0 to 5.25 V)	$\overline{NMI}$, $\overline{IRQ1}$	I_{in}	–	–	2.5	μA
Input Current (V_{in} = 0 to 0.4 V) See Note	$\overline{RESET}$/V_{PP}	I_{in}	–	– 2.0	–	mA
(V_{in} = 4.0 V to V_{CC})			–	–	8.0	
Hi-Z (Off State) Input Current (V_{in} = 0.5 to 2.4 V)	P10-P17, P20-P24, P30-P37	I_{TSI}	–	–	10	μA
Output High Voltage (I_{load} = – 65 μA, V_{CC} = min)	P40-P47, SC1, SC2	V_{OH}	V_{SS} + 2.4	–	–	V
(I_{load} = – 100 μA, V_{CC} = min)	Other Outputs		V_{SS} + 2.4	–	–	
Output Low Voltage (I_{load} = 2.0 mA, V_{CC} = min)	All Outputs	V_{OL}	–	–	V_{SS} + 0.5	V
Darlington Drive Current (V_O = 1.5 V)	P10-P17	I_{OH}	1.0	–	4.0	mA
Internal Power Dissipation (measured at T_A = 0°C in Steady-State Operation)		P_{INT}	–	–	1200	mW
Input Capacitance (V_{in} = 0, T_A = 25°C, f_o = 1.0 MHz)	P30-P37, P40-P47, SC1	C_{in}	–	–	12.5	pF
	Other Inputs		–	–	10.0	
V_{CC} Standby	Power Down	V_{SBB}	4.0	–	5.25	V
	Power Up	V_{SB}	4.75	–	5.25	
Standby Current	Power Down	I_{SBB}	–	–	3.0	mA
Programming Time (Per Byte) (T_A = 25°C)		t_{PP}	25	–	50	ms
Programming Voltage (T_A = 25°C)		V_{PP}	20.0	21.0	22.0	V
Programming Current (V_{RESET} = V_{PP}) (T_A = 25°C)		I_{PP}	–	30.0	50.0	mA

*Except Mode Programming Levels; See Figure 16.

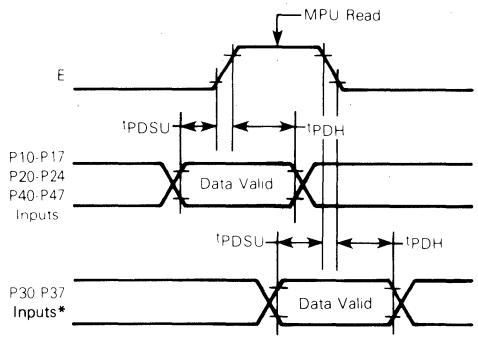
NOTE: $\overline{RESET}$/V_{PP}, V_{IL}, and I_{in} values differ from MC6801U4 values.

PERIPHERAL PORT TIMING (Refer to Figures 1-4)

Characteristics	Symbol	Min	Typ	Max	Unit
Peripheral Data Setup Time	t_{PDSU}	200	–	–	ns
Peripheral Data Hold Time	t_{PDH}	200	–	–	ns
Delay Time, Enable Positive Transition to OS3 Negative Transition	t_{OSD1}	–	–	350	ns
Delay Time, Enable Positive Transition to OS3 Positive Transition	t_{OSD2}	–	–	350	ns
Delay Time, Enable Negative Transition to Peripheral Data Valid	t_{PWD}	–	–	350	ns
Delay Time, Enable Negative Transition to Peripheral CMOS Data Valid	t_{CMOS}	–	–	2.0	μs
Input Strobe Pulse Width	t_{PWIS}	200	–	–	ns
Input Data Hold Time	t_{IH}	50	–	–	ns
Input Data Setup Time	t_{IS}	20	–	–	ns

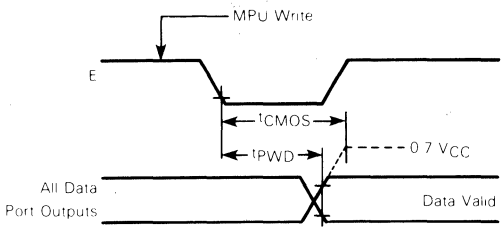
3

FIGURE 1 — DATA SETUP AND HOLD TIMES
(MPU READ)



* Port 3 non-latched operation (Latch enable = 0)

FIGURE 2 — DATA SETUP AND HOLD TIMES
(MPU WRITE)



NOTES
1. 10 k pullup resistor required for port 2 to reach 0.7 V_{CC}
2. Not applicable to P21
3. Port 4 cannot be pulled above V_{CC}

3

FIGURE 3 — PORT 3 OUTPUT STROBE TIMING
(SINGLE-CHIP MODE)

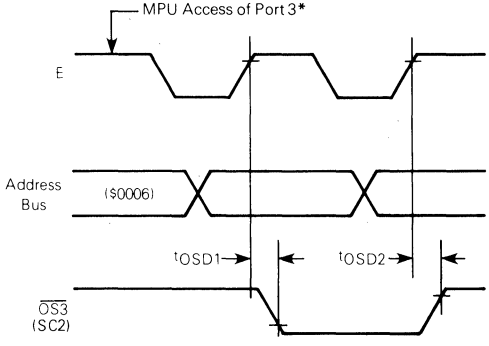
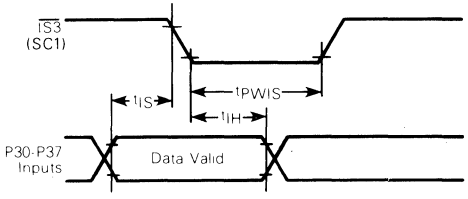


FIGURE 4 — PORT 3 LATCH TIMING
(SINGLE-CHIP MODE)



* Access matches output strobe select (OSS = 0, a read; OSS = 1, a write)

NOTE: Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted.

FIGURE 5 — CMOS LOAD

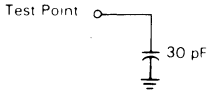
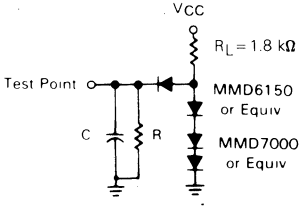


FIGURE 6 — TIMING TEST
LOAD PORTS 1, 2, 3, AND 4



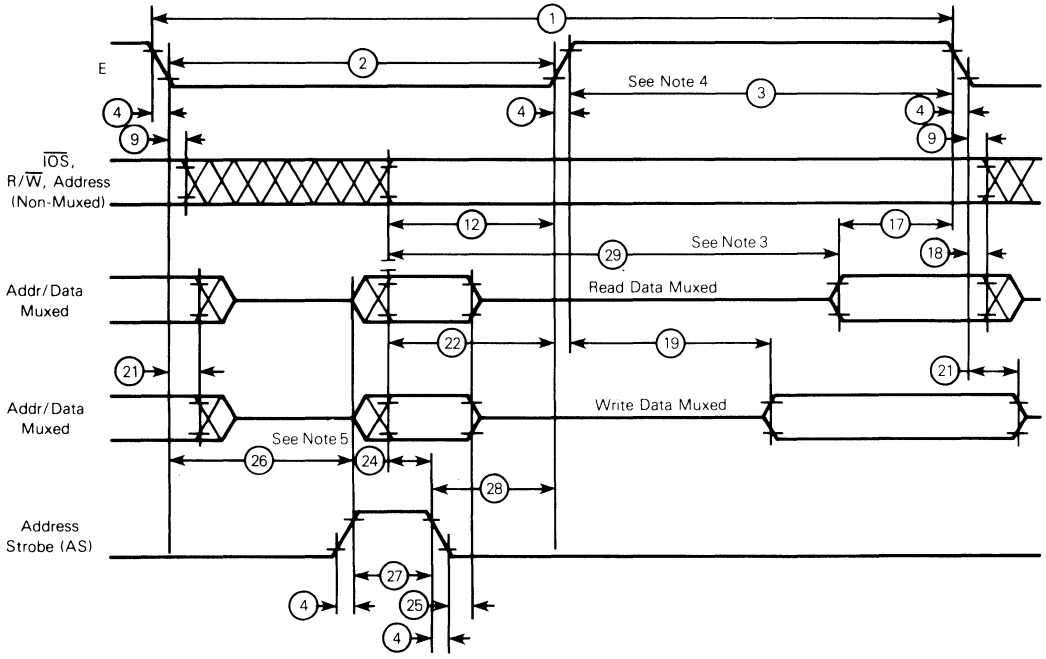
C = 90 pF for P30-P37, P40-P47, E, SC1, SC2
= 30 pF for P10-P17, P20-P24
R = 37 kΩ for P40-P47, SC1, SC2
= 24 kΩ for P10-P17, P20-P24, P30-P37, E

BUS TIMING (See Notes 1 and 2, and Figure 7)

Ident. Number	Characteristics	Symbol	MC68701U4		MC68701U4-1		Unit
			Min	Max	Min	Max	
1	Cycle Time	t_{cyc}	1.0	2.0	0.8	2.0	μs
2	Pulse Width, E Low	PW_{EL}	430	1000	360	1000	ns
3	Pulse Width, E High	PW_{EH}	450	1000	360	1000	ns
4	Clock Rise and Fall Time	t_r, t_f	–	25	–	25	ns
9	Address Hold Time	t_{AH}	20	–	20	–	ns
12	Non-Muxed Address Valid Time to E*	t_{AV}	200	–	150	–	ns
17	Read Data Setup Time	t_{DSR}	80	–	70	–	ns
18	Read Data Hold Time	t_{DHR}	10	–	10	–	ns
19	Write Data Delay Time	t_{DDW}	–	225	–	200	ns
21	Write Data Hold Time	t_{DHW}	20	–	20	–	ns
22	Muxed Address Valid Time to E Rise*	t_{AVM}	160	–	120	–	ns
24	Muxed Address Valid Time to AS Fall*	t_{ASL}	40	–	30	–	ns
25	Muxed Address Hold Time	t_{AHL}	20	–	20	–	ns
26	Delay Time, E to AS Rise*	t_{ASD}	200	–	170	–	ns
27	Pulse Width, AS High*	PW_{ASH}	100	–	80	–	ns
28	Delay Time, AS to E Rise*	t_{ASED}	90	–	70	–	ns
29	Usable Access Time* (See Note 3)	t_{ACC}	530	–	410	–	ns

* At specified cycle time.

FIGURE 7 — BUS TIMING



NOTES:
1. Voltage levels shown are $V_L \leq 0.5$ V, $V_H \geq 2.4$ V, unless otherwise specified.
2. Measurement points shown are 0.8 V and 2.0 V, unless otherwise specified.
3. Usable access time is computed by 22 + 3 – 17.
4. Memory devices should be enabled only during E high to avoid port 3 bus contention.
5. Item 26 is different from the MC6801, but it is upward compatible.

INTRODUCTION

The MC68701U4 is an 8-bit monolithic microcontroller which an be configured to function in a wide variety of applications. The facility which provides this extraordinary flexibility is its ability to be hardware programmed into seven different operating modes. The operating mode controls the configuration of 18 of the 40 MCU pins, available on-chip resources, memory map, location (internal or external) of interrupt vectors, and type of external bus. The configuration of the remaining 22 pins is not dependent on the operating mode.

Twenty-nine pins are organized as three 8-bit ports and one 5-bit port. Each port consists of at least a data register and a write-only data direction register. The data direction register is used to define whether corresponding bits in the data register are configured as an input (clear) or output (set).

The term "port" by itself refers to all of the hardware associated with the port. When the port is used as a "data port" or "I/O port," it is controlled by the port data direction register and the programmer has direct access to the port pins using the port data register. Port pins are labeled as Pij where i identifies one of four ports and j indicates the particular bit.

The microprocessor unit (MPU) is an enhanced MC6800 MPU with additional capabilities and greater throughput. It is upward source and object code compatible with the MC6800 and the MC6801. The programming model is depicted in Figure 8 where accumulator D is a concatenation of accumulators A and B. A list of new operations added to the MC6800 instruction set are shown in Table 1.

The basic difference between the MC6801U4 and the MC68701U4 is that the MC6801U4 has an on-chip ROM while the MC68701U4 has an on-chip EPROM. The

3

FIGURE 8 — PROGRAMMING MODEL

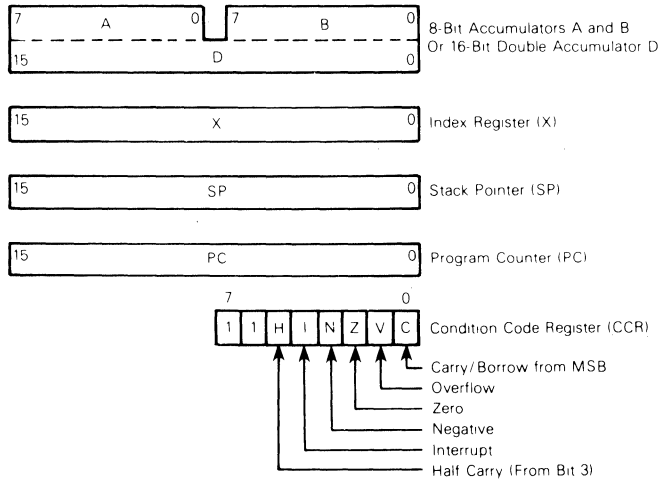


TABLE 1 — NEW INSTRUCTIONS

Instruction	Description
ABX	Unsigned addition of accumulator B to index register
ADDD	Adds (without carry) the double accumulator to memory and leaves the sum in the double accumulator
ASLD or LSLD	Shifts the double accumulator left (towards MSB) one bit, the LSB is cleared, and the MSB is shifted into the C bit
BHS	Branch if higher or same, unsigned conditional branch (same as BCC)
BLO	Branch if lower, unsigned conditional branch (same as BCS)
BRN	Branch never
JSR	Additional addressing mode direct
LDD	Loads double accumulator from memory
LSL	Shifts memory or accumulator left (towards MSB) one bit, the LSB is cleared, and the MSB is shifted into the C bit (same as ASL)
LSRD	Shifts the double accumulator right (towards LSB) one bit, the MSB is cleared, and the LSB is shifted into the C bit
MUL	Unsigned multiply, multiplies the two accumulators and leaves the product in the double accumulator
PSHX	Pushes the index register to stack
PULX	Pulls the index register from stack
STD	Stores the double accumulator to memory
SUBD	Subtracts memory from the double accumulator and leaves the difference in the double accumulator
CPX	Internal processing modified to permit its use with any conditional branch instruction

MC68701U4 is pin and code compatible with the MC6801U4 and can be used to emulate the MC6801U4, allowing easy software development using the on-chip EPROM. Software developed using the MC68701U4 can then be masked into the MC6801U4 ROM.

In order to support the on-chip EPROM, the MC68701U4 differs from the MC6801U4 as follows:

(1) Mode 0 in the MC6801U4 is a test mode only, while in the MC68701U4 mode 0 is also used to program the on-chip EPROM.
(2) The MC68701U4 RAM/EPROM control register has two bits used to control the EPROM in mode 0 that are not defined in the MC6801U4 RAM control register.
(3) The $\overline{\text{RESET}}$/V_{PP} pin in the MC68701U4 is dual purpose, used to supply EPROM power as well as to reset the device; while in the MC6801U4 the pin is called $\overline{\text{RESET}}$ and is used only to reset the device.

OPERATING MODES

The MC68701U4 provides seven different operating modes (modes 0 through 3 and 5 through 7). The operating modes are hardware selectable and determine the device memory map, the configuration of port 3, port 4, SC1, SC2, and the physical location of the interrupt vectors.

FUNDAMENTAL MODES

The seven operating modes (0-3, 5-7) can be grouped into three fundamental modes which refer to the type of bus it supports: single chip, expanded non-multiplexed, and expanded multiplexed. Single chip is mode 7, expanded non-multiplexed is mode 5, and the remaining 5 are expanded multiplexed modes. Table 2 summarizes the characteristics of the operating modes.

SINGLE-CHIP MODE (7) — In the single-chip mode, the four MCU ports are configured as parallel input/output data ports, as shown in Figure 9. The MCU functions as a monolithic microcontroller in this mode without external address or data buses. A maximum of 29 I/O lines and two port 3 control lines are provided. Peripherals or another MCU can be interfaced to port 3 in a loosely coupled dual-processor configuration, as shown in Figure 10.

3

TABLE 2 — SUMMARY OF OPERATING MODES

Single-Chip (Mode 7)
192 bytes of RAM, 4096 bytes of EPROM
Port 3 is a parallel I/O port with two control lines
Port 4 is a parallel I/O port
Expanded Non-Multiplexed (Mode 5)
192 bytes of RAM, 4096 bytes of EPROM
256 bytes of external memory space
Port 3 is an 8-bit data bus
Port 4 is an input port/address bus
Expanded Multiplexed (Modes 0, 1, 2, 3, 6)
Four memory space options (total 64K address space)
(1) Internal RAM and EPROM with partial address bus (mode 1)
(2) Internal RAM, no EPROM (mode 2)
(3) Extended addressing of internal I/O and RAM
(4) Internal RAM and EPROM with partial address bus (mode 6)
Port 3 is multiplexed address/data bus
Port 4 is address bus (inputs/address in mode 6)
Test/Program mode (mode 0):
May be used to test internal RAM and EPROM
May be used to test ports 3 and 4 as I/O ports by writing into mode 7
Used to program EPROM
Only modes 5, 6, and 7 can be irreversibly entered from mode 0
Resources Common to All Modes
Reserved register area
Port 1 input/output operation
Port 2 input/output operation
Timer operation
Serial communications interface operation

FIGURE 9 — SINGLE-CHIP MODE

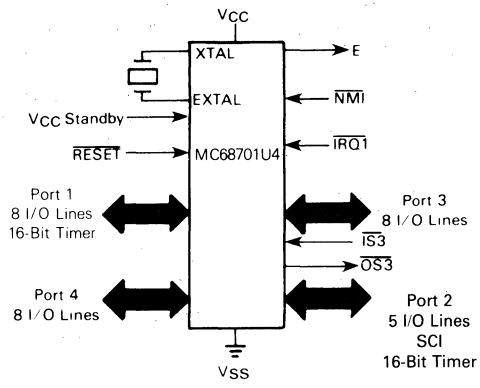
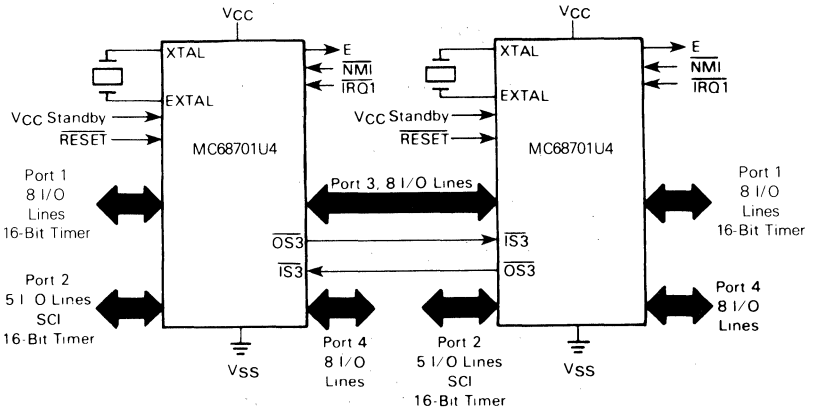


FIGURE 10 — SINGLE-CHIP DUAL PROCESSOR CONFIGURATION



EXPANDED NON-MULTIPLEXED MODE (5) — A modest amount of external memory space is provided in the expanded non-multiplexed mode while significant on-chip resources are retained. Port 3 functions as an 8-bit bidirectional data bus and port 4 is configured initially as an input data port. Any combination of the eight least significant address lines may be obtained by writing to the port 4 data direction register. Stated alternatively, any combination of A0 to A7 may be provided while retaining the remainder as input data lines. Internal pullup resistors pull the port 4 lines high until the port is configured.

Figure 11 illustrates a typical system configuration in the expanded non-multiplexed mode. The MCU interfaces directly with M6800 Family parts and can access 256 bytes of external address space at $100 through $1FF. IOS provides an address decode of external memory ($100-$1FF) and can be used as a memory-page select or chip-select line.

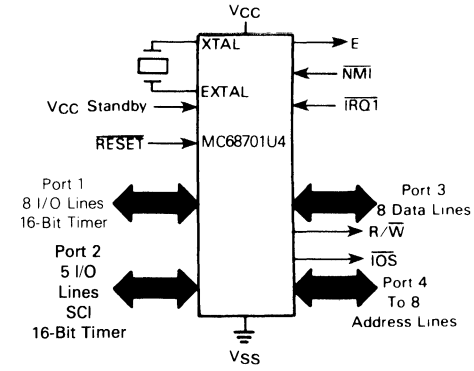
EXPANDED MULTIPLEXED MODES (0, 1, 2, 3, 6) — A 64K byte memory space is provided in the expanded multiplexed modes. In each of the expanded multiplexed modes, port 3 functions as a time multiplexed address/data bus with address valid on the negative edge of address strobe (AS) and data valid while E is high. In modes 0, 2, and 3, port 4 provides address lines A8 to A15. In modes 1 and 6, however, port 4 initially is configured at reset as an input data port. The port 4 data direction register can then be changed to provide any combination of address lines A8 to A15. Stated alternatively, any subset of A8 to A15 can be provided while retaining the remaining port 4 lines as input data lines. Internal pullup resistors pull the port 4 lines high until software configures the port.

In mode 0, the reset and interrupt vectors are located at $BFF0-$BFFF. In addition, the internal and external data

buses are connected, so there must be no memory map overlap in order to avoid potential bus conflicts. By writing the PC0-PC2 bits in the port 2 data register, modes 5, 6, and 7 can be irreversibly entered from mode 0. Mode 0 is used to program the on-chip EPROM.

Figure 12 depicts a typical configuration for the expanded multiplexed modes. Address strobe can be used to control a transparent D-type latch to capture addresses A0-A7, as shown in Figure 13. This allows port 3 to function as a data bus when E is high.

FIGURE 11 — EXPANDED NON-MULTIPLEXED CONFIGURATION



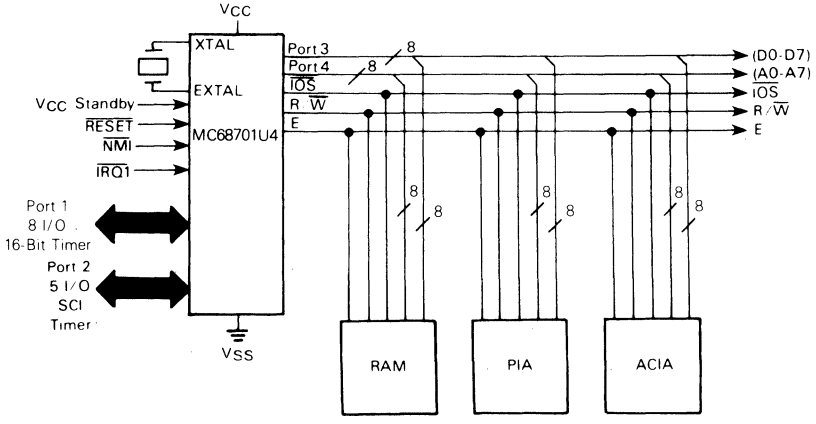
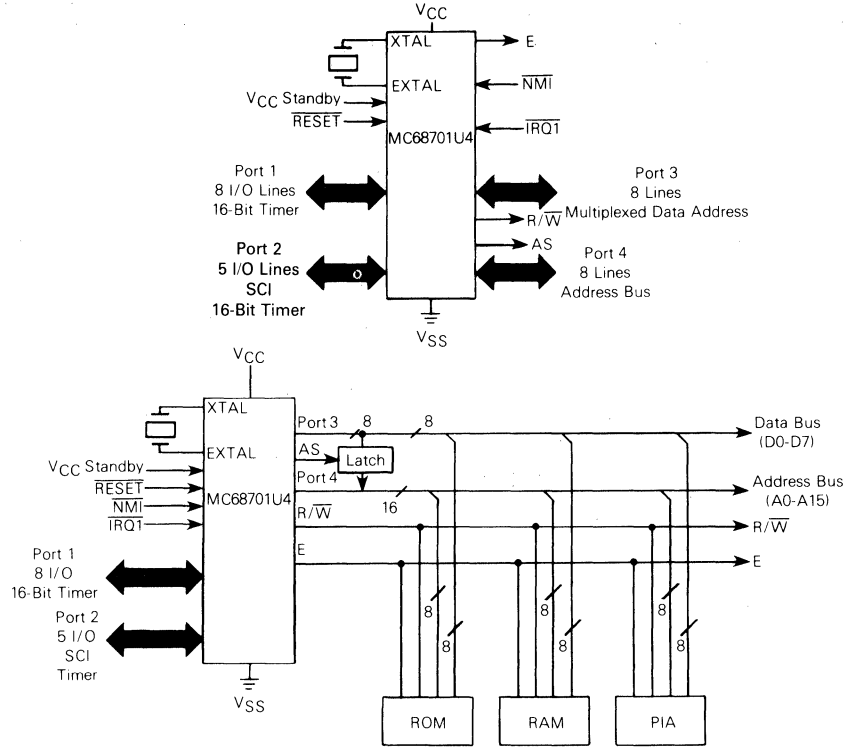
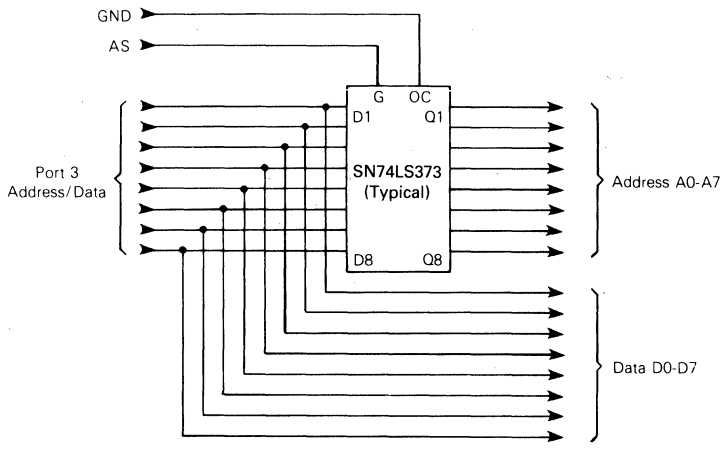


FIGURE 12 — EXPANDED MULTIPLEXED CONFIGURATION



NOTE: To avoid data bus (port 3) contention in the expanded multiplexed modes, memory devices should be enabled only during E high time.

FIGURE 13 — TYPICAL LATCH ARRANGEMENT



PROGRAMMING THE MODE

The operating mode is determined at $\overline{\text{RESET}}$ by the levels asserted on P22, P21, and P20. These levels are latched into PC2, PC1, and PC0 of the program control register on the positive edge of $\overline{\text{RESET}}$. The operating mode may be read from the port 2 data register, as shown below, and programming levels and timing must be met a shown in Figure 14. A brief outline of the operating modes is shown in Table 3.

PORT 2 DATA REGISTER

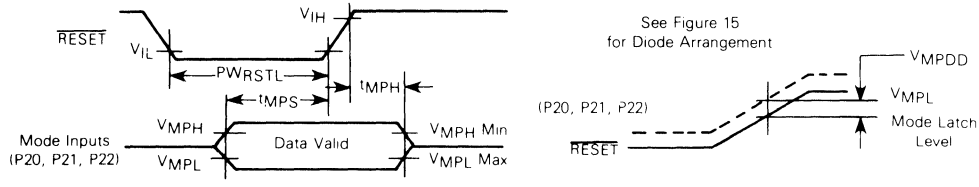
7	6	5	4	3	2	1	0	
PC2	PC1	PC0	P24	P23	P22	P21	P20	$03

Circuitry to provide the programming levels is dependent primarily on the normal system usage of the three pins. If configured as outputs, the circuit shown in Figure 15 may be used; otherwise, three-state buffers can be used to provide isolation while programming the mode. Note that if diodes are used to program the mode, the diode forward voltage drop must not exceed the V_{MPDD} minimum.

MEMORY MAPS

The MC68701U4 can provide up to 64K byte address space depending on the operating mode. A memory map for each operating mode is shown in Figure 16. The first 32 locations of each map are reserved for the internal register area, as shown in Table 4, with exceptions as indicated.

FIGURE 14 — MODE PROGRAMMING TIMING



MODE PROGRAMMING (Refer to Figure 14)

Characteristic	Symbol	Min	Max	Unit
Mode Programming Input Voltage Low (for T_A = 0 to 70°C)	V_{MPL}	–	1.8	V
Mode Programming Input Voltage High	V_{MPH}	4.0	–	V
Mode Programming Diode Differential (If Diodes are Used) (for T_A = 0 to 70°C)	V_{MPDD}	0.6	–	V
$\overline{\text{RESET}}$ Low Pulse Width	PW_{RSTL}	3.0	–	E Cycles
Mode Programming Setup Time	t_{MPS}	2.0	–	E Cycles
Mode Programming Hold Time $\overline{\text{RESET}}$ Rise Time ≥ 1 μs $\overline{\text{RESET}}$ Rise Time < 1 μs	t_{MPH}	0 100	– –	ns

NOTE: For T_A = −40 to 85°C, Maximum V_{MPL} = 1.7, and Minimum V_{MPDD} = 0.4.

TABLE 3 — MODE SELECTION SUMMARY

Mode	P22 PC2	P21 PC1	P20 PC0	EPROM	RAM	Interrupt Vectors	Bus Mode	Operating Mode
7	H	H	H	I	I	I	I	Single Chip
6	H	H	L	I	I	I	MUX(2, 3)	Multiplexed/Partial Decode
5	H	L	H	I	I	I	NMUX(2, 3)	Non-Multiplexed/Partial Decode
4	H	L	L	–	–	–	–	Undefined(4)
3	L	H	H	E	I	E	MUX(1, 5)	Multiplexed/RAM
2	L	H	L	E	I	E	MUX(1)	Multiplexed/RAM
1	L	L	H	I	I	E	MUX(2, 3)	Multiplexed/RAM and EPROM
0	L	L	L	I	I	E	MUX(1)	Multiplex ed Test/Programming

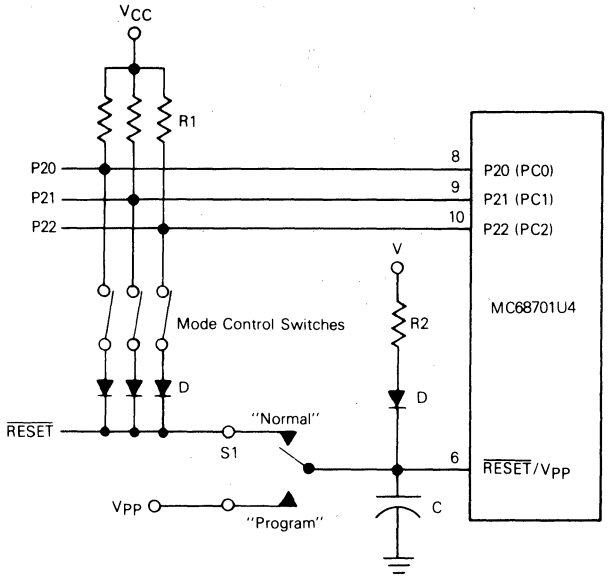
LEGEND

I — Internal
E — External
MUX — Multiplexed
NMUX — Non-Multiplexed
L — Logic "0"
H — Logic "1"

NOTES:

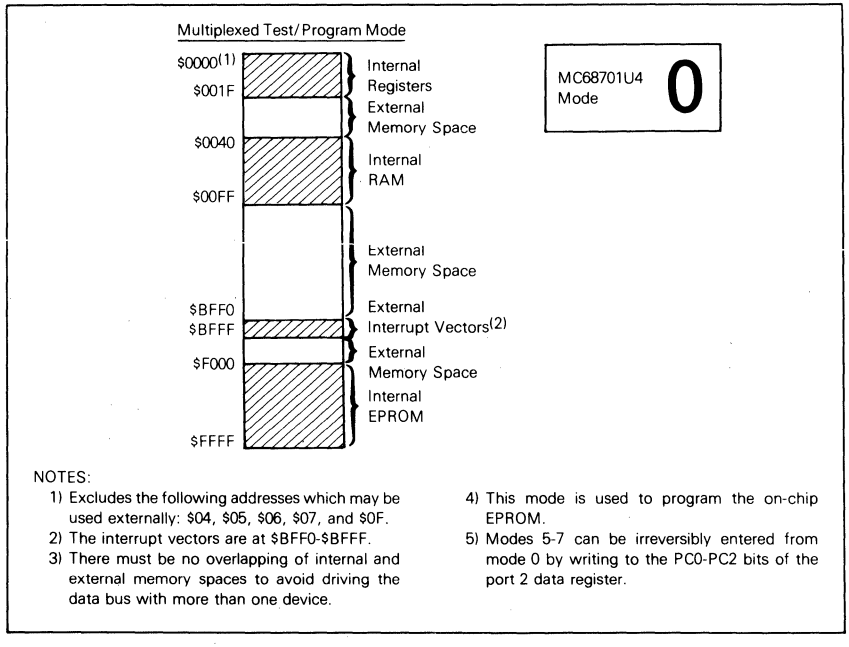
1. Addresses associated with ports 3 and 4 are considered external in modes 0, 2, and 3.
2. Addresses associated with port 3 are considered external in modes 1, 5, and 6.
3. Port 4 default is user data input; address output is optional by writing to port 4 data direction register.
4. Mode 4 is a non-user mode and should not be used as an operating mode.
5. Mode 3 has the internal RAM and internal registers relocated at $D000-$D0FF.

FIGURE 15 — TYPICAL MODE PROGRAMMING CIRCUIT



NOTES:
1. Mode 0 as shown (switches closed).
2. R1 = 10 kilohms (typical).
3. The RESET time constant is equal to RC where R is the equivalent parallel resistance of R2 and the number of resistors (R1) placed in the circuit by closed mode contol switches.
4. D = 1N914, 1N4001 in the 0 to 70°C range
   D = 1N270, MBD201 in the −40 to 85°C range
5. If V = V_{CC}, the R2 = 50 ohms (typical) to meet V_{IH} for the RESET/V_{PP} pin. V = V_{CC} is also compatible with MC6801U4. The RESET time constant in this case is approximately R2 × C.
6. Switch S1 allows selection of normal (RESET) or programming (V_{PP}) as the input to the RESET/V_{PP} pin. During switching, the input level is held at a value determined by a diode (D), resistor (R2) and input voltage (V).
7. While S1 in the "Program" position, RESET should not be asserted.
8. From powerup, RESET must be held low for at least t_{RC}. The capacitor, C, is shown for conceptual purposes only and is on the order of 1000 μF for the circuit shown. Typically, a buffer with an RC input will be used to drive RESET, eliminating the need for the larger capacitor.
9. Diode V_f should not exceed V_{MPDD} min.

FIGURE 16 — MEMORY MAPS (Sheet 1 of 3)

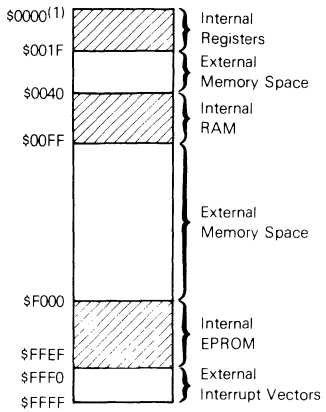


NOTES:
1) Excludes the following addresses which may be used externally: $04, $05, $06, $07, and $0F.
2) The interrupt vectors are at $BFF0-$BFFF.
3) There must be no overlapping of internal and external memory spaces to avoid driving the data bus with more than one device.
4) This mode is used to program the on-chip EPROM.
5) Modes 5-7 can be irreversibly entered from mode 0 by writing to the PC0-PC2 bits of the port 2 data register.

FIGURE 16 — MEMORY MAPS (Sheet 2 of 3)

MC68701U4 Mode 1

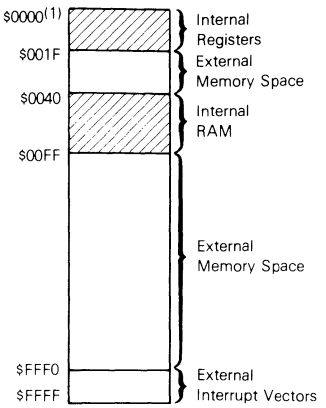
Multiplexed/RAM and EPROM



NOTES:
1) Excludes the following addresses which may be used externally: $04, $06, and $0F.
2) Internal EPROM addresses $FFF0 to $FFFF are not usable.
3) Address lines A8-A15 will not contain addresses until the data direction register for port 4 has been written with "1s" in the appropriate bits. These address lines will assert "1s" until made outputs by writing the data direction register.

MC68701U4 Mode 2

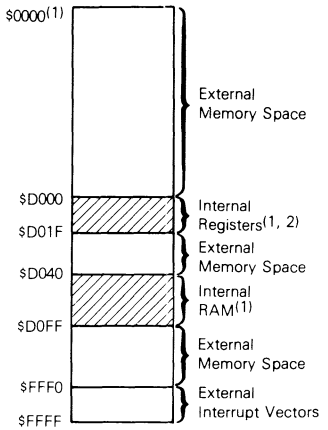
Multiplexed/RAM



NOTE:
1) Excludes the following addresses which may be used externally: $04, $05, $06, $07, and $0F.

MC68701U4 Mode 3

Multiplexed/RAM



NOTES:
1) Relocating the internal registers and the internal RAM to high memory allows the processor to make use of direct addressing.
2) Excludes the following addresses which may be used externally: $D004, $D005, $D006, $D007, and $D00F.

FIGURE 16 — MEMORY MAPS (Sheet 3 of 3)

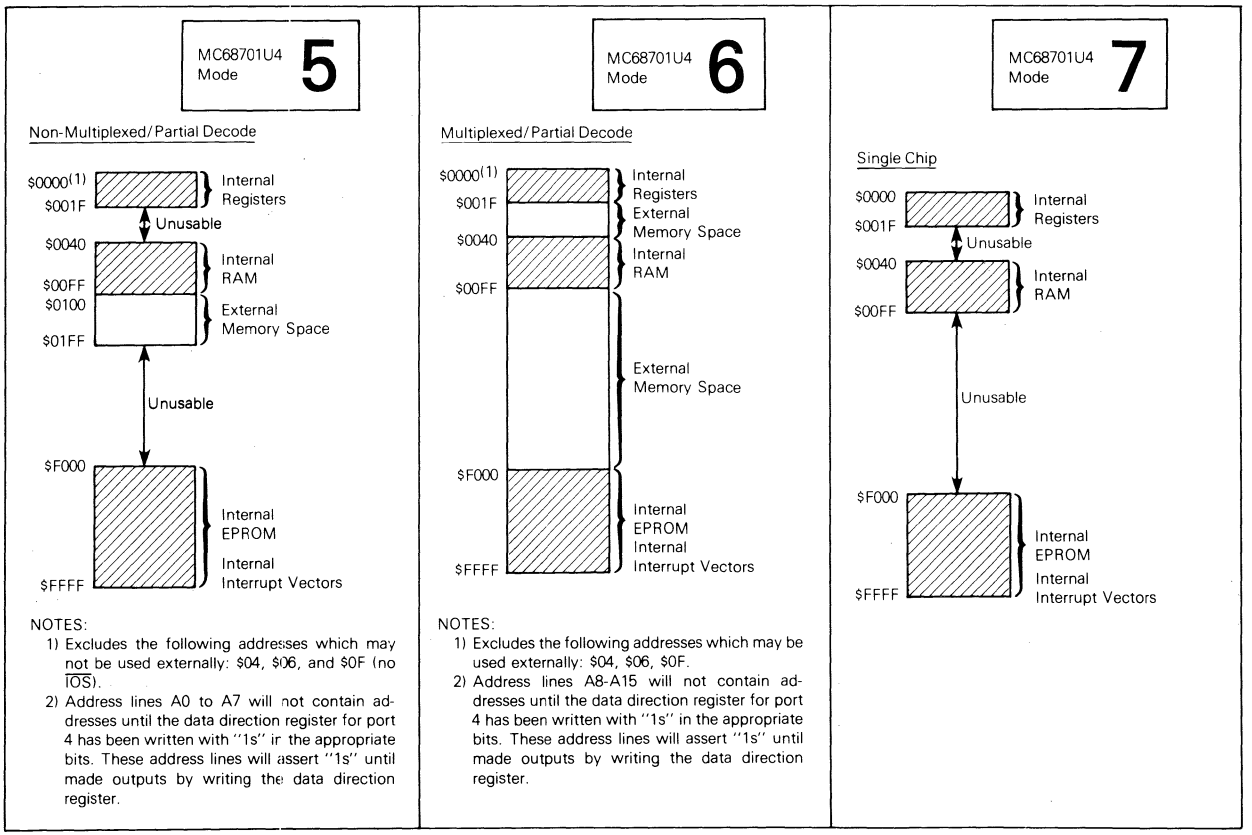


TABLE 4 — INTERNAL REGISTER AREA

Register	Address
Port 1 Data Direction Register***	00
Port 2 Data Direction Register***	01
Port 1 Data Register	02
Port 2 Data Register	03
Port 3 Data Direction Register***	04*
Port 4 Data Direction Register***	05**
Port 3 Data Register	06*
Port 4 Data Register	07**
Timer Control and Status Register	08
Counter (High Byte)	09
Counter (Low Byte)	0A
Output Compare Register (High Byte)	0B
Output Compare Register (Low Byte)	0C
Input Capture Register (High Byte)	0D
Input Capture Register (Low Byte)	0E
Port 3 Control and Status Register	0F*
Rate and Mode Control Register	10
Transmit/Receive Control and Status Register	11
Receive Data Register	12
Transmit Data Register	13
RAM Control Register	14
Counter Alternate Address (High Byte)	15
Counter Alternate Address (Low Byte)	16
Timer Control Register 1	17
Timer Control Register 2	18
Timer Status Register	19
Output Compare Register 2 (High Byte)	1A
Output Compare Register 2 (Low Byte)	1B
Output Compare Register 3 (High Byte)	1C
Output Compare Register 3 (Low Byte)	1D
Input Capture Register 2 (High Byte)	1E
Input Capture Register 2 (Low Byte)	1F

*External addresses in modes 0, 1, 2, 3, 5, and 6; cannot be accessed in mode 5 (no IOS)
**External addresses in modes 0, 2, and 3
***1 = Output, 0 = Input

MC68701U4 INTERRUPTS

The M6801 Family supports two types of interrupt requests: maskable and non-maskable. A non-maskable interrupt ($\overline{NMI}$) is always recognized and acted upon at the completion of the current instruction. Maskable interrupts are controlled by the condition code register I bit and by individual enable bits. The I bit controls all maskable interrupts. Of the maskable interrupts, there are two types: $\overline{IRQ1}$ and $\overline{IRQ2}$. The programmable timer and serial communications interface use an internal $\overline{IRQ2}$ interrupt line, as shown in the block diagram. External devices and IS3 use $\overline{IRQ1}$. An $\overline{IRQ1}$ interrupt is serviced before $\overline{IRQ2}$ if both are pending.

NOTE

After reset, an $\overline{NMI}$ will not be serviced until the first program load of the stack pointer. Any $\overline{NMI}$ generated before this load will be remembered by the processor and serviced subsequent to the stack pointer load.

All $\overline{IRQ2}$ interrupts use hardware prioritized vectors. The single SCI interrupt and three timer interrupts are serviced in a prioritized order and each is vectored to a separate location. All interrupt vector locations are shown in Table 5. In mode 0, reset and interrupt vectors are defined as $BFF0-$BFFF.

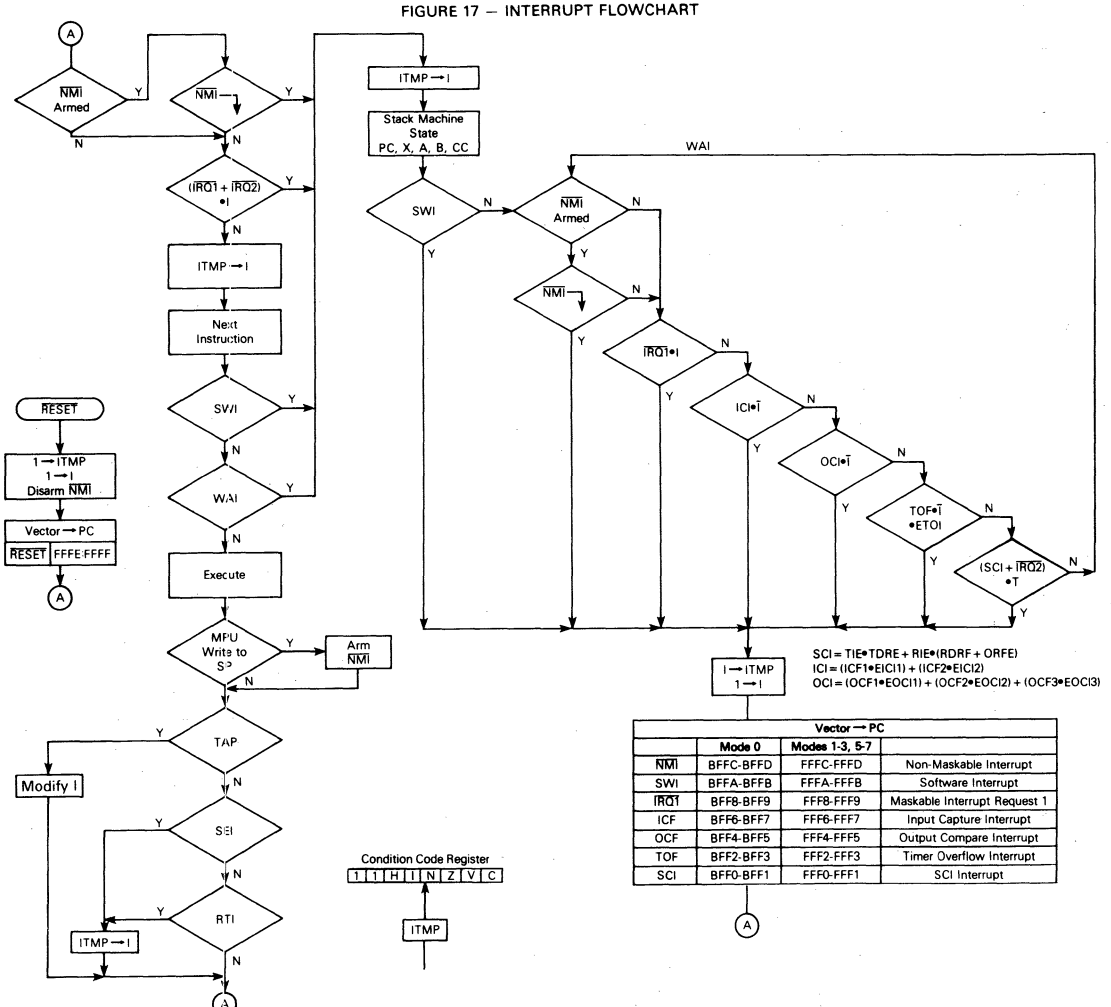
The interrupt flowchart is depicted in Figure 17 and is common to every interrupt excluding reset. During interrupt servicing, the program counter, index register, A accumulator, B accumulator, and condition code register are pushed to the stack. The I bit is set to inhibit maskable interrupts and a vector is fetched corresponding to the current highest priority interrupt. The vector is transferred to the program counter and instruction execution is resumed. Interrupt and $\overline{RESET}$ timing are illustrated in Figures 18 and 19.

TABLE 5 — MCU INTERRUPT VECTOR LOCATIONS

Mode 0		Modes 1-3, 5-7		
MSB	LSB	MSB	LSB	Interrupt
BFFE	BFFF	FFFE	FFFF	RESET
BFFC	BFFD	FFFC	FFFD	Non-Maskable Interrupt**
BFFA	BFFB	FFFA	FFFB	Software Interrupt
BFF8	BFF9	FFF8	FFF9	Maskable Interrupt Request 1
BFF6	BFF7	FFF6	FFF7	Input Capture Flag*
BFF4	BFF5	FFF4	FFF5	Output Compare Flag*
BFF2	BFF3	FFF2	FFF3	Timer Overflow Flag*
BFF0	BFF1	FFF0	FFF1	Serial Communications Interface*

*$\overline{IRQ2}$ interrupt
**$\overline{NMI}$ must be armed (by accessing stack pointer) before an $\overline{NMI}$ is executed

FIGURE 17 – INTERRUPT FLOWCHART



Vector → PC			
	Mode 0	Modes 1-3, 5-7	
NMI	BFFC-BFFD	FFFC-FFFD	Non-Maskable Interrupt
SWI	BFFA-BFFB	FFFA-FFFB	Software Interrupt
IRQ1	BFF8-BFF9	FFF8-FFF9	Maskable Interrupt Request 1
ICF	BFF6-BFF7	FFF6-FFF7	Input Capture Interrupt
OCF	BFF4-BFF5	FFF4-FFF5	Output Compare Interrupt
TOF	BFF2-BFF3	FFF2-FFF3	Timer Overflow Interrupt
SCI	BFF0-BFF1	FFF0-FFF1	SCI Interrupt

FIGURE 18 — INTERRUPT SEQUENCE

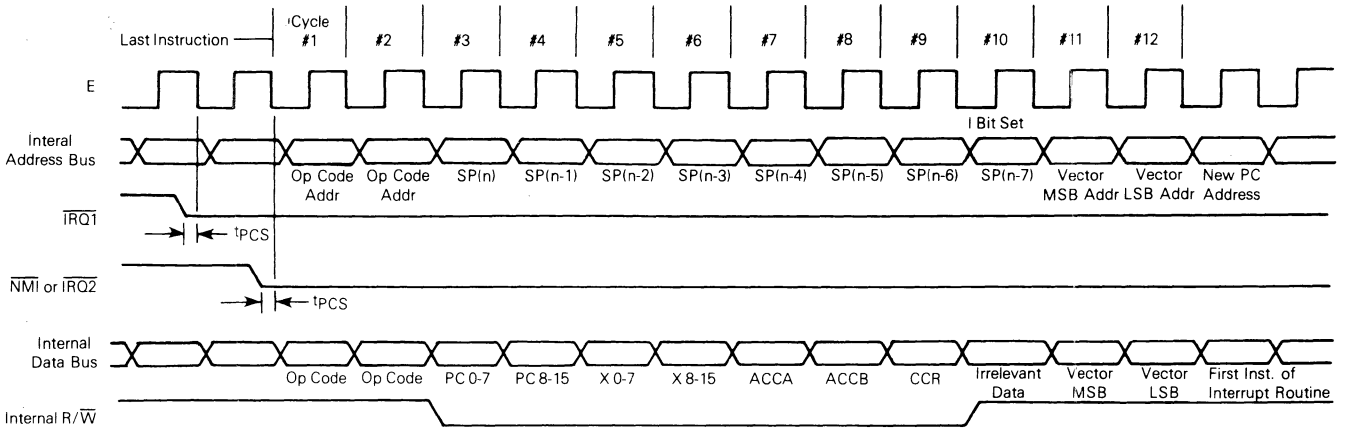
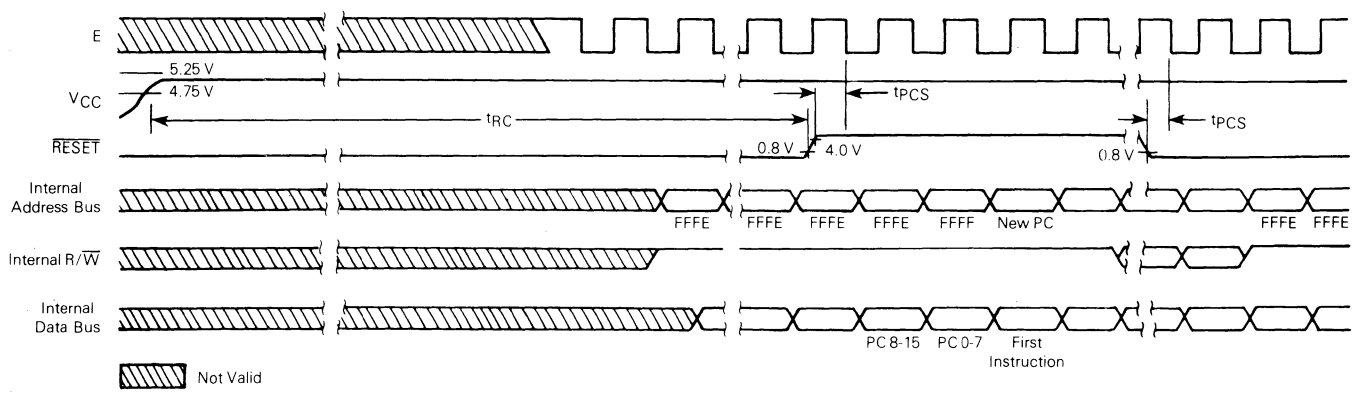


FIGURE 19 — RESET TIMING



FUNCTIONAL PIN DESCRIPTIONS

V_{CC} AND V_{SS}

V_{CC} and V_{SS} provide power to a large portion of the MCU. The power supply should provide +5 volts (±5%) to V_{CC} and V_{SS} should be tied to ground. Total power dissipation (including V_{CC} standby) will not exceed P_D milliwatts.

V_{CC} STANDBY

V_{CC} standby provides power to the standby portion ($40 through $5F) of the RAM and the STBY PWR and RAME bits of the RAM control register. Voltage requirements depend on whether the device is in a power-up or power-down state. In the power-up state, the power supply should provide +5 volts (±5%) and must reach V_{SB} volts before $\overline{RESET}$ reaches 4.0 volts. During power down, V_{CC} standby must remain above V_{SBB} (minimum) to sustain the standby RAM and STBY PWR bit. While in power-down operation, the standby current will not exceed I_{SBB}.

It is typical to power both V_{CC} and V_{CC} standby from the same source during normal operation. A diode must be used between them to prevent supplying power to V_{CC} during power-down operation.

3

XTAL AND EXTAL

These two input pins interface either a crystal or TTL-compatible clock to the MCU internal clock generator. Divide-by-four circuitry is included which allows use of the inexpensive 3.58 MHz or 4.4336 MHz color burst TV crystals. A 20 pF capacitor should be tied from each crystal pin to ground to ensure reliable startup and operation. Alternatively, EXTAL may be driven by an external TTL-compatible clock at 4 f_o with a duty cycle of 50% (±5%) with XTAL connected ground.

The internal oscillator is designed to interface with an AT-cut quartz crystal resonator operated in parallel resonance mode in the frequency range specified for f_{XTAL}. The crystal should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time. The MCU is compatible with most commercially available crystals. Nominal crystal parameters are shown in Figure 20.

$\overline{RESET}$/V_{PP}

This input is used to reset the internal state of the device and provide an orderly startup procedure. During power up, $\overline{RESET}$ must be held below 0.8 volts: (1) at least t_{RC} after V_{CC} reaches 4.75 volts in order to provide sufficient time for the clock generator to stabilize, and (2) until V_{CC} standby reaches 4.75 volts. $\overline{RESET}$ must be held low at least three E cycles if asserted during power-up operation.

This pin is also used to supply V_{PP} in mode 0 for programming the EPROM.

E (ENABLE)

This is an output clock used primarily for bus synchronization. It is TTL compatible and is the slightly skewed divide-by-four result of the device input clock frequency. It will drive one Schottky TTL load and 90 pF, and all data given in cycles is referenced to this clock unless otherwise noted.

$\overline{NMI}$ (NON-MASKABLE INTERRUPT)

An $\overline{NMI}$ negative edge requests an MCU interrupt sequence, but the current instruction will be completed before it responds to the request. The MCU will then begin an interrupt sequence. Finally, a vector is fetched from $FFFC and $FFFD ($BFFC and $BFFD in mode 0), transferred to the program counter, and instruction execution is resumed. $\overline{NMI}$ typically requires a 3.3 kΩ (nominal) resistor to V_{CC}. There is no internal $\overline{NMI}$ pullup resistor. $\overline{NMI}$ must be held low for at least one E cycle to be recognized under all conditions.

NOTE

After reset, an $\overline{NMI}$ will not be serviced until the first program load of the stack pointer. Any $\overline{NMI}$ generated before this load will remain pending by the processor.

$\overline{IRQ1}$ (MASKABLE INTERRUPT REQUEST 1)

$\overline{IRQ1}$ is a level-sensitive input which can be used to request an interrupt sequence. The MPU will complete the current instruction before it responds to the request. If the interrupt mask bit (I bit) in the condition code register is clear, the MCU will begin an interrupt sequence. A vector is fetched from $FFF8 and $FFF9 ($BFF8 and $BFF9 in mode 0), transferred to the program counter, and instruction execution is resumed.

$\overline{IRQ1}$ typically requires an external 3.3 kΩ (nominal) resistor to V_{CC} for wire-OR application. $\overline{IRQ1}$ has no internal pullup resistor.

SC1 and SC2 (STROBE CONTROL 1 AND 2)

The function of SC1 and SC2 depends on the operating mode. SC1 is configured as an output in all modes except single-chip mode, whereas SC2 is always an output. SC1 and SC2 can drive one Schottky load and 90 pF.

SC1 AND SC2 IN SINGLE-CHIP MODE – In single-chip mode, SC1 and SC2 are configured as an input and output, respectively, and both function as port 3 control lines. SC1 functions as IS3 and can be used to indicate that port 3 input data is ready or output data has been accepted. Three options associated with IS3 are controlled by the port 3 control and status register and are discussed in the port 3 description; refer to **P30-P37 (PORT 3)**. If unused, IS3 can remain unconnected.

SC2 is configured as OS3 and can be used to strobe output data or acknowledge input data. It is controlled by output strobe select (OSS) in the port 3 control and status register. The strobe is generated by a read (OSS = 0) or write (OSS = 1) to the port 3 data register. OS3 timing is shown in Figure 3.

SC1 AND SC2 IN EXPANDED NON-MULTIPLEXED MODE – In the expanded non-multiplexed mode, both SC1 and SC2 are configured as outputs. SC1 functions as input/output select (IOS) and is asserted only when $0100 through $01FF is sensed on the internal address bus.

SC2 is configured as read/write and is used to control the direction of data bus transfers. An MPU read is enabled when read/write and E are high.

SC1 AND SC2 IN EXPANDED MULTIPLEXED MODE – In the expanded multiplexed modes, both SC1 and SC2 are configured as outputs. SC1 functions as address strobe and can be used to demultiplex the eight least significant addresses and the data bus. A latch controlled by address strobe captures the lower address on the negative edge, as shown in Figure 13.

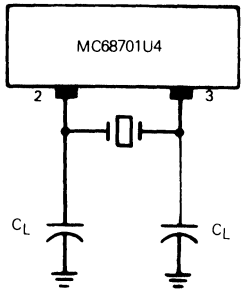
FIGURE 20 — OSCILLATOR CHARACTERISTICS

(a) Nominal Recommended Crystal Parameters

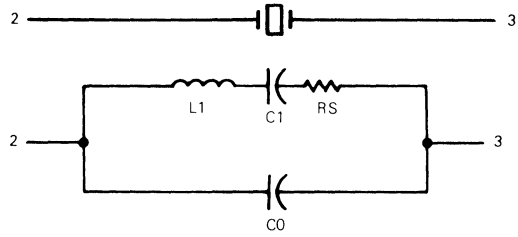
Nominal Crystal Parameters*

	3.58 MHz	4.00 MHz	5.0 MHz
RS	60 Ω	50 Ω	30-50 Ω
C0	3.5 pF	6.5 pF	4-6 pF
C1	0.015 pF	0.025 pF	0.01-0.02 pF
Q	>40 K	>30 K	>20 K

*NOTE: These are representative AT-cut crystal parameters only. Crystals of other types of cut may also be used.



C_L = 20 pF (typical)

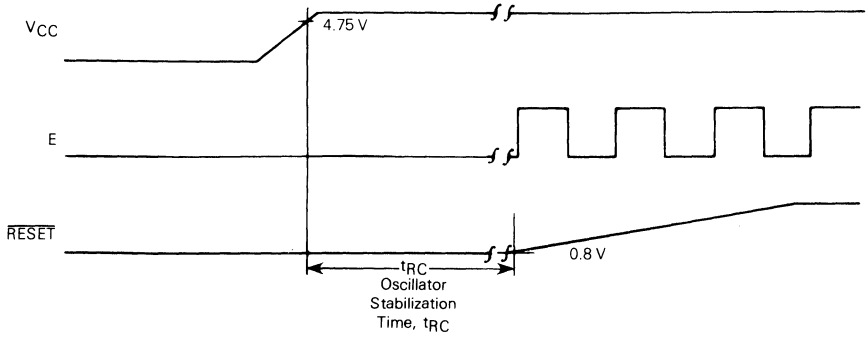


Equivalent Circuit

NOTE
TTL-compatible oscillators may be obtained from:

Motorola Component Products
Attn: Crystal Clock Oscillators
2553 N. Edgington St.
Franklin Park, IL 60131
Tel: 312-451-1000
Telex: 433-0067

(b) Oscillator Stabilization Time (t_{RC})



SC2 is configured as read/write and is used to control the direction of data bus transfers. An MPU read is enabled when read/write and E are high.

P10-P17 (PORT 1)

Port 1 is a mode independent 8-bit I/O and timer port. Each line can be configured as either an input or output as defined by the port 1 data direction register. Port 1 bits 0, 1, and 2 (P10, P11, and P12) can also be used to exercise one input edge function and two output compare functions of the timer. The TTL compatible three-state buffers can drive one Schottky TTL load and 30 pF, Darlington transistors, or CMOS devices using external pullup resistors. It is configured as a data input port during $\overline{\text{RESET}}$. Unused pins can remain unconnected.

P20-P24 (PORT 2)

Port 2 is a mode-independent, 5-bit, multipurpose I/O port. The voltage levels present on P20, P21, and P22 on the rising edge of $\overline{\text{RESET}}$ determine the operating mode of the MCU. The entire port is then configured as a data input port. The port 2 lines can be selectively configured as data output lines by setting the appropriate bits in the port 2 data direction register. The port 2 data register is used to move data through the port. However, if P21 is configured as an output, it is tied to the timer output compare 1 function and cannot be used to provide output from the port 2 data register unless output enable 1 (OE1) is cleared in timer control register 1.

Port 2 can also be used to provide an interface for the serial communications interface and the timer input edge function. These configurations are described in **SERIAL COMMUNICATIONS INTERFACE** and **PROGRAMMABLE TIMER**.

The port 2 three-state TTL-compatible output buffers are capable of driving one Schottky TTL load and 30 pF, or CMOS devices using external pullup resistors.

PORT 2 DATA REGISTER

7	6	5	4	3	2	1	0	
PC2	PC1	PC0	P24	P23	P22	P21	P20	$03

P30-P37 (PORT 3)

Port 3 can be configured as an I/O port, a bidirectional 8-bit data bus, or a multiplexed address/data bus depending on the operating mode. The TTL compatible three-state output buffers can drive one Schottky TTL load and 90 pF. Unused lines can remain unconnected.

PORT 3 IN SINGLE-CHIP MODE – Port 3 is an 8-bit I/O port in the single-chip mode with each line configured by the port 3 data direction register. There are also two lines, IS3 and OS3, which can be used to control port 3 data transfers.

Three port 3 options are controlled by the port 3 control and status register and are available only in single-chip mode: 1) port 3 input data can be latched using IS3 (SC1) as a control signal, 2) OS3 (SC2) can be generated by either an MPU read or write to the port 3 data register, and 3) an $\overline{\text{IRQ1}}$ interrupt can be enabled by an IS3 negative edge. Port 3 latch timing is shown in Figure 4.

PORT 3 CONTROL AND STATUS REGISTER

7	6	5	4	3	2	1	0	
IS3 Flag	IS3 $\overline{\text{IRQ1}}$	X	OSS	Latch Enable	X	X	X	$0F

Bits 0-2 Not used.

Bit 3 **Latch Enable** – This bit controls the input latch for port 3. If set, input data is latched by an IS3 negative edge. The latch is transparent after a read of the port 3 data register. Latch enable is cleared during reset.

Bit 4 **OSS (Output Strobe Select)** – This bit determines whether OS3 will be generated by a read or write of the port 3 data register. When clear, the strobe is generated by a read; when set, it is generated by a write. OSS is cleared during reset.

Bit 5 Not used.

Bit 6 **IS3 $\overline{\text{IRQ1}}$ Enable** – When set, an $\overline{\text{IRQ1}}$ interrupt will be enabled whenever the IS3 flag is set; when clear, the interrupt is inhibited. This bit is cleared during reset.

Bit 7 **IS3 Flag** – This read-only status bit is set by an IS3 negative edge. It is cleared by a read of the port 3 control and status register (with IS3 flag set) followed by a read or write to the port 3 data register or during reset.

PORT 3 IN EXPANDED NON-MULTIPLEXED MODE – Port 3 is configured as a bidirectional data bus (D7-D0) in the expanded non-multiplexed mode. The direction of data transfers is controlled by read/write (SC2). Data is clocked by E (enable).

PORT 3 IN EXPANDED MULTIPLEXED MODE – Port 3 is configured as a time multiplexed address (A7-A0) and data bus (D7-D0) in the expanded multiplexed mode where address strobe (AS) can be used to demultiplex the two buses. Port 3 is held in a high-impedance state between valid address and data to prevent bus conflicts.

P40-P47 (PORT 4)

Port 4 is configured as an 8-bit I/O port, as address outputs, or as data inputs depending on the operating mode. Port 4 can drive one Schottky TTL load and 90 pF, and is the only port with internal pullup resistors. Unused lines can remain unconnected.

PORT 4 IN SINGLE-CHIP MODE – In single-chip mode, port 4 functions as an 8-bit I/O port with each line configured by the port 4 data direction register. Internal pullup resistors allow the port to directly interface with CMOS at 5-volt levels. External pullup resistors to more than 5 volts, however, cannot be used.

PORT 4 IN EXPANDED NON-MULTIPLEXED MODE – Port 4 is configured from reset as an 8-bit input port where the port 4 data direction register can be written to provide any or all of eight address lines A0 to A7. Internal pullup resistors pull the lines high until the port 4 data direction register is configured.

PORT 4 IN EXPANDED MULTIPLEXED MODE — In all expanded multiplexed modes except modes 1 and 6, port 4 functions as half of the address bus and provides A8 to A15. In modes 1 and 6, the port is configured from reset as an 8-bit parallel input port where the port 4 data direction register can be written to provide any or all of upper address lines A8 to A15. Internal pullup resistors pull the lines high until the port 4 data direction register is configured where bit 0 controls A8.

RESIDENT MEMORY

The MC68701U4 has 192 bytes of on-chip RAM and 4096 bytes of on-chip UV erasable EPROM. This memory is controlled by four bits in the RAM/EPROM control register.

Thirty-two bytes of the RAM are powered through the V_{CC} standby pin and are maintainable during V_{CC} power-down. This standby portion of the RAM consists of 32 bytes located from $40 through $5F.

Power must be supplied to V_{CC} standby if the internal RAM is to be used, regardless of whether standby power operation is anticipated.

The RAM is controlled by the RAM/EPROM control register.

RAM/EPROM CONTROL REGISTER ($14)

The RAM/EPROM control register includes four bits: STBY PWR, RAME, PLC, and PPC. Two of these bits, STBY PWR and RAME, are used to control RAM access and determine the adequacy of the standby power source during power-down operation. It is intended that RAME be cleared and STBY PWR be set as part of a power-down procedure. RAME and STBY PWR are read/write bits.

The remaining two bits, PLC and PPC, control the operation of the EPROM. PLC and PPC are readable in all modes but can be changed only in mode 0. The PLC bit can be written without restriction in mode 0, but operation of the PPC bit is controlled by the state of PLC.

Associated with the EPROM are an 8-bit data latch and a 16-bit address latch. The data latch is enabled at all times, latching each data byte written to the EPROM. The address latch is controlled by the PLC bit.

A description of the RAM/EPROM control register follows.

RAM/EPROM CONTROL REGISTER

7	6	5	4	3	2	1	0	
STBY PWR	RAME	X	X	X	X	PPC	PLC	$14

Bit 0 **Programming Latch Control (PLC).** This bit controls the latch which captures the EPROM address to be programmed and whether the PCC bit can be cleared. The latch is triggered by an MPU write to a location in the EPROM. This bit is set during reset and can be cleared only in mode 0. The PLC bit is defined as follows:

PLC = 0 — EPROM address latch enabled; EPROM address is latched during MPU writes to the EPROM.

PLC = 1 — EPROM address latch is transparent.

Bit 1 **Programming Power Control (PPC).** This bit gates power from the $\overline{RESET}/V_{PP}$ pin to the EPROM programming circuit. PPC is set during reset and whenever the PLC bit is set. It can be cleared only if operating in mode 0, and if PLC has been previously cleared. The PPC bit is defined as follows:

PPC = 0 — EPROM programming power (V_{PP}) applied.

PPC = 1 — EPROM programming power (V_{PP}) is not applied.

Bit 2-5 Unused.

Bit 6 **RAM Enable (RAME).** This read/write bit can be used to remove the entire RAM from the internal memory map. RAME is set (enabled) during reset provided standby power is available on the positive edge of reset. If RAME is clear, any access to a RAM address is external. If RAME is set, the RAM is included in the internal map.

Bit 7 **Standby Power (STBY PWR).** This bit is a read/write status bit which when cleared indicates that V_{CC} standby has decreased sufficiently below V_{SBB} (minimum) to make data in the standby RAM suspect. It can be set only by software and is not affected during reset.

Note that if PPC and PLC are set, they cannot be simultaneously cleared with a single MPU write. The PLC bit must be cleared prior to attempting to clear PPC. If both PPC and PLC are clear, setting PLC will also set PPC. In addition, it is assumed that V_{PP} is applied to the $\overline{RESET}/V_{PP}$ pin whenever PCC is clear. If this is not the case, the result is undefined.

ERASING THE MC68701U4 EPROM

Ultraviolet erasure will clear all bits of the EPROM to the zero state. The MC68701U4 EPROM is programmed by erasing it to zeros and entering ones into the desired bit locations.

The MC68701U4 EPROM can be erased by exposure to high intensity ultraviolet light with a wave length of 2537 angstroms for a minimum of 30 minutes. The recommended integrated dose (ultraviolet intensity times exposure time) is 15 watts/centimeter. The lamps should be used without shortwave filters, the MC68701U4 should be positioned about one inch away from the ultraviolet tubes, and the transparent lid should not be covered.

The MC68701U4 transparent lid should always be covered after erasing. This protects both the EPROM and light-sensitive nodes from accidental exposure to ultraviolet light.

PROGRAMMING THE MC68701U4 EPROM

When the MC68701U4 is released from reset in mode 0, a vector is fetched from location $BFFE:$BFFF. This provides a method for an external program to obtain control of the microcomputer with access to every location in the EPROM.

To program the EPROM, it is necessary to operate the MC68701U4 in mode 0 under the control of a program resident in external memory which can facilitate loading and programming of the EPROM. After the pattern has been loaded

into external memory, the EPROM can be programmed as follows:

a. Apply programming power (V_{PP}) to the $\overline{RESET}/V_{PP}$ pin.
b. Clear the PLC control bit and set the PPC bit by writing $FE to the RAM/EPROM control register.
c. Write data to the next EPROM location to be programmed. Triggered by an MPU write to the EPROM, internal latches capture both the EPROM address and the data byte.
d. Clear the PPC bit for programming time, t_{pp}, by writing $FC to the RAM/EPROM control register and waiting for time, t_{pp}. This step gates the programming power (V_{PP}) from the $\overline{RESET}/V_{PP}$ pin to the EPROM which programs the location.
e. Repeat steps b through d for each byte to be programmed.
f. Set the PLC and PPC bits by writing $FF to the RAM/EPROM control register.
g. Remove the programming power (V_{PP}) from the $\overline{RESET}/V_{PP}$ pin. The EPROM can now be read and verified.

Because the erased state of an EPROM byte is $00, it is not necessary to program a location which is to contain $00. Finally, it should be noted that the result of inadvertently programming a location more than once is the logical OR of the data patterns.

PROGRAMMABLE TIMER

The programmable timer can be used to perform measurements on two separate input waveforms while independently generating three output waveforms. Pulse widths can vary from several microseconds to many seconds. A block diagram of the timer is shown in Figure 21.

COUNTER ($09:0A), ($15, $16)

The key timer element is a 16-bit free-running counter which is incremented by E (enable). It is cleared during reset and is read-only with one exception: in mode 0 a write to the counter ($09) will preset it to $FFF8. This feature, intended for testing, can disturb serial operations because the counter provides the SCI internal bit rate clock. The TOF bit is set whenever the counter contains all ones. If ETOI is set, an interrupt will occur when the TOF is set. The counter may also be read at $15 and $16 to avoid inadvertently clearing the TOF.

OUTPUT COMPARE REGISTERS ($0B:0C), ($1A:1B), ($1C:1D)

The three output compare registers are 16-bit read/write registers, each used to control an output waveform or provide an arbitrary time-out flag. They are compared with the free-running counter during the negative half of each E cycle. When a match occurs, the corresponding output compare flag (OCF) is set and the corresponding output level (OLVL) is clocked to an output level register. If both the corresponding output enable bit and data direction register bit are set, the value represented in the output level register will appear on the corresponding port pin. The appropriate OLVL bit can then be changed for the next compare.

The function is inhibited for one cycle after a write to its high byte ($0B, $1A, or $1C) to ensure a valid compare after a double byte write. Writes can be made to either byte of the output compare register without affecting the other byte. The OLVL value will be clocked out independently of whether the OCF had previously been cleared. The output compare registers are set to $FFFF during reset.

INPUT CAPTURE REGISTERS ($0D:0E), ($1E:1F)

The two input capture registers are 16-bit read-only registers used to store the free-running counter when a "proper" input transition occurs as defined by the corresponding input edge bit (IEDG1 or IEDG2). The input pin's data direction register should be configured as an input, but the edge detect circuit always senses P10 and P20 even when configured as an output. The counter value will be latched into the input capture registers on the second negative edge of the E clock following the transition.

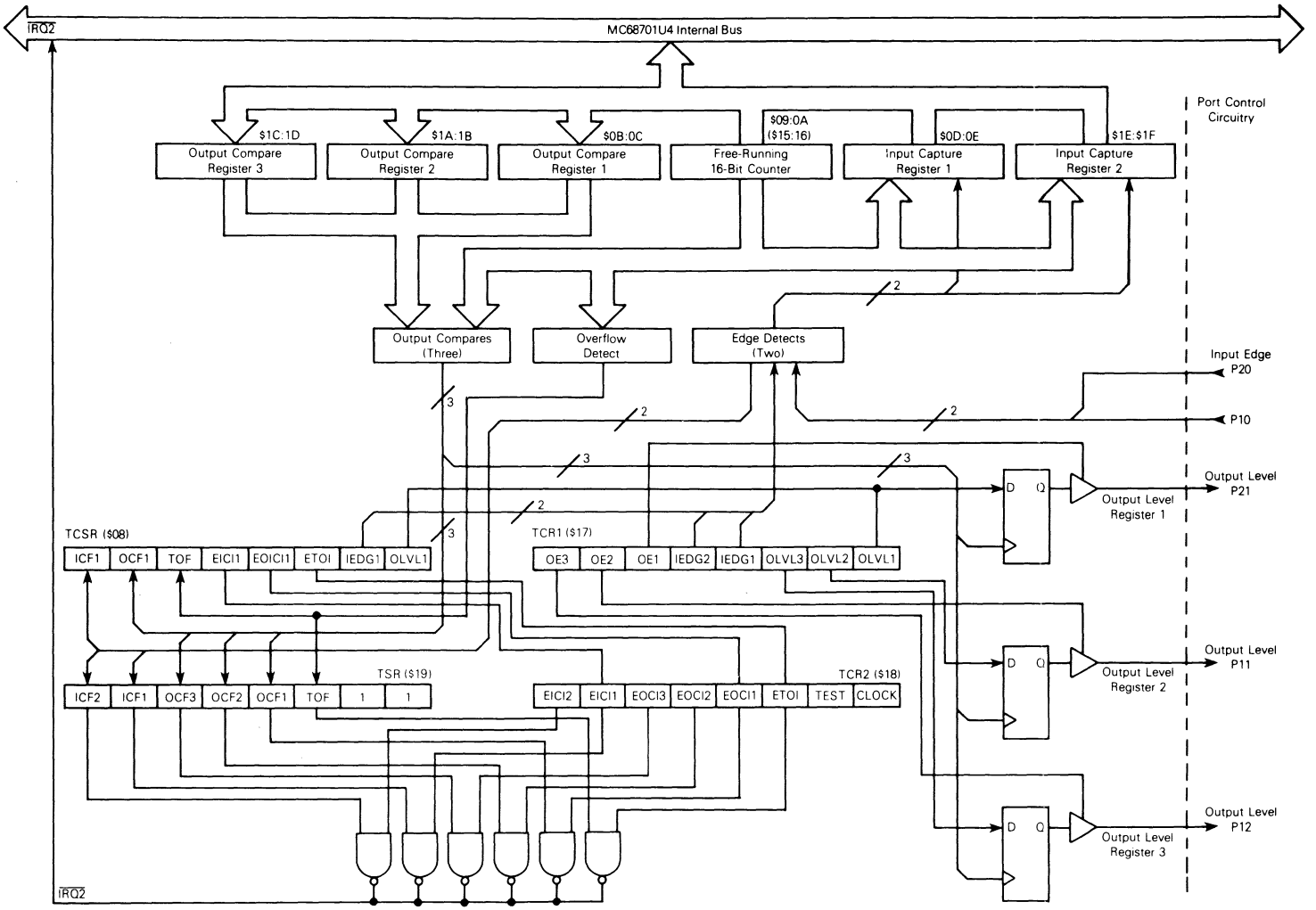
An input capture can occur independently of ICF; the register always contains the most current value. Counter transfer is inhibited, however, between accesses of a double byte MPU read. The input pulse width must be at least two E cycles to ensure an input capture under all conditions.

TIMER CONTROL AND STATUS REGISTERS

Four registers are used to provide the MC68701U4 with control and status information about the three output compare functions, the timer overflow function, and the two input edge functions of the timer. They are:

Timer Control and Status Register (TCSR)
Timer Control Register 1 (TCR1)
Timer Control Register 2 (TCR2)
Timer Status Register (TSR)

FIGURE 21 — BLOCK DIAGRAM OF PROGRAMMABLE TIMER



TIMER CONTROL AND STATUS REGISTER (TCSR) ($08) — The timer control and status register is an 8-bit register of which all bits are readable, while only bits 0-4 can be written. All the bits in this register are also accessible through the two timer control registers and the timer status register. The three most significant bits provide the timer status and indicate if:

1. a proper level transition has been detected at P20,
2. a match has occurred between the free-running counter and output compare register 1, or
3. the free-running counter has overflowed.

Each of the three events can generate an $\overline{IRQ2}$ interrupt and is controlled by an individual enable bit in the TCSR.

TIMER CONTROL AND STATUS REGISTER

7	6	5	4	3	2	1	0	
ICF1	OCF1	TOF	EICI1	EOCI1	ETOI	IEDG1	OLVL1	$08

Bit 0 **Output Level 1** — OLVL1 is clocked to output level register 1 by a successful output compare and will appear at P21 if bit 1 of the port 2 data direction register is set and the OE1 control bit in timer control register 1 is set. OLVL1 and output level register 1 are cleared during reset. Refer to **TIMER CONTROL REGISTER 1 (TCR1) ($17)**.

Bit 1 **Input Edge 1** — IEDG1 is cleared during reset and controls which level transition on P20 will trigger a counter transfer to input capture register 1:
IEDG1 = 0 transfer on a negative-edge
IEDG1 = 1 transfer on a positive-edge
Refer to **TIMER CONTROL REGISTER 1 (TCR1) ($17)**.

Bit 2 **Enable Timer Overflow Interrupt** — When set, an $\overline{IRQ2}$ interrupt will be generated when the timer overflow flag is set; when clear, the interrupt is inhibited. ETOI is cleared during reset. Refer to **TIMER CONTROL REGISTER 2 (TCR2) ($18)**.

Bit 3 **Enable Output Compare Interrupt 1** — When set, an $\overline{IRQ2}$ interrupt will be generated when output compare flag 1 is set; when clear, the interrupt is inhibited. EOCI1 is cleared during reset. Refer to **TIMER CONTROL REGISTER 2 (TCR2) ($18)**.

Bit 4 **Enable Input Capture Interrupt 1** — When set, an $\overline{IRQ2}$ interrupt will be generated when input capture flag 1 is set; when clear, the interrupt is inhibited. EICI1 is cleared during reset. Refer to **TIMER CONTROL REGISTER 2 (TCR2) ($18).**

Bit 5 **Timer Overflow Flag** — The TOF is set when the counter contains all ones ($FFFF). It is cleared by reading the TCSR or the TSR (with TOF set) and the counter high byte ($09), or during reset. Refer to **TIMER STATUS REGISTER (TSR) ($19)**.

Bit 6 **Output Compare Flag 1** — OCF1 is set when output compare register 1 matches the free-running counter. OCF1 is cleared by reading the TCSR or the TSR (with OCF1 set) and then writing to output compare register 1 ($0B or $0C), or during reset. Refer to **TIMER STATUS REGISTER (TSR) ($19)**.

Bit 7 **Input Capture Flag** — ICF1 is set to indicate that a proper level transition has occurred; it is cleared by reading the TCSR or the TSR (with ICF1 set) and the input capture register 1 high byte ($0D), or during reset. Refer to **TIMER STATUS REGISTER (TSR) ($19)**.

TIMER CONTROL REGISTER 1 (TCR1) ($17) — Timer control register 1 is an 8-bit read/write register which contains the control bits for interfacing the output compare and input capture registers to the corresponding I/O pins.

TIMER CONTROL REGISTER 1

7	6	5	4	3	2	1	0	
OE3	OE2	OE1	IEDG2	IEDG1	OLVL3	OLVL2	OLVL1	$17

Bit 0 **Output Level 1** — OLVL1 is clocked to output level register 1 by a successful output compare and will appear at P21 if bit 1 of the port 2 data direction register is set and the OE1 control bit is set. OLVL1 and output level register 1 are cleared during reset. Refer to **TIMER CONTROL AND STATUS REGISTER (TCSR) ($08)**.

Bit 1 **Output Level 2** — OLVL2 is clocked to output level register 2 by a successful output compare and will appear at P11 if bit 1 of port 1 data direction register is set and the OE2 control bit is set. OLVL2 and output level register 2 are cleared during reset.

Bit 2 **Output Level 3** — OLVL3 is clocked to output level register 3 by a successful output compare and will appear at P12 if bit 2 of port 1 data direction register is set and the OE3 control bit is set. OLVL3 and output level register 3 are cleared during reset.

Bit 3 **Input Edge 1** — IEDG1 is cleared during reset and controls which level transition on P20 will trigger a counter transfer to input capture register 1.
IEDG1 = 0 transfer on a negative-edge
IEDG1 = 1 transfer on a positive-edge
Refer to **TIMER CONTROL AND STATUS REGISTER (TCSR) ($08)**.

Bit 4 **Input Edge 2** — IEDG2 is cleared during reset and controls which level transition on P10 will trigger a counter transfer to input capture register 2.
IEDG2 = 0 transfer on a negative-edge
IEDG2 = 1 transfer on a positive-edge

Bit 5 **Output Enable 1** — OE1 is set during reset and enables the contents of output level register 1 to be connected to P21 when bit 1 of port 2 data direction register is set.
OE1 = 0 port 2 bit 1 data register output
OE1 = 1 output level register 1

Bit 6 **Output Enable 2** — OE2 is cleared during reset and enables the contents of output level register 2 to be connected to P11 when bit 1 of port 1 data direction register is set.
OE2 = 0 port 1 bit 1 data register output
OE2 = 1 output level register 2

Bit 7 **Output Enable 3** – OE3 is cleared during reset and enables the contents of output level register 3 to be connected to P12 when bit 2 of port 1 data direction register is set
OE3 = 0 port 1 bit 2 data register output
OE3 = 1 output level register 3

TIMER CONTROL REGISTER 2 ($18) – Timer control register 2 is an 8-bit read/write register (except bits 0 and 1) which enable the interrupts associated with the free-running counter, the output compare registers, and the input capture registers. In test mode 0, two more bits (clock and test) are available for checking the timer.

TIMER CONTROL REGISTER 2
(Non-Test Modes)

7	6	5	4	3	2	1	0	
EICI2	EICI1	EOCI3	EOCI2	EOCI1	ETOI	1	1	$18

Bits 0-1 **Read-Only Bits** – When read, these bits return a value of 1. Refer to **TIMER CONTROL REGISTER 2 (Test Mode)**.

Bit 2 **Enable Timer Overflow Interrupt** – When set, an $\overline{IRQ2}$ interrupt will be generated when the timer overflow flag is set; when clear, the interrupt is inhibited. ETOI is cleared during reset. Refer to **TIMER CONTROL AND STATUS REGISTER (TCSR) ($08)**.

Bit 3 **Enable Output Compare Interrupt 1** – When set, an $\overline{IRQ2}$ interrupt will be generated when the output compare flag 1 is set; when clear, the interrupt is inhibited. EOCI1 is cleared during reset. Refer to **TIMER CONTROL AND STATUS REGISTER (TCSR) ($08)**.

Bit 4 **Enable Output Compare Interrupt 2** – When set, an $\overline{IRQ2}$ interrupt will be generated when the output compare flag 2 is set; when clear, the interrupt is inhibited. EOCI2 is cleared during reset.

Bit 5 **Enable Output Compare Interrupt 3** – When set, an $\overline{IRQ2}$ interrupt will be generated when the output compare flag 3 is set; when clear, the interrupt is inhibited. EOCI3 is cleared during reset.

Bit 6 **Enable Input Capture Interrupt 1** – When set, an $\overline{IRQ2}$ interrupt will be generated when the input capture flag 1 is set; when clear, the interrupt is inhibited. EICI1 is cleared during reset. Refer to **TIMER CONTROL AND STATUS REGISTER (TCSR) ($08)**.

Bit 7 **Enable Input Capture Interrupt 2** – When set, an $\overline{IRQ2}$ interrupt will be generated when the input capture flag 2 is set; when clear, the interrupt is inhibited. EICI2 is cleared during reset.

The timer test bits (test and clock) allow the free-running counter to be tested as two separate 8-bit counters to speed testing.

TIMER CONTROL REGISTER 2
(Test Mode)

7	6	5	4	3	2	1	0	
EICI2	EICI1	EOCI3	EOCI2	EOCI1	ETOI	TEST	CLOCK	$18

Bit 0 **CLOCK** – The CLOCK control bit selects which half of the 16-bit free-running counter (MSB or LSB) should be clocked with E. The CLOCK bit is a read/write bit only in mode 0 and is set during reset.
CLOCK = 0 – Only the eight most significant bits of the free-running counter run with TEST = 0.
CLOCK = 1 – Only the eight least significant bits of the free-running counter run when TEST = 0.

Bit 1 **TEST** – the TEST control bit enables the timer test mode. TEST is a read/write bit in mode 0 and is set during reset.
TEST = 0 – Timer test mode enabled:
a) The timer LSB latch is transparent which allows the LSB to be read independently of the MSB.
b) Either the MSB or the LSB of the timer is clocked by E, as defined by the CLOCK bit.
TEST = 1 – Timer test mode disabled.

Bits 2-7 See **TIMER CONTROL REGISTER 2 (Non-Test Modes)**. (These bits function the same as in the non-test modes.)

TIMER STATUS REGISTER (TSR) ($19) – The timer status register is an 8-bit read-only register which contains the flags associated with the free-running counter, the output compare registers, and the input capture registers.

TIMER STATUS REGISTER

7	6	5	4	3	2	1	0	
ICF2	ICF1	OCF3	OCF2	OCF1	TOF	1	1	$19

Bits 0-1 Not used.

Bit 2 **Timer Overflow Flag** – The TOF is set when the counter contains all ones ($FFFF). It is cleared by reading the TSR or the TCSR (with TOF set) and then the counter high byte ($09), or during reset. Refer to **TIMER CONTROL AND STATUS REGISTER (TCSR) ($08)**.

Bit 3 **Output Compare Flag 1** – OCF1 is set when output compare register 1 matches the free-running counter. OCF1 is cleared by reading the TSR or the TCSR (with OCF1 set) and then writing to output compare register 1 ($0B or $0C), or during reset. Refer to **TIMER CONTROL AND STATUS REGISTER (TCSR) ($08)**.

Bit 4 **Output Compare Flag 2** – OCF2 is set when output compare register 2 matches the free-running counter. OCF2 is cleared by reading the TSR (with OCF2 set) and then writing to output compare register 2 ($1A or $1B), or during reset.

Bit 5 **Output Compare Flag 3** – OCF3 is set when output compare register 3 matches the free-running counter. OCF3 is cleared by reading the TSR (with OCF3 set) and then writing to output compare register 3 ($1C or $1D), or during reset.

Bit 6 **Input Capture Flag 1** – ICF1 is set to indicate that a proper level transition has occurred; it is cleared by reading the TSR or the TCSR (with ICF1 set) and the input capture register 1 high byte ($0D), or during reset. Refer to **TIMER CONTROL AND STATUS REGISTER (TCSR) ($08)**.

3

Bit 7 **Input Capture Flag 2** — ICF2 is set to indicate that a proper level transition has occurred; it is cleared by reading the TSR (with ICF2 set) and the input capture register 2 high byte ($1E), or during reset.

SERIAL COMMUNICATIONS INTERFACE

A full-duplex asynchronous serial communications interface (SCI) is provided with two data formats and a variety of rates. The SCI transmitter and receiver are functionally independent but use the same data format and bit rate. Serial data formats include standard mark/space (NRZ) and bi-phase. Both provide one start bit, eight data bits, and one stop bit. "Baud" and "bit rate" are used synonymously in the following description.

WAKE-UP FEATURE

In a typical serial loop multiprocessor configuration, the software protocol will usually identify the addressee(s) at the beginning of the message. In order to permit uninterested MPUs to ignore the remainder of the message, wake-up feature is included whereby all further SCI receiver flag (and interrupt) processing can be inhibited until its data line goes idle. An SCI receiver is re-enabled by an idle string of eleven consecutive ones or during reset. Software must provide for the required idle string between consecutive messages and prevent it within messages.

PROGRAMMABLE OPTIONS

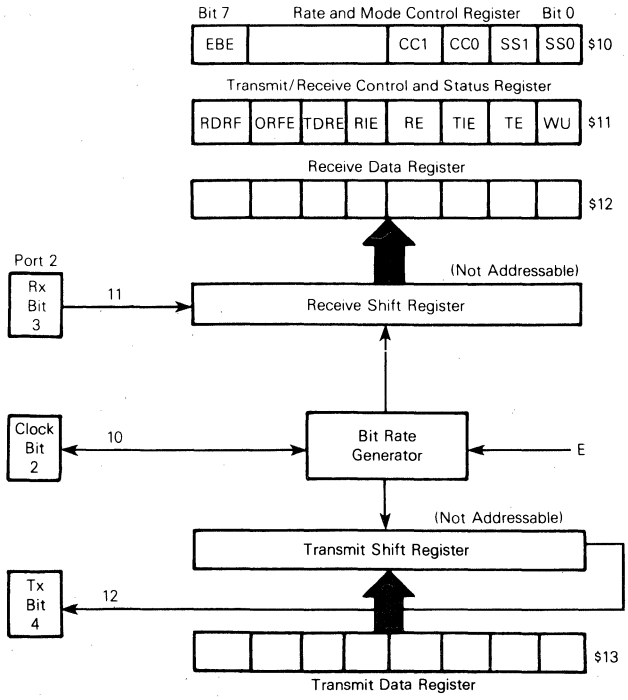
The following features of the SCI are programmable:

- Format: standard mark/space (NRZ) or bi-phase
- Clock: external or internal bit rate clock
- Baud: one of eight per E clock frequency or external clock (×8 desired baud)
- Wake-Up Feature: enabled or disabled
- Interrupt Requests: enabled individually for transmitter and receiver
- Clock Output: internal bit rate clock enabled or disabled to P22

SERIAL COMMUNICATIONS REGISTERS

The serial communications interface includes four addressable registers as depicted in Figure 22. It is controlled by the rate and mode contol register and the transmit/receive control and status register. Data is transmitted and received utilizing a write-only transmit register and a read-only receive register. The shift registers are not accessible to software.

FIGURE 22 — SCI REGISTERS



RATE AND MODE CONTROL REGISTER (RMCR) ($10) – The rate and mode control register controls the SCI bit rate, format, clock source, and under certain conditions, the configuration of P22. The register consists of five write-only bits which are cleared during reset. The two least significant bits in conjunction with bit 7 control the bit rate of the internal clock and the remaining two bits control the format and clock source.

RATE AND MODE CONTROL REGISTER

7	6	5	4	3	2	1	0	
EBE	X	X	X	CC1	CC0	SS1	SS0	$10

Bit 1:Bit 0 **SS1:SS0 Speed Select** – These two bits select the baud when using the internal clock. Eight rates may be selected (in conjunction with bit 7) which are a function of the MCU input frequency. Table 6 lists bit time and rates for three selected MCU frequencies.

Bit 3:Bit 2 **CC1:CC0 Clock Control and Format Select** – These two bits control the format and select the serial clock source. If CC1 is set, the DDR value for P22 is forced to the complement of CC0 and cannot be altered until CC1 is cleared. If CC1 is cleared after having been set, its DDR value is unchanged. Table 7 defines the formats, clock source, and use of P22.

Bits 4-6 Not used.

Bit 7 **EBE Enhanced Baud Enable** – EBE selects the standard MC6801 baud rates when clear and the additional baud rates when set (Table 6). This bit is cleared by reset and is a write-only control bit.

EBE = 0 standard MC6801 baud rates
EBE = 1 additional baud rates

If both CC1 and CC0 are set, an external TTL-compatible clock must be connected to P22 at eight times (8×) the desired bit rate, but not greater than E, with a duty cycle of 50% (±10%). If CC1:CC0 = 10, the internal bit rate clock is provided at P22 regardless of the values for TE or RE.

NOTE

The source of SCI internal bit rate clock is the timer free-running counter. An MPU write to the counter in mode 0 can disturb serial operations.

3

TABLE 6 – SCI BIT TIMES AND RATES

EBE	SS1:SS0		4 f_o →	2.4576 MHz		4.0 MHz		4.9152 MHz	
				614.4 kHz		1.0 MHz		1.2288 MHz	
			E	Baud	Time	Baud	Time	Baud	Time
0	0	0	÷16	38400.0	26 μs	62500.0	16.0 μs	76800.0	13.0 μs
0	0	1	÷128	4800.0	208.3 μs	7812.5	128.0 μs	9600.0	104.2 μs
0	1	0	÷1024	600.0	1.67 ms	976.6	1.024 ms	1200.0	833.3 μs
0	1	1	÷4096	150.0	6.67 ms	244.1	4.096 ms	300.0	3.33 ms
1	0	0	÷64	9600.0	104.2 μs	15625.0	64 μs	19200.0	52.0 μs
1	0	1	÷256	2400.0	416.6 μs	3906.3	256 μs	4800.0	208.3 μs
1	1	0	÷512	1200.0	833.3 μs	1953.1	512 μs	2400.0	416.6 μs
1	1	1	÷2048	300.0	3.33 ms	488.3	2.05 ms	600.0	01.67 ms
External (P22)*				76800.0	13.0 μs	125000.0	8.0 μs	153600.0	6.5 μs

*Using maximum clock rate

TABLE 7 – SCI FORMAT AND CLOCK SOURCE CONTROL

CC1:CC0	Format	Clock Source	Port 2 Bit 2
00	Bi-Phase	Internal	Not Used
01	NRZ	Internal	Not Used
10	NRZ	Internal	Output
11	NRZ	External	Input

TRANSMIT/RECEIVE CONTROL AND STATUS REGISTER (TRCSR) ($11) — The transmit/receive control and status register controls the transmitter, receiver, wake-up feature, and two individual interrupts, and monitors the status of serial operations. All eight bits are readable while bits 0 to 4 are also writable. The register is initialized to $20 by RESET.

TRANSMIT/RECEIVE CONTROL AND STATUS REGISTER

7	6	5	4	3	2	1	0	
RDRF	ORFE	TDRE	RIE	RE	TIE	TE	WU	$11

Bit 0 **"Wake-Up on Idle Line** — When set, WU enables the wake-up function; it is cleared by eleven consecutive ones or during reset. WU will not be set if the line is idle. Refer to **WAKE-UP FEATURE.**

Bit 1 **Transmit Enable** — When set, P24 DDR bit is set, cannot be changed, and will remain set if TE is subsequently cleared. When TE is changed from clear to set, the transmitter is connected to P24 and a preamble of nine consecutive ones is transmitted. TE is cleared during reset.

Bit 2 **Transmit Interrupt Enable** — When set, an $\overline{IRQ2}$ is set; when clear, the interrupt is inhibited. TE is cleared during reset.

Bit 3 **Receive Enable** — When set, the P23 DDR bit is cleared, cannot be changed, and will remain clear if RE is subsequently cleared. While RE is set, the SCI receiver is enabled. RE is cleared during reset.

Bit 4 **Receiver Interrupt Enable** — When set, an $\overline{IRQ2}$ interrupt is enabled when RDRF and/or ORFE is set; when clear, the interrupt is inhibited. RIE is cleared during reset.

Bit 5 **Transmit Data Register Empty** — TDRE is set when the transmit data register is transfered to the output serial shift register or during reset. It is cleared by reading the TRCSR (with TDRE set) and then writing to the transmit data register. Additional data will be transmitted only if TDRE has been cleared.

Bit 6 **Overrun Framing Error** — If set, ORFE indicates either an overrun or framing error. An overrun is a new byte ready to transfer to the receiver data register with RDRF still set. A receiver framing error has occurred when the stop bit (1) is not found in the tenth bit time. An overrun can be distinguished from a framing error by the state of RDRF: if RDRF is set, then an overrun has occurred; otherwise, a framing error has been detected. Data is not transferred to the receive data register in an overrun condition. Unframed data causing a framing error is transferred to the receive data register. However, subsequent data transfer is blocked until the framing error flag is cleared. ORFE is cleared by reading the TRCSR (with ORFE set) then the receive data register, or during reset.

Bit 7 **Receive Data Register Full** — RDRF is set when the input serial shift register is transferred to the receive data register, or during reset.

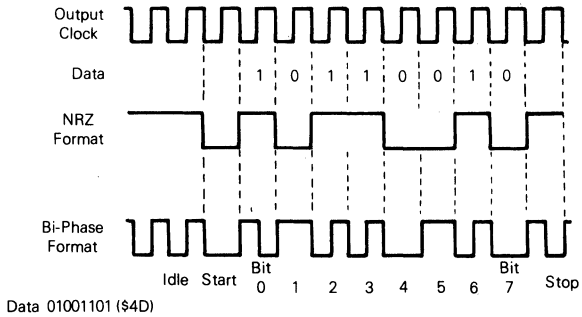
SERIAL OPERATIONS

The SCI is initialized by writing control bytes first to the rate and mode control register and then to the transmit/receive control and status register. When TE is set, the output of the transmit serial shift register is connected to P24 and serial output is initiated by transmitting a 9-bit preamble of ones.

At this point, one of two situations exists: 1) if the transmit data register is empty (TDRE = 1); a continuous string of ones will be sent indicating an idle line; or 2) if a byte has been written to the transmit data register (TDE = 0), it will be transferred to the output serial shift register (synchronized with the bit rate clock), TDRE will be set, and transmission will begin.

The start bit (0), eight data bits (beginning with bit 0), and a stop bit (1) will be transmitted. If TDRE is still set when the next byte transfer occurs, ones will be sent until more data is provided. In bi-phase format, the output toggles at the start of each bit and at half-bit time when a one is sent. Receive operation is controlled by RE which configures P23 as an input and enables the receiver. SCI data formats are illustrated in Figure 23.

FIGURE 23 — SCI DATA FORMATS



Data 01001101 ($4D)

INSTRUCTION SET

The MC68701U4 is directly source compatible with the MC6801 and upward source and object code compatible with the MC6800. Execution times of key instructions have been reduced and several instructions have been added, including a hardware multiply. A list of new operations added to the MC6800 instruction set is shown in Table 1.

In addition, two special opcodes, 4E and 5E, are provided for test purposes. These opcodes force the program counter to increment like a 16-bit counter causing address lines used in the expanded modes to increment until the device is reset. These opcodes have no mnemonics.

The coding of the first (or only) byte corresponding to an executable instruction is sufficient to identify the instruction and the addressing mode. The hexadecimal equivalents of the binary codes, which result from the translation of the 82 instructions in all valid modes of addressing, are shown in Table 8. There are 220 valid machine codes, 34 unassigned codes, and 2 codes reserved for test purposes.

TABLE 8 — CPU INSTRUCTION MAP

OP	MNEM	MODE	~	#	OP	MNEM	MODE	~	#	OP	MNEM	MODE	~	#	OP	MNEM	MODE	~	#	OP	MNEM	MODE	~	#
00	•				34	DES	INHER	3	1	68	ASL	INDXD	6	2	9C	CPX	DIR	5	2	D0	SUBB	DIR	3	2
01	NOP	INHER	2	1	35	TXS		3	1	69	ROL		6	2	9D	JSR		5	2	D1	CMPB		3	2
02	•				36	PSHA		3	1	6A	DEC		6	2	9E	LDS		4	2	D2	SBCB		3	2
03	•				37	PSHB		3	1	6B	•				9F	STS	DIR	4	2	D3	ADDD		5	2
04	LSRD		3	1	38	PULX		5	1	6C	INC		6	2	A0	SUBA	INDXD	4	2	D4	ANDB		3	2
05	ASLD		3	1	39	RTS		5	1	6D	TST		6	2	A1	CMPA		4	2	D5	BITB		3	2
06	TAP		2	1	3A	ABX		3	1	6E	JMP		3	2	A2	SBCA		4	2	D6	LDAB		3	2
07	TPA		2	1	3B	RTI		10	1	6F	CLR	INDXD	6	2	A3	SUBD		6	2	D7	STAB		3	2
08	INX		3	1	3C	PSHX		4	1	70	NEG	EXTND	6	3	A4	ANDA		4	2	D8	EORB		3	2
09	DEX		3	1	3D	MUL		10	1	71	•				A5	BITA		4	2	D9	ADCB		3	2
0A	CLV		2	1	3E	WAI		9	1	72	•				A6	LDAA		4	2	DA	ORAB		3	2
0B	SEV		2	1	3F	SWI		12	1	73	COM		6	3	A7	STAA		4	2	DB	ADDB		3	2
0C	CLC		2	1	40	NEGA		2	1	74	LSR		6	3	A8	EORA		4	2	DC	LDD		4	2
0D	SEC		2	1	41	•				75	•				A9	ADCA		4	2	DD	STD		4	2
0E	CLI		2	1	42	•				76	ROR		6	3	AA	ORAA		4	2	DE	LDX		4	2
0F	SEI		2	1	43	COMA		2	1	77	ASR		6	3	AB	ADDA		4	2	DF	STX	DIR	4	2
10	SBA		2	1	44	LSRA		2	1	78	ASL		6	3	AC	CPX		6	2	E0	SUBB	INDXD	4	2
11	CBA		2	1	45	•				79	ROL		6	3	AD	JSR		6	2	E1	CMPB		4	2
12	•				46	RORA		2	1	7A	DEC		6	3	AE	LDS		5	2	E2	SBCB		4	2
13	•				47	ASRA		2	1	7B	•				AF	STS	INDXD	5	2	E3	ADDD		6	2
14	•				48	ASLA		2	1	7C	INC		6	3	B0	SUBA	EXTND	4	3	E4	ANDB		4	2
15	•				49	ROLA		2	1	7D	TST		6	3	B1	CMPA		4	3	E5	BITB		4	2
16	TAB		2	1	4A	DECA		2	1	7E	JMP		3	3	B2	SBCA		4	3	E6	LDAB		4	2
17	TBA		2	1	4B	•				7F	CLR	EXTND	6	3	B3	SUBD		6	3	E7	STAB		4	2
18	•				4C	INCA		2	1	80	SUBA	IMMED	2	2	B4	ANDA		4	3	E8	EORB		4	2
19	DAA	INHER	2	1	4D	TSTA		2	1	81	CMPA		2	2	B5	BITA		4	3	E9	ADCB		4	2
1A	•				4E	T				82	SBCA		2	2	B6	LDAA		4	3	EA	ORAB		4	2
1B	ABA	INHER	2	1	4F	CLRA		2	1	83	SUBD		4	3	B7	STAA		4	3	EB	ADDB		4	2
1C	•				50	NEGB		2	1	84	ANDA		2	2	B8	EORA		4	3	EC	LDD		5	2
1D	•				51	•				85	BITA		2	2	B9	ADCA		4	3	ED	STD		5	2
1E	•				52	•				86	LDAA		2	2	BA	ORAA		4	3	EE	LDX		5	2
1F	•				53	COMB		2	1	87	•				BB	ADDA		4	3	EF	STX	INDXD	5	2
20	BRA	REL	3	2	54	LSRB		2	1	88	EORA		2	2	BC	CPX		6	3	F0	SUBB	EXTND	4	3
21	BRN		3	2	55	•				89	ADCA		2	2	BD	JSR		6	3	F1	CMPB		4	3
22	BHI		3	2	56	RORB		2	1	8A	ORAA		2	2	BE	LDS		5	3	F2	SBCB		4	3
23	BLS		3	2	57	ASRB		2	1	8B	ADDA		2	2	BF	STS	EXTND	5	3	F3	ADDD		6	3
24	BCC		3	2	58	ASLB		2	1	8C	CPX	IMMED	4	3	C0	SUBB	IMMED	2	2	F4	ANDB		4	3
25	BCS		3	2	59	ROLB		2	1	8D	BSR	REL	6	2	C1	CMPB		2	2	F5	BITB		4	3
26	BNE		3	2	5A	DECB		2	1	8E	LDS	IMMED	3	3	C2	SBCB		2	2	F6	LDAB		4	3
27	BEQ		3	2	5B	•				8F	•				C3	ADDD		4	3	F7	STAB		4	3
28	BVC		3	2	5C	INCB		2	1	90	SUBA	DIR	3	2	C4	ANDB		2	2	F8	EORB		4	3
29	BVS		3	2	5D	TSTB		2	1	91	CMPA		3	2	C5	BITB		2	2	F9	ADCB		4	3
2A	BPL		3	2	5E	T				92	SBCA		3	2	C6	LDAB		2	2	FA	ORAB		4	3
2B	BMI		3	2	5F	CLRB	INHER	2	1	93	SUBD		5	2	C7	•				FB	ADDB		4	3
2C	BGE		3	2	60	NEG	INDXD	6	2	94	ANDA		3	2	C8	EORB		2	2	FC	LDD		5	3
2D	BLT		3	2	61	•				95	BITA		3	2	C9	ADCB		2	2	FD	STD		5	3
2E	BGT		3	2	62	•				96	LDAA		3	2	CA	ORAB		2	2	FE	LDX		5	3
2F	BLE	REL	3	2	63	COM		6	2	97	STAA		3	2	CB	ADDB		2	2	FF	STX	EXTND	5	3
30	TSX	INHER	3	1	64	LSR		6	2	98	EORA		3	2	CC	LDD		3	3					
31	INS		3	1	65	•				99	ADCA		3	2	CD	•								
32	PULA		4	1	66	ROR		6	2	9A	ORAA		3	2	CE	LDX	IMMED	3	3					
33	PULB		4	1	67	ASR	INDXD	6	2	9B	ADDA		3	2	CF	•								

* UNDEFINED OP CODE

NOTES:

1. Addressing Modes
   INHER ≡ Inherent INDXD ≡ Indexed IMMED ≡ Immediate
   REL ≡ Relative EXTND ≡ Extended DIR ≡ Direct
2. Unassigned opcodes are indicated by "•" and should not be executed.
3. Codes marked by "T" force the PC to function as a 16-bit counter.

PROGRAMMING MODEL

A programming model for the MC68701U4 is shown in Figure 8. Accumulator A can be concatenated with accumulator B and jointly referred to as accumulator D where A is the most significant byte. Any operation which modifies the double accumulator will also modify accumulators A and/or B. Other registers are defined as follows:

PROGRAM COUNTER – The program counter is a 16-bit register which always points to the next instruction.

STACK POINTER – The stack pointer is a 16-bit register which contains the address of the next available location in a pushdown/pullup (LIFO) queue. The stack resides in random-access memory at a location defined by the programmer.

INDEX REGISTER – The index register is a 16-bit register which can be used to store data or provide an address for the indexed mode of addressing.

ACCUMULATORS – The MPU contains two 8-bit accumulators, A and B, which are used to store operands and results from the arithmetic logic unit (ALU). They can also be concatenated and referred to as the D (double) accumulator.

CONDITION CODE REGISTER – The condition code register indicates the results of an instruction and includes the following five condition bits: negative (N), zero (Z), overflow (V), carry/borrow from MSB (C), and half carry from bit 3 (H). These bits are testable by the conditional branch instructions. Bit 4 is the interrupt mask (I bit) and inhibits all maskable interrupts when set. The two unused bits, B6 and B7, are read as ones.

ADDRESSING MODES

Six addressing modes can be used to reference memory. A summary of addressing modes for all instructions is presented in Tables 9, 10, 11, and 12 where execution times are provided in E cycles. Instruction execution times are summarized in Table 13. With an input frequency of 4 MHz, one E cycle is equivalent to one microsecond. A cycle-by-cycle description of bus activity for each instruction is provided in Table 14 and descriptions of selected instructions are shown in Figure 24.

IMMEDIATE ADDRESSING – The operand or "immediate byte(s)" is contained in the following byte(s) of the instruction where the number of bytes matches the size of the register. These are two or three byte instructions.

DIRECT ADDRESSING – The least significant byte of the operand address is contained in the second byte of the instruction and the most significant byte is assumed to be $00. Direct addressing allows the user to access $00 through $FF using two byte instructions and execution time is reduced by eliminating the additional memory access. In most applications, the 256-byte area is reserved for frequently referenced data.

EXTENDED ADDRESSING – The second and third bytes of the instruction contain the absolute address of the operand. These are three byte instructions.

INDEXED ADDRESSING – The unsigned offset contained in the second byte of the instruction is added with carry to the index register and is used to reference memory without changing the index register. These are two byte instructions.

INHERENT ADDRESSING – The operand(s) is a register and no memory reference is required. These are single byte instructions.

RELATIVE ADDRESSING – Relative addressing is used only for branch instructions. If the branch condition is true, the program counter is overwritten with the sum of a signed single byte displacement in the second byte of the instruction and the current program counter. This provides a branch range of −126 to +129 bytes from the first byte of the instruction. These are two byte instructions.

SUMMARY OF CYCLE-BY-CYCLE OPERATION

Table 14 provides a detailed description of the information present on the address bus, data bus, and the read/write (R/$\overline{W}$) line during each cycle of each instruction.

The information is useful in comparing actual with expected results during debug of both software and hardware as the program is executed. The information is categorized in groups according to addressing mode and number of cycles per instruction. In general, instructions with the same addressing mode and number of cycles execute in the same manner. Exceptions are indicated in the table.

Note that during MPU reads of internal locations, the resultant value will not appear on the external data bus except in mode 0. "High order" byte refers to the most significant byte of a 16-bit value. During unused bus cycles, the address bus is forced to $FFFF and R/$\overline{W}$ is high.

TABLE 9 — INDEX REGISTER AND STACK MANIPULATION INSTRUCTIONS

Pointer Operations	MNEM	Immed Op	Immed ~	Immed #	Direct Op	Direct ~	Direct #	Index Op	Index ~	Index #	Extend Op	Extend ~	Extend #	Inherent Op	Inherent ~	Inherent #	Boolean/ Arithmetic Operation	Condition Codes 5 H	4 I	3 N	2 Z	1 V	0 C
Compare Index Register	CPX	8C	4	3	9C	5	2	AC	6	2	BC	6	3				X – M:M + 1	•	•	↕	↕	↕	↕
Decrement Index Register	DEX													09	3	1	X – 1 → X	•	•	•	↕	•	•
Decrement Stack Pointer	DES													34	3	1	SP – 1 → SP	•	•	•	•	•	•
Increment Index Register	INX													08	3	1	X + 1 → X	•	•	•	↕	•	•
Increment Stack Pointer	INS													31	3	1	1 SP + 1 → SP	•	•	•	•	•	•
Load Index Register	LDX	CE	3	3	DE	4	2	EE	5	2	FE	5	3				M → X_H, (M + 1) → X_L	•	•	↕	↕	R	•
Load Stack Pointer	LDS	8E	3	3	9E	4	2	AE	5	2	BE	5	3				M → SP_H, (M + 1) → SP_L	•	•	↕	↕	R	•
Store Index Register	STX				DF	4	2	EF	5	2	FF	5	3				X_H → M, X_L → (M + 1)	•	•	↕	↕	R	•
Store Stack Pointer	STS				9F	4	2	AF	5	2	BF	5	3				SP_H → M, SP_L → (M + 1)	•	•	↕	↕	R	•
Index Reg → Stack Pointer	TXS													35	3	1	X – 1 → SP	•	•	•	•	•	•
Stack Pntr → Index Register	TSX													30	3	1	SP + 1 → X	•	•	•	•	•	•
Add	ABX													3A	3	1	B + X → X	•	•	•	•	•	•
Push Data	PSHX													3C	4	1	X_L → M_{SP}, SP – 1 → SP X_H → M_{SP}, SP – 1 → SP	•	•	•	•	•	•
Pull Data	PULX													38	5	1	SP + 1 → SP, M_{SP} → X_H SP + 1 → SP, M_{SP} → X_L	•	•	•	•	•	•

3

TABLE 10 — ACCUMULATOR AND MEMORY INSTRUCTIONS (Sheet 1 of 2)

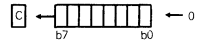
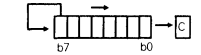
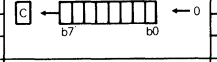
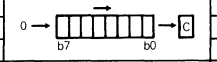
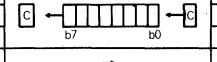
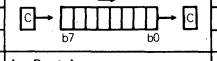
Accumulator and Memory Operations	MNEM	Immed Op	Immed ~	Immed #	Direct Op	Direct ~	Direct #	Index Op	Index ~	Index #	Extend Op	Extend ~	Extend #	Inher Op	Inher ~	Inher #	Boolean Expression	Condition Codes 5 H	4 I	3 N	2 Z	1 V	0 C
Add Accumulators	ABA													1B	2	1	A + B → A	↕	•	↕	↕	↕	↕
Add B to X	ABX													3A	3	1	00:B + X → X	•	•	•	•	•	•
Add with Carry	ADCA	89	2	2	99	3	2	A9	4	2	B9	4	3				A + M + C → A	↕	•	↕	↕	↕	↕
	ADCB	C9	2	2	D9	3	2	E9	4	2	F9	4	3				B + M + C → B	↕	•	↕	↕	↕	↕
Add	ADDA	8B	2	2	9B	3	2	AB	4	2	BB	4	3				A + M → A	↕	•	↕	↕	↕	↕
	ADDB	CB	2	2	DB	3	2	EB	4	2	FB	4	3				B + M → A	↕	•	↕	↕	↕	↕
Add Double	ADDD	C3	4	3	D3	5	2	E3	6	2	F3	6	3				D + M:M + 1 → D	•	•	↕	↕	↕	↕
And	ANDA	84	2	2	94	3	2	A4	4	2	B4	4	3				A•M → A	•	•	↕	↕	R	•
	ANDB	C4	2	2	D4	3	2	E4	4	2	F4	4	3				B•M → B	•	•	↕	↕	R	•
Shift Left, Arithmetic	ASL							68	6	2	78	6	3					•	•	↕	↕	↕	↕
	ASLA													48	2	1		•	•	↕	↕	↕	↕
	ASLB													58	2	1		•	•	↕	↕	↕	↕
Shift Left Double	ASLD													05	3	1		•	•	↕	↕	↕	↕
Shift Right, Arithmetic	ASR							67	6	2	77	6	3					•	•	↕	↕	↕	↕
	ASRA													47	2	1		•	•	↕	↕	↕	↕
	ASRB													57	2	1		•	•	↕	↕	↕	↕
Bit Test	BITA	85	2	2	95	3	2	A5	4	2	B5	4	3				A•M	•	•	↕	↕	R	•
	BITB	C5	2	2	D5	3	2	E5	4	2	F5	4	3				B•M	•	•	↕	↕	R	•
Compare Accumulators	CBA													11	2	1	A – B	•	•	↕	↕	↕	↕
Clear	CLR							6F	6	2	7F	6	3				00 → M	•	•	R	S	R	R
	CLRA													4F	2	1	00 → A	•	•	R	S	R	R
	CLRB													5F	2	1	00 → B	•	•	R	S	R	R
Compare	CMPA	81	2	2	91	3	2	A1	4	2	B1	4	3				A – M	•	•	↕	↕	↕	↕
	CMPB	C1	2	2	D1	3	2	E1	4	2	F1	4	3				B – M	•	•	↕	↕	↕	↕
1's Complement	COM							63	6	2	73	6	3				M → M	•	•	↕	↕	R	S
	COMA													43	2	1	A → A	•	•	↕	↕	R	S
	COMB													53	2	1	B → B	•	•	↕	↕	R	S

TABLE 10 — ACCUMULATOR AND MEMORY INSTRUCTIONS (Sheet 2 of 2)

Accumulator and Memory Operations		Immed			Direct			Index			Extend			Inher			Boolean Expression	Condition Codes 5 H	4 I	3 N	2 Z	1 V	0 C
	MNEM	Op	~	#	Op	~	#	Op	~	#	Op	~	#	Op	~	#							
Decimal Adjust, A	DAA													19	2	1	Adj binary sum to BCD	•	•	↕	↕	↕	↕
Decrement	DEC							6A	6	2	7A	6	3				M − 1 → M	•	•	↕	↕	↕	•
	DECA													4A	2	1	A − 1 → A	•	•	↕	↕	↕	•
	DECB													5A	2	1	B − 1 → B	•	•	↕	↕	↕	•
Exclusive OR	EORA	88	2	2	98	3	2	A8	4	2	B8	4	3				A ⊕ M → A	•	•	↕	↕	R	•
	EORB	C8	2	2	D8	3	2	E8	4	2	F8	4	3				B ⊕ M → B	•	•	↕	↕	R	•
Increment	INC							6C	6	2	7C	6	3				M + 1 → M	•	•	↕	↕	↕	•
	INCA													4C	2	1	A + 1 → A	•	•	↕	↕	↕	•
	INCB													5C	2	1	B + 1 → B	•	•	↕	↕	↕	•
Load Accumulators	LDAA	86	2	2	96	3	2	A6	4	2	B6	4	3				M → A	•	•	↕	↕	R	•
	LDAB	C6	2	2	D6	3	2	E6	4	2	F6	4	3				M → B	•	•	↕	↕	R	•
Load Double	LDD	CC	3	3	DC	4	2	EC	5	2	FC	5	3				M:M + 1 → D	•	•	↕	↕	R	•
Logical Shift, Left	LSL							68	6	2	78	6	3					•	•	↕	↕	↕	↕
	LSLA													48	2	1		•	•	↕	↕	↕	↕
	LSLB													58	2	1		•	•	↕	↕	↕	↕
	LSLD													05	3	2		•	•	↕	↕	↕	↕
Shift Right, Logical	LSR							64	6	2	74	6	3					•	•	R	↕	↕	↕
	LSRA													44	2	1		•	•	R	↕	↕	↕
	LSRB													54	2	1		•	•	R	↕	↕	↕
	LSRD													04	3	1		•	•	R	↕	↕	↕
Multiply	MUL													3D	10	1	A × B → D	•	•	•	•	•	↕
2's Complement (Negate)	NEG							60	6	2	70	6	3				00 − M → M	•	•	↕	↕	↕	↕
	NEGA													40	2	1	00 − A → A	•	•	↕	↕	↕	↕
	NEGB													50	2	1	00 − B → B	•	•	↕	↕	↕	↕
No Operation	NOP													01	2	1	PC + 1 → PC	•	•	•	•	•	•
Inclusive OR	ORAA	8A	2	2	9A	3	2	AA	4	2	BA	4	3				A + M → A	•	•	↕	↕	R	•
	ORAB	CA	2	2	DA	3	2	EA	4	2	FA	4	3				B + M → B	•	•	↕	↕	R	•
Push Data	PSHA													36	3	1	A → Stack	•	•	•	•	•	•
	PSHB													37	3	1	B → Stack	•	•	•	•	•	•
Pull Data	PULA													32	4	1	Stack → A	•	•	•	•	•	•
	PULB													33	4	1	Stack → B	•	•	•	•	•	•
Rotate Left	ROL							69	6	2	79	6	3					•	•	↕	↕	↕	↕
	ROLA													49	2	1		•	•	↕	↕	↕	↕
	ROLB													59	2	1		•	•	↕	↕	↕	↕
Rotate Right	ROR							66	6	2	76	6	3					•	•	↕	↕	↕	↕
	RORA													46	2	1		•	•	↕	↕	↕	↕
	RORB													56	2	1		•	•	↕	↕	↕	↕
Subtract Accumulator	SBA													10	2	1	A − B → A	•	•	↕	↕	↕	↕
Subtract with Carry	SBCA	82	2	2	92	3	2	A2	4	2	B2	4	3				A − M − C → A	•	•	↕	↕	↕	↕
	SBCB	C2	2	2	D2	3	2	E2	4	2	F2	4	3				B − M − C → B	•	•	↕	↕	↕	↕
Store Accumulators	STAA				97	3	2	A7	4	2	B7	4	3				A → M	•	•	↕	↕	R	•
	STAB				D7	3	2	E7	4	2	F7	4	3				B → M	•	•	↕	↕	R	•
	STD				DD	4	2	ED	5	2	FD	5	3				D → M:M + 1	•	•	↕	↕	R	•
Subtract	SUBA	80	2	2	90	3	2	A0	4	2	B0	4	3				A − M → A	•	•	↕	↕	↕	↕
	SUBB	C0	2	2	D0	3	2	E0	4	2	F0	4	3				B − M → B	•	•	↕	↕	↕	↕
Subtract Double	SUBD	83	4	3	93	5	2	A3	6	2	B3	6	3				D − M:M + 1 → D	•	•	↕	↕	↕	↕
Transfer Accumulator	TAB													16	2	1	A → B	•	•	↕	↕	R	•
	TBA													17	2	1	B → A	•	•	↕	↕	R	•
Test, Zero or Minus	TST							6D	6	2	7D	6	3				M − 00	•	•	↕	↕	R	R
	TSTA													4D	2	1	A − 00	•	•	↕	↕	R	R
	TSTB													5D	2	1	B − 00	•	•	↕	↕	R	R

The condition code register notes are listed after Table 12.

TABLE 11 — JUMP AND BRANCH INSTRUCTIONS

Operations	MNEM	Direct			Relative			Index			Extend			Inherent			Branch Test	Condition Code Reg.					
		Op	~	#	Op	~	#	Op	~	#	Op	~	#	Op	~	#		5 H	4 I	3 N	2 Z	1 V	0 C
Branch Always	BRA				20	3	2										None	•	•	•	•	•	•
Branch Never	BRN				21	3	2										None	•	•	•	•	•	•
Branch If Carry Clear	BCC				24	3	2										C = 0	•	•	•	•	•	•
Branch If Carry Set	BCS				25	3	2										C = 1	•	•	•	•	•	•
Branch If = Zero	BEQ				27	3	2										Z = 1	•	•	•	•	•	•
Branch If ≥ Zero	BGE				2C	3	2										N ⊕ V = 0	•	•	•	•	•	•
Branch If > Zero	BGT				2E	3	2										Z + (N ⊕ V) = 0	•	•	•	•	•	•
Branch If Higher	BHI				22	3	2										C + Z = 0	•	•	•	•	•	•
Branch If Higher or Same	BHS				24	3	2										C = 0	•	•	•	•	•	•
Branch If ≤ Zero	BLE				2F	3	2										Z + (N ⊕ V) = 1	•	•	•	•	•	•
Branch If Carry Set	BLO				25	3	2										C = 1	•	•	•	•	•	•
Branch If Lower Or Same	BLS				23	3	2										C + Z = 1	•	•	•	•	•	•
Branch If < Zero	BLT				2D	3	2										N ⊕ V = 1	•	•	•	•	•	•
Branch If Minus	BMI				2B	3	2										N = 1	•	•	•	•	•	•
Branch If Not Equal Zero	BNE				26	3	2										Z = 0	•	•	•	•	•	•
Branch If Overflow Clear	BVC				28	3	2										V = 0	•	•	•	•	•	•
Branch If Overflow Set	BVS				29	3	2										V = 1	•	•	•	•	•	•
Branch If Plus	BPL				2A	3	2										N = 0	•	•	•	•	•	•
Branch To Subroutine	BSR				8D	6	2											•	•	•	•	•	•
Jump	JMP							6E	3	2	7E	3	3				See Special Operations-Figure 24	•	•	•	•	•	•
Jump To Subroutine	JSR	9D	5	2				AD	6	2	BD	6	3					•	•	•	•	•	•
No Operation	NOP													01	2	1		•	•	•	•	•	•
Return From Interrupt	RTI													3B	10	1		↕	↕	↕	↕	↕	↕
Return From Subroutine	RTS													39	5	1	See Special Operations-Figure 24	•	•	•	•	•	•
Software Interrupt	SWI													3F	12	1		•	S	•	•	•	•
Wait For Interrupt	WAI													3E	9	1		•	•	•	•	•	•

TABLE 12 — CONDITION CODE REGISTER MANIPULATION INSTRUCTIONS

Operations	Inherent				Boolean Operation	Condition Code Register					
	MNEM	Op	~	#		5 H	4 I	3 N	2 Z	1 V	0 C
Clear Carry	CLC	0C	2	1	0 → C	•	•	•	•	•	R
Clear Interrupt Mask	CLI	0E	2	1	0 → I	•	R	•	•	•	•
Clear Overflow	CLV	0A	2	1	0 → V	•	•	•	•	R	•
Set Carry	SEC	0D	2	1	1 → C	•	•	•	•	•	S
Set Interrupt Mask	SEI	0F	2	1	1 → I	•	S	•	•	•	•
Set Overflow	SEV	0B	2	1	1 → V	•	•	•	•	S	•
Accumulator A → CCR	TAP	06	2	1	A → CCR	↕	↕	↕	↕	↕	↕
CCR → Accumulator A	TPA	07	2	1	CCR → A	•	•	•	•	•	•

LEGEND

Op Operation Code (Hexadecimal)
~ Number of MPU Cycles
M_{SP} Contents of memory location pointed to by Stack Pointer
Number of Program Bytes
+ Arithmetic Plus
− Arithmetic Minus
• Boolean AND
X Arithmetic Multiply
+ Boolean Inclusive OR
⊕ Boolean Exclusive OR
M Complement of M
→ Transfer Into
0 Bit = Zero
00 Byte = Zero

CONDITION CODE SYMBOLS

H Half-carry from bit 3
I Interrupt mask
N Negative (sign bit)
Z Zero (byte)
V Overflow, 2's complement
C Carry/Borrow from MSB
R Reset Always
S Set Always
↕ Affected
• Not Affected

TABLE 13 — INSTRUCTION EXECUTION TIMES IN E CYCLES

	ADDRESSING MODE					
	Immediate	Direct	Extended	Indexed	Inherent	Relative
ABA	●	●	●	●	2	●
ABX	●	●	●	●	3	●
ADC	2	3	4	4	●	●
ADD	2	3	4	4	●	●
ADDD	4	5	6	6	●	●
AND	2	3	4	4	●	●
ASL	●	●	6	6	2	●
ASLD	●	●	●	●	3	●
ASR	●	●	6	6	2	●
BCC	●	●	●	●	●	3
BCS	●	●	●	●	●	3
BEQ	●	●	●	●	●	3
BGE	●	●	●	●	●	3
BGT	●	●	●	●	●	3
BHI	●	●	●	●	●	3
BHS	●	●	●	●	●	3
BIT	2	3	4	4	●	●
BLE	●	●	●	●	●	3
BLO	●	●	●	●	●	3
BLS	●	●	●	●	●	3
BLT	●	●	●	●	●	3
BMI	●	●	●	●	●	3
BNE	●	●	●	●	●	3
BPL	●	●	●	●	●	3
BRA	●	●	●	●	●	3
BRN	●	●	●	●	●	3
BSR	●	●	●	●	●	6
BVC	●	●	●	●	●	3
BVS	●	●	●	●	●	3
CBA	●	●	●	●	2	●
CLC	●	●	●	●	2	●
CLI	●	●	●	●	2	●
CLR	●	●	6	6	2	●
CLV	●	●	●	●	2	●
CMP	2	3	4	4	●	●
COM	●	●	6	6	2	●
CPX	4	5	6	6	●	●
DAA	●	●	●	●	2	●
DEC	●	●	6	6	2	●
DES	●	●	●	●	3	●
DEX	●	●	●	●	3	●
EOR	2	3	4	4	●	●
INC	●	●	6	6	●	●
INS	●	●	●	●	3	●

	ADDRESSING MODE					
	Immediate	Direct	Extended	Indexed	Inherent	Relative
INX	●	●	●	●	3	●
JMP	●	●	3	3	●	●
JSR	●	5	6	6	●	●
LDA	2	3	4	4	●	●
LDD	3	4	5	5	●	●
LDS	3	4	5	5	●	●
LDX	3	4	5	5	●	●
LSL	●	●	6	6	2	●
LSLD	●	●	●	●	3	●
LSR	●	●	6	6	2	●
LSRD	●	●	●	●	3	●
MUL	●	●	●	●	10	●
NEG	●	●	6	6	2	●
NOP	●	●	●	●	2	●
ORA	2	3	4	4	●	●
PSH	●	●	●	●	3	●
PSHX	●	●	●	●	4	●
PUL	●	●	●	●	4	●
PULX	●	●	●	●	5	●
ROL	●	●	6	6	2	●
ROR	●	●	6	6	2	●
RTI	●	●	●	●	10	●
RTS	●	●	●	●	5	●
SBA	●	●	●	●	2	●
SBC	2	3	4	4	●	●
SEC	●	●	●	●	2	●
SEI	●	●	●	●	2	●
SEV	●	●	●	●	2	●
STA	●	3	4	4	●	●
STD	●	4	5	5	●	●
STS	●	4	5	5	●	●
STX	●	4	5	5	●	●
SUB	2	3	4	4	●	●
SUBD	4	5	6	6	●	●
SWI	●	●	●	●	12	●
TAB	●	●	●	●	2	●
TAP	●	●	●	●	2	●
TBA	●	●	●	●	2	●
TPA	●	●	●	●	2	●
TST	●	●	6	6	2	●
TSX	●	●	●	●	3	●
TXS	●	●	●	●	3	●
WAI	●	●	●	●	9	●

TABLE 14 — CYCLE-BY-CYCLE OPERATION (Sheet 1 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
IMMEDIATE					
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	2	1 2	Opcode Address Opcode Address+1	1 1	Opcode Operand Data
LDS LDX LDD	3	1 2 3	Opcode Address Opcode Address+1 Opcode Address+2	1 1 1	Opcode Operand Data (High Order Byte) Operand Data (Low Order Byte)
CPX SUBD ADDD	4	1 2 3 4	Opcode Address Opcode Address+1 Opcode Address+2 Address Bus FFFF	1 1 1 1	Opcode Operand Data (High Order Byte) Operand Data (Low Order Byte) Low Byte of Restart Vector
DIRECT					
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	3	1 2 3	Opcode Address Opcode Address+1 Address of Operand	1 1 1	Opcode Address of Operand Operand Data
STA	3	1 2 3	Opcode Address Opcode Address+1 Destination Address	1 1 0	Opcode Destination Address Data from Accumulator
LDS LDX LDD	4	1 2 3 4	Opcode Address Opcode Address+1 Address of Operand Operand Address+1	1 1 1 1	Opcode Address of Operand Operand Data (High Order Byte) Operand Data (Low Order Byte)
STS STX STD	4	1 2 3 4	Opcode Address Opcode Address+1 Address of Operand Address of Operand+1	1 1 0 0	Opcode Address of Operand Register Data (High Order Byte) Register Data (Low Order Byte)
CPX SUBD ADDD	5	1 2 3 4 5	Opcode Address Opcode Address+1 Operand Address Operand Address+1 Address Bus FFFF	1 1 1 1 1	Opcode Address of Operand Operand Data (High Order Byte) Operand Data (Low Order Byte) Low Byte of Restart Vector
JSR	5	1 2 3 4 5	Opcode Address Opcode Address+1 Subroutine Address Stack Pointer Stack Pointer−1	1 1 1 0 0	Opcode Irrelevant Data First Subroutine Opcode Return Address (Low Order Byte) Return Address (High Order Byte)

3

TABLE 14 — CYCLE-BY-CYCLE OPERATION (Sheet 2 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/W Line	Data Bus
EXTENDED					
JMP	3	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Jump Address (High Order Byte)
		3	Opcode Address + 2	1	Jump Address (Low Order Byte)
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	4	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Address of Operand
		3	Opcode Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	1	Operand Data
STA	4	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Destination Address (High Order Byte)
		3	Opcode Address + 2	1	Destination Address (Low Order Byte)
		4	Operand Destination Address	0	Data from Accumulator
LDS LDX LDD	5	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Address of Operand (High Order Byte)
		3	Opcode Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	1	Operand Data (High Order Byte)
		5	Address of Operand + 1	1	Operand Data (Low Order Byte)
STS STX STD	5	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Address of Operand (High Order Byte)
		3	Opcode Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	0	Operand Data (High Order Byte)
		5	Address of Operand + 1	0	Operand Data (Low Order Byte)
ASL LSR ASR NEG CLR ROL COM ROR DEC TST* INC	6	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Address of Operand (High Order Byte)
		3	Opcode Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	1	Current Operand Data
		5	Address Bus FFFF	1	Low Byte of Restart Vector
		6	Address of Operand	0	New Operand Data
CPX SUBD ADDD	6	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Operand Address (High Order Byte)
		3	Opcode Address + 2	1	Operand Address (Low Order Byte)
		4	Operand Address	1	Operand Data (High Order Byte)
		5	Operand Address + 1	1	Operand Data (Low Order Byte)
		6	Address Bus FFFF	1	Low Byte of Restart Vector
JSR	6	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Address of Subroutine (High Order Byte)
		3	Opcode Address + 2	1	Address of Subroutine (Low Order Byte)
		4	Subroutine Starting Address	1	Opcode of Next Instruction
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer − 1	0	Return Address (High Order Byte)

*TST does not perform the write cycle during the sixth cycle. The sixth cycle is another address bus = $FFFF.

TABLE 14 — CYCLE-BY-CYCLE OPERATION (Sheet 3 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
INDEXED					
JMP	3	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
ADC EOR	4	1	Opcode Address	1	Opcode
ADD LDA		2	Opcode Address + 1	1	Offset
AND ORA		3	Address Bus FFFF	1	Low Byte of Restart Vector
BIT SBC		4	Index Register Plus Offset	1	Operand Data
CMP SUB					
STA	4	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	0	Operand Data
LDS	5	1	Opcode Address	1	Opcode
LDX		2	Opcode Address + 1	1	Offset
LDD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	1	Operand Data (High Order Byte)
		5	Index Register Plus Offset + 1	1	Operand Data (Low Order Byte)
STS	5	1	Opcode Address	1	Opcode
STX		2	Opcode Address + 1	1	Offset
STD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	0	Operand Data (High Order Byte)
		5	Index Register Plus Offset + 1	0	Operand Data (Low Order Byte)
ASL LSR	6	1	Opcode Address	1	Opcode
ASR NEG		2	Opcode Address + 1	1	Offset
CLR ROL		3	Address Bus FFFF	1	Low Byte of Restart Vector
COM ROR		4	Index Register Plus Offset	1	Current Operand Data
DEC TST*		5	Address Bus FFFF	1	Low Byte of Restart Vector
INC		6	Index Register Plus Offset	0	New Operand Data
CPX	6	1	Opcode Address	1	Opcode
SUBD		2	Opcode Address + 1	1	Offset
ADDD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register + Offset	1	Operand Data (High Order Byte)
		5	Index Register + Offset + 1	1	Operand Data (Low Order Byte)
		6	Address Bus FFFF		Low Byte of Restart Vector
JSR	6	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register + Offset	1	First Subroutine Opcode
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer − 1	0	Return Address (High Order Byte)

*TST does not perform the write cycle during the sixth cycle. The sixth cycle is another address bus = $FFFF.

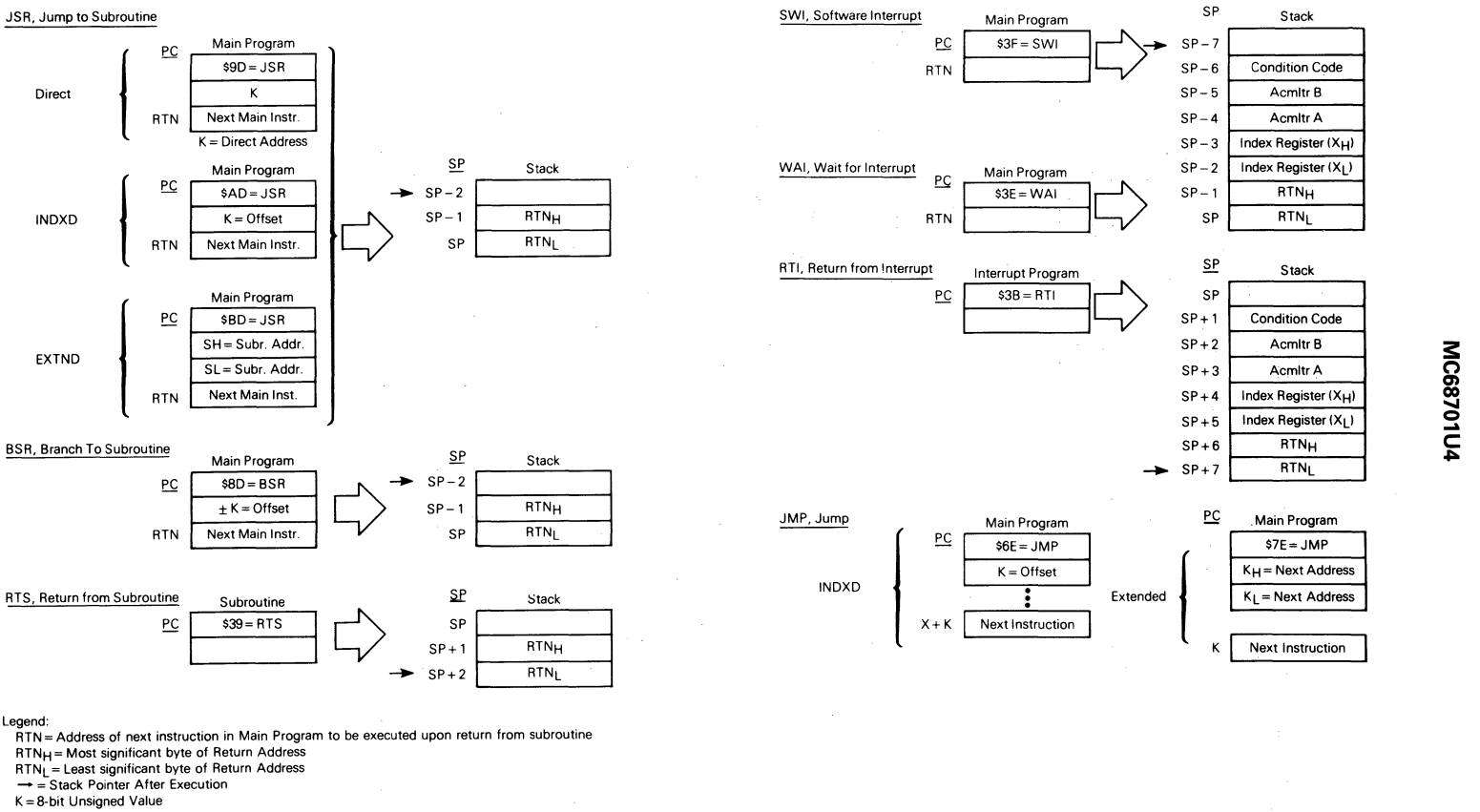
TABLE 14 — CYCLE-BY-CYCLE OPERATION (Sheet 4 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
INHERENT					
ABA DAA SEC ASL DEC SEI ASR INC SEV CBA LSR TAB CLC NEG TAP CLI NOP TBA CLR ROL TPA CLV ROR TST COM SBA	2	1 2	Opcode Address Opcode Address + 1	1 1	Opcode Opcode of Next Instruction
ABX	3	1 2 3	Opcode Address Opcode Address + 1 Address Bus FFFF	1 1 1	Opcode Irrelevant Data Low Byte of Restart Vector
ASLD LSRD	3	1 2 3	Opcode Address Opcode Address + 1 Address Bus FFFF	1 1 1	Opcode Irrelevant Data Low Byte of Restart Vector
DES INS	3	1 2 3	Opcode Address Opcode Address + 1 Previous Stack Pointer Contents	1 1 1	Opcode Opcode of Next Instruction Irrelevant Data
INX DEX	3	1 2 3	Opcode Address Opcode Address + 1 Address Bus FFFF	1 1 1	Opcode Opcode of Next Instruction Low Byte of Restart Vector
PSHA PSHB	3	1 2 3	Opcode Address Opcode Address + 1 Stack Pointer	1 1 0	Opcode Opcode of Next Instruction Accumulator Data
TSX	3	1 2 3	Opcode Address Opcode Address + 1 Stack Pointer	1 1 1	Opcode Opcode of Next Instruction Irrelevant Data
TXS	3	1 2 3	Opcode Address Opcode Address + 1 Address Bus FFFF	1 1 1	Opcode Opcode of Next Instruction Low Byte of Restart Vector
PULA PULB	4	1 2 3 4	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer + 1	1 1 1 1	Opcode Opcode of Next Instruction Irrelevant Data Operand Data from Stack
PSHX	4	1 2 3 4	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer − 1	1 1 0 0	Opcode Irrelevant Data Index Register (Low Order Byte) Index Register (High Order Byte)
PULX	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer + 1 Stack Pointer + 2	1 1 1 1 1	Opcode Irrelevant Data Irrelevant Data Index Register (High Order Byte) Index Register (Low Order Byte)
RTS	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer + 1 Stack Pointer + 2	1 1 1 1 1	Opcode Irrelevant Data Irrelevant Data Address of Next Instruction (High Order Byte) Address of Next Instruction (Low Order Byte)
WAI	9	1 2 3 4 5 6 7 8 9	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer − 1 Stack Pointer − 2 Stack Pointer − 3 Stack Pointer − 4 Stack Pointer − 5 Stack Pointer − 6	1 1 0 0 0 0 0 0 0	Opcode Opcode of Next Instruction Return Address (Low Order Byte) Return Address (High Order Byte) Index Register (Low Order Byte) Index Register (High Order Byte) Contents of Accumulator A Contents of Accumulator B Contents of Condition Code Register

TABLE 14 — CYCLE-BY-CYCLE OPERATION (Sheet 5 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
INHERENT (Continued)					
MUL	10	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Irrelevant Data
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Address Bus FFFF	1	Low Byte of Restart Vector
		5	Address Bus FFFF	1	Low Byte of Restart Vector
		6	Address Bus FFFF	1	Low Byte of Restart Vector
		7	Address Bus FFFF	1	Low Byte of Restart Vector
		8	Address Bus FFFF	1	Low Byte of Restart Vector
		9	Address Bus FFFF	1	Low Byte of Restart Vector
		10	Address Bus FFFF	1	Low Byte of Restart Vector
RTI	10	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Irrelevant Data
		3	Stack Pointer	1	Irrelevant Data
		4	Stack Pointer + 1	1	Contents of Condition Code Register from Stack
		5	Stack Pointer + 2	1	Contents of Accumulator B from Stack
		6	Stack Pointer + 3	1	Contents of Accumulator A from Stack
		7	Stack Pointer + 4	1	Index Register from Stack (High Order Byte)
		8	Stack Pointer + 5	1	Index Register from Stack (Low Order Byte)
		9	Stack Pointer + 6	1	Next Instruction Address from Stack (High Order Byte)
		10	Stack Pointer + 7	1	Next Instruction Address from Stack (Low Order Byte)
SWI	12	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Irrelevant Data
		3	Stack Pointer	0	Return Address (Low Order Byte)
		4	Stack Pointer − 1	0	Return Address (High Order Byte)
		5	Stack Pointer − 2	0	Index Register (Low Order Byte)
		6	Stack Pointer − 3	0	Index Register (High Order Byte)
		7	Stack Pointer − 4	0	Contents of Accumulator A
		8	Stack Pointer − 5	0	Contents of Accumulator B
		9	Stack Pointer − 6	0	Contents of Condition Code Register
		10	Stack Pointer − 7	1	Irrelevant Data
		11	Vector Address FFFA (Hex)	1	Address of Subroutine (High Order Byte)
		12	Vector Address FFFB (Hex)	1	Address of Subroutine (Low Order Byte)
RELATIVE					
BCC BHT BNE BLO BCS BLE BPL BHS BEQ BLS BRA BRN BGE BLT BVC BGT BMI BVS	3	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Branch Offset
		3	Address Buss FFFF	1	Low Byte of Restart Vector
BSR	6	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Branch Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Subroutine Starting Address	1	Opcode of Next Instruction
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer − 1	0	Return Address (High Order Byte)

FIGURE 24 — SPECIAL OPERATIONS



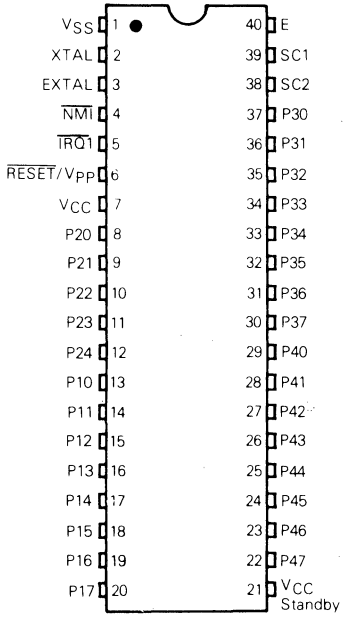
Legend:
RTN = Address of next instruction in Main Program to be executed upon return from subroutine
RTN_H = Most significant byte of Return Address
RTN_L = Least significant byte of Return Address
→ = Stack Pointer After Execution
K = 8-bit Unsigned Value

ORDERING INFORMATION

GENERIC INFORMATION
($T_A = 0°$ to $70°C$)

Package Type	Frequency	Generic Number
Cerdip — S Suffix	1.0 MHz 1.25 MHz	MC68701U4S MC68701U4S-1

PIN ASSIGNMENTS



3

MC6802

Microprocessor With Clock and Optional RAM

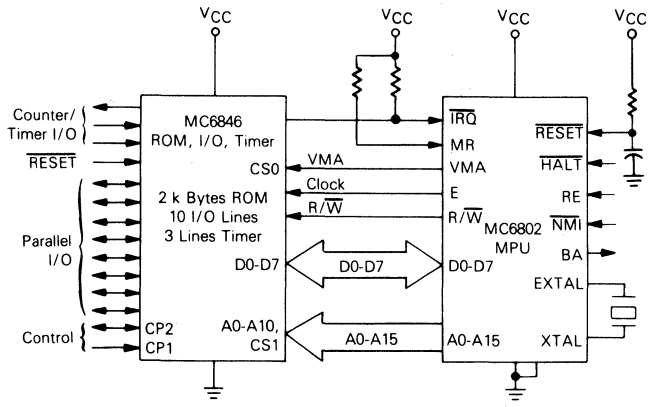
The MC6802 is a monolithic 8-bit microprocessor that contains all the registers and accumulators of the present MC6800 plus an internal clock oscillator and driver on the same chip. In addition, the MC6802 has 128 bytes of on-board RAM located at hex addresses $0000 to $007F. The first 32 bytes of RAM, at hex addresses $0000 to $001F, may be retained in a low power mode by utilizing V_{CC} standby; thus, facilitating memory retention during a power-down situation.

The MC6802 is completely software compatible with the MC6800 as well as the entire M6800 family of parts. Hence, the MC6802 is expandable to 64K words.

- On-Chip Clock Circuit
- 128×8 Bit On-Chip RAM
- 32 Bytes of RAM are Retainable
- Software-Compatible with the MC6800
- Expandable to 64K Words
- Standard TTL-Compatible Inputs and Outputs
- 8-Bit Word Size
- 16-Bit Memory Addressing
- Interrupt Capability

3

TYPICAL MICROCOMPUTER



This block diagram shows a typical cost effective microcomputer. The MPU is the center of the microcomputer system and is shown in a minimum system interfacing with a ROM combination chip. It is not intended that this system be limited to this function but that it be expandable with other parts in the M6800 Microcomputer family.

This document contains information on a new product. Specifications and information herein are subject to change without notice.

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range MC6802, MC680A02, MC680B02 MC6802C, MC680A02C	T_A	 0 to +70 −40 to +85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C

This input contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high-impedance circuit. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Average Thermal Resistance (Junction to Ambient) Plastic	θ_{JA}	100	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

DC ELECTRICAL CHARACTERISTICS (V_{DD} = +5.0 Vdc ± 0.5%, V_{SS} = 0, T_A = 0 to 70°C, unless otherwise noted)

Characteristic		Symbol	Min	Typ	Max	Unit
Input High Voltage	Logic, EXTAL RESET	V_{IH}	V_{SS} + 2.0 V_{SS} + 4.0	— —	V_{CC} V_{CC}	V
Input Low Voltage	Logic, EXTAL, RESET	V_{IL}	V_{SS} − 0.3	—	V_{SS} + 0.8	V
Input Leakage Current (V_{in} = 0 to 5.25 V, V_{DD} = max)	Logic	I_{in}	—	1.0	2.5	μA
Output High Voltage (I_{Load} = −205 μA, V_{CC} = min) (I_{Load} = −145 μA, V_{CC} = min) (I_{Load} = −100 μA, V_{CC} = min)	 D0-D7 A0-A15, R/W, VMA, E BA	V_{OH}	 V_{SS} + 2.4 V_{SS} + 2.4 V_{SS} + 2.4	 — — —	 — — —	V
Output Low Voltage (I_{Load} = 1.6 mA, V_{CC} = min)		V_{OL}	—	—	V_{SS} + 0.4	V
Internal Power Dissipation (Measured at T_A = 0°C)		P_{INT}	—	0.750	1.0	W
V_{DD} Standby	Power Down Power Up	V_{SBB} V_{SB}	4.0 4.75	— —	5.25 5.25	V
Standby Current		I_{SBB}	—	—	8.0	mA
Capacitance # (V_{in} = 0, T_A = 25°C, f = 1.0 MHz)	D0-D7 Logic Inputs, EXTAL A0-A15, R/W, VMA	C_{in} C_{out}	— — —	10 6.5 —	12.5 10 12	pF

*In power-down mode, maximum power dissipation is less than 42 mW.
#Capacitances are periodically sampled rather than 100% tested.

3

CONTROL TIMING (V_{CC} = 5.0 V ±5%, V_{SS} = 0, T_A = T_L to T_H), unless otherwise noted)

Characteristic	Symbol	MC6802		MC68A02		MC68B02		Unit
		Min	Max	Min	Max	Min	Max	
Frequency of Operation	f_o	0.1	1.0	0.1	1.5	0.1	2.0	MHz
Crystal Frequency	f_{XTAL}	1.0	4.0	1.0	6.0	1.0	8.0	MHz
External Oscillator Frequency	$4xf_o$	0.4	4.0	0.4	6.0	0.4	8.0	MHz
Crystal Oscillator Start Up Time	t_{rc}	100	—	100	—	100	—	ms
Processor Controls (HALT, MR, RE, RESET, IRQ, NMI) Processor Control Setup Time Processor Control Rise and Fall Time (Does Not Apply to RESET)	 t_{PCS} t_{PCr}, t_{PCf}	 200 —	 — 100	 140 —	 — 100	 110 —	 — 100	ns

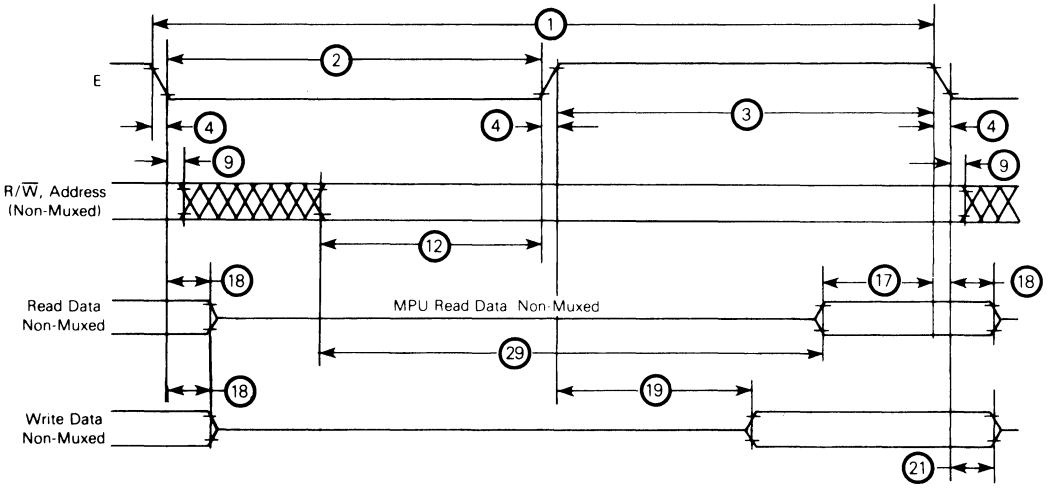
BUS TIMING CHARACTERISTICS

Ident. Number	Characteristic	Symbol	MC6802		MC68A02		MC68B02		Unit
			Min	Max	Min	Max	Min	Max	
1	Cycle Time	t_{cyc}	1.0	10	0.667	10	0.5	10	μs
2	Pulse Width, E Low	PW_{EL}	450	5000	280	5000	210	5000	ns
3	Pulse Width, E High	PW_{EH}	450	9500	280	9700	220	9700	ns
4	Clock Rise and Fall Time	t_r, t_f	—	25	—	25	—	25	ns
9	Address Hold Time*	t_{AH}	20	—	20	—	20	—	ns
12	Non-Muxed Address Valid Time to E (see Note 4)	t_{AV1} t_{AV2}	160 —	— 270	100 —	— —	50 —	— —	ns
17	Read Data Setup Time	t_{DSR}	100	—	70	—	60	—	ns
18	Read Data Hold Time	t_{DHR}	10	—	10	—	10	—	ns
19	Write Data Delay Time	t_{DDW}	—	225	—	170	—	160	ns
21	Write Data Hold Time*	t_{DHW}	30	—	20	—	20	—	ns
29	Usable Access Time (see Note 4)	t_{ACC}	535	—	335	—	235	—	ns

*Address and data hold times are periodically tested rather than 100% tested.

3

FIGURE 2 – BUS TIMING



NOTES:
1. Voltage levels shown are $V_L \leq 0.4$ V, $V_H \geq 2.4$ V, unless otherwise specified.
2. Measurement points shown are 0.8 V and 2.0 V, unless otherwise noted.
3. Usable access time is computed by: 12+3+4−17.
4. If programs are not executed from on-board RAM, TAV1 applies. If programs are to be stored and executed from on-board RAM, TAV2 applies. For normal data storage in the on-board RAM, this extended delay does not apply. Programs cannot be executed from on-board RAM when using A and B parts (MC68A02, MC68B02). On-board RAM can be used for data storage with all parts.
5. All electrical and control characteristics are referenced from: $T_L = 0°C$ minimum and $T_H = 70°C$ maximum.

FIGURE 3 — BUS TIMING TEST LOAD

C = 130 pF for D0-D7, E
= 90 pF for A0-A15, R/$\overline{W}$, and VMA
= 30 pF for BA
R = 11.7 kΩ for D0-D7, E
= 16.5 kΩ for A0-A15, R/$\overline{W}$, and VMA
= 24 kΩ for BA

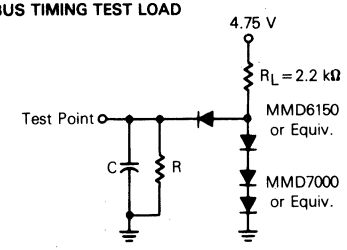


FIGURE 4 — TYPICAL DATA BUS OUTPUT DELAY versus CAPACITIVE LOADING

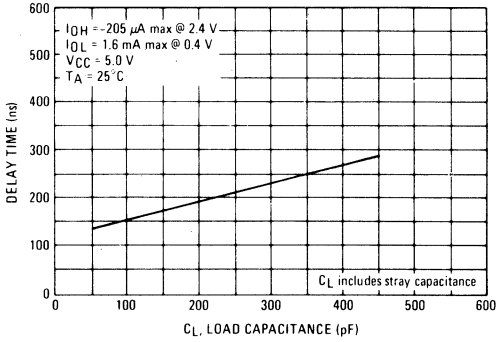


FIGURE 5 — TYPICAL READ/WRITE, VMA AND ADDRESS OUTPUT DELAY versus CAPACITIVE LOADING

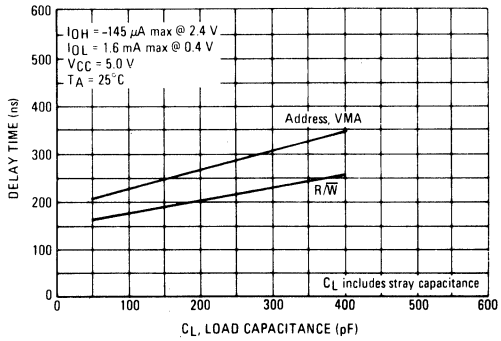
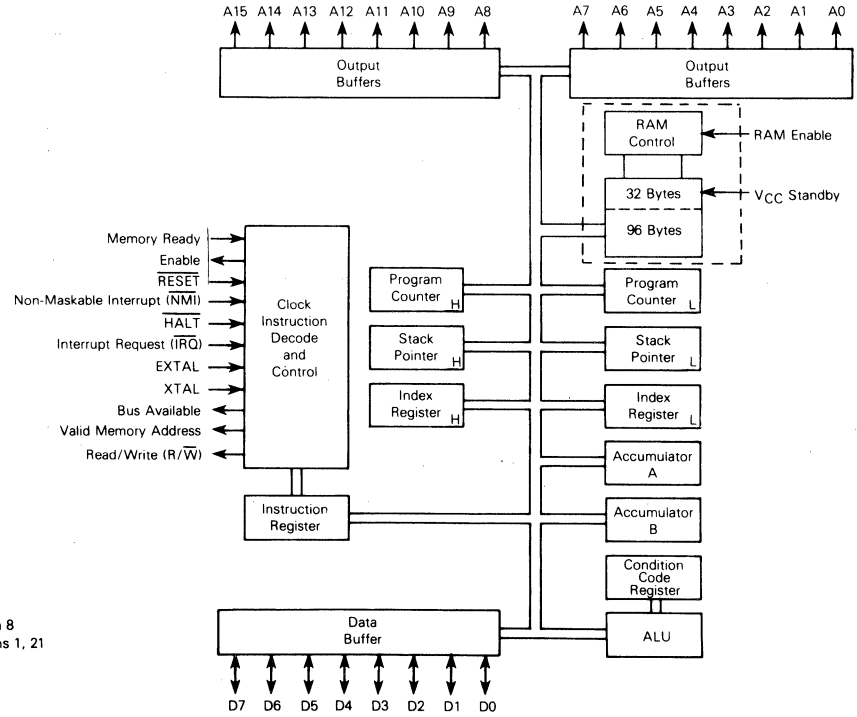


FIGURE 6 — EXPANDED BLOCK DIAGRAM



MPU REGISTERS

A general block diagram of the MC6802 is shown in Figure 1. As shown, the number and configuration of the registers are the same as for the MC6800. The 128×8-bit RAM* has been added to the basic MPU. The first 32 bytes can be retained during powerup and power-down conditions via the RE signal.

The MPU has three 16-bit registers and three 8-bit registers available for use by the programmer (Figure 7).

PROGRAM COUNTER

The program conter is a two byte (16-bit) register that points to the current program address.

STACK POINTER

The stack pointer is a two byte register that contains the address of the next available location in an external pushdown/pop-up stack. This stack is normally a random access read/write memory that may have any location (address) that is convenient. In those applications that require storage of information in the stack when power is lost, the stack must be non-volatile.

INDEX REGISTER

The index register is a two byte register that is used to store data or a 16-bit memory address for the indexed mode of memory addressing.

ACCUMULATORS

The MPU contains two 8-bit accumulators that are used to hold operands and results from an arithmetic logic unit (ALU).

CONDITION CODE REGISTER

The condition code register indicates the results of an Arithmetic Logic Unit operation: Negative (N), Zero (Z), Overflow (V), Carry from bit 7 (C), and Half Carry from bit 3 (H). These bits of the Condition Code Register are used as testable conditions for the conditional branch instructions. Bit 4 is the interrupt mask bit (I). The unused bits of the Condition Code Register (b6 and b7) are ones.

Figure 8 shows the order of saving the microprocessor status within the stack.

3

*If programs are not executed from on-board RAM, TAV1 applies. If programs are to be stored and executed from on-board RAM, TAV2 applies. For normal data storage in the on-board RAM, this extended delay does not apply. Programs cannot be executed from on-board RAM when using A and B parts (MC68A02 and MC68B02). On-board RAM can be used for data storage with all parts.

FIGURE 7 — PROGRAMMING MODEL OF THE MICROPROCESSING UNIT

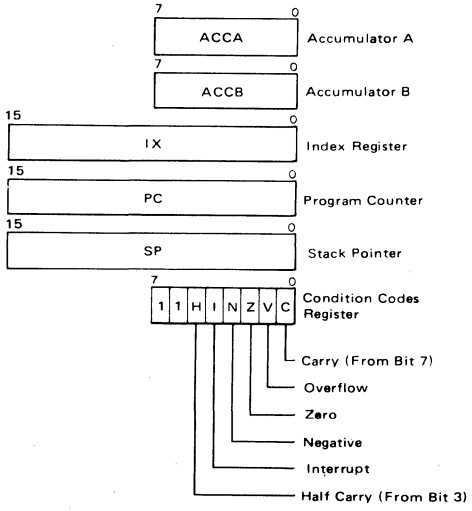
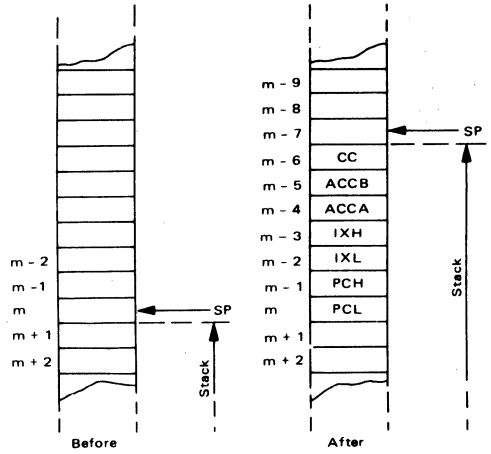


FIGURE 8 — SAVING THE STATUS OF THE MICROPROCESSOR IN THE STACK

SP = Stack Pointer
CC = Condition Codes (Also called the Processor Status Byte)
ACCB = Accumulator B
ACCA = Accumulator A
IXH = Index Register, Higher Order 8 Bits
IXL = Index Register, Lower Order 8 Bits
PCH = Program Counter, Higher Order 8 Bits
PCL = Program Counter, Lower Order 8 Bits



3

MPU SIGNAL DESCRIPTION

Proper operation of the MPU requires that certain control and timing signals be provided to accomplish specific functions and that other signal lines be monitored to determine the state of the processor. These control and timing signals are similar to those of the MC6800 except that TSC, DBE, $\phi 1$, $\phi 2$ input, and two unused pins have been eliminated, and the following signal and timing lines have been added:

RAM Enable (RE)
Crystal Connections EXTAL and XTAL
Memory Ready (MR)
V_{CC} Standby
Enable $\phi 2$ Output (E)

The following is a summary of the MPU signals:

ADDRESS BUS (A0-A15)

Sixteen pins are used for the address bus. The outputs are capable of driving one standard TTL load and 90 pF. These lines do not have three-state capability.

DATA BUS (D0-D7)

Eight pins are used for the data bus. It is bidirectional, transferring data to and from the memory and peripheral devices. It also has three-state output buffers capable of driving one standard TTL load and 130 pF.

Data bus will be in the output mode when the internal RAM is accessed and RE will be high. This prohibits external data entering the MPU. It should be noted that the internal RAM is fully decoded from $0000 to $007F. External RAM at $0000 to $007F must be disabled when internal RAM is accessed.

$\overline{\text{HALT}}$

When this input is in the low state, all activity in the machine will be halted. This input is level sensitive. In the $\overline{\text{HALT}}$ mode, the machine will stop at the end of an instruction, bus available will be at a high state, valid memory address will be at a low state. The address bus will display the address of the next instruction.

To ensure single instruction operation, transition of the $\overline{\text{HALT}}$ line must occur t_{PCS} before the rising edge of E and the $\overline{\text{HALT}}$ line must go high for one clock cycle.

$\overline{\text{HALT}}$ should be tied high if not used. This is good engineering design practice in general and necessary to ensure proper operation of the part.

READ/WRITE (R/$\overline{\text{W}}$)

This TTL-compatible output signals the peripherals and memory devices whether the MPU is in a read (high) or write (low) state. The normal standby state of this signal is read (high). When the processor is halted, it will be in the read state. This output is capable of driving one standard TTL load and 90 pF.

VALID MEMORY ADDRESS (VMA)

This output indicates to peripheral devices that there is a valid address on the address bus. In normal operation, this signal should be utilized for enabling peripheral interfaces such as the PIA and ACIA. This signal is not three-state. One standard TTL load and 90 pF may be directly driven by this active high signal.

BUS AVAILABLE (BA) — The bus available signal will normally be in the low state; when activated, it will go to the high state indicating that the microprocessor has stopped and that the address bus is available (but not in a three-state condition). This will occur if the $\overline{\text{HALT}}$ line is in the low state or the processor is in the WAIT state as a result of the execution of a WAIT instruction. At such time, all three-state output drivers will go to their off-state and other outputs to their normally inactive level. The processor is removed from the

WAIT state by the occurrence of a maskable (mask bit I = 0) or nonmaskable interrupt. This output is capable of driving one standard TTL load and 30 pF.

INTERRUPT REQUEST ($\overline{IRQ}$)

A low level on this input requests that an interrupt sequence be generated within the machine. The processor will wait until it completes the current instruction that is being excuted before it recognizes the request. At that time, if the interrupt mask bit in the condition code register is not set, the machine will begin an interrupt sequence. The index register, program counter, accumulators, and condition code register are stored away on the stack. Next the MPU will respond to the interrupt request by setting the interrupt mask bit high so that no further interrupts may occur. At the end of the cycle, a 16-bit vectoring address which is located in memory locations $FFF8 and $FFF9 is loaded which causes the MPU to branch to an interrupt routine in memory.

The $\overline{HALT}$ line must be in the high state for interrupts to be serviced. Interrupts will be latched internally while $\overline{HALT}$ is low.

A nominal 3 kΩ pullup resistor to V_{CC} should be used for wire-OR and optimum control of interrupts. $\overline{IRQ}$ may be tied directly to V_{CC} if not used.

$\overline{RESET}$

This input is used to reset and start the MPU from a power-down condition, resulting from a power failure or an initial start-up of the processor. When this line is low, the MPU is inactive and the information in the registers will be lost. If a high level is detected on the input, this will signal the MPU to begin the restart sequence. This will start execution of a routine to initialize the processor from its reset condition. All the higher order address lines will be forced high. For the restart, the last two ($FFFE, $FFFF) locations in memory will be used to load the program that is addressed by the program counter. During the restart routine, the interrupt mask bit is set and must be reset before the MPU can be interrupted by $\overline{IRQ}$. Power-up and reset timing and power-down sequences are shown in Figures 9 and 10, respectively.

$\overline{RESET}$, when brought low, must be held low at least three clock cycles. This allows adequate time to respond internally to the reset. This is independent of the t_{rc} power-up reset that is required.

When $\overline{RESET}$ is released it *must* go through the low-to-high threshold without bouncing, oscillating, or otherwise causing an erroneous reset (less than three clock cycles). This may cause improper MPU operation until the next valid reset.

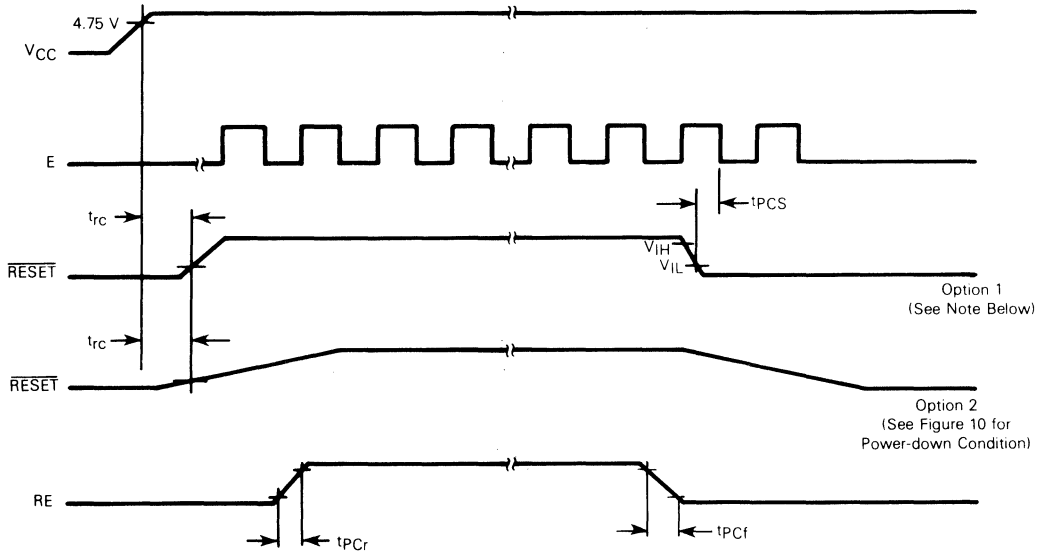
NON-MASKABLE INTERRUPT ($\overline{NMI}$)

A low-going edge on this input requests that a non-maskable interrupt sequence be generated within the processor. As with the interrupt request signal, the processor will complete the current instruction that is being executed before it recognizes the $\overline{NMI}$ signal. The interrupt mask bit in the condition code register has no effect on $\overline{NMI}$.

The index register, program counter, accumulators, and condition code registers are stored away on the stack. At the end of the cycle, a 16-bit vectoring address which is located in memory locations $FFFC and $FFFD is loaded causing the MPU to branch to an interrupt service routine in memory.

A nominal 3 kΩ pullup resistor to V_{CC} should be used for wire-OR and optimum control of interrupts. $\overline{NMI}$ may be tied

FIGURE 9 — POWER-UP AND RESET TIMING



NOTE: If option 1 is chosen, $\overline{RESET}$ and RE pins can be tied together.

directly to V_{CC} if not used.

Inputs $\overline{IRQ}$ and $\overline{NMI}$ are hardware interrupt lines that are sampled when E is high and will start the interrupt routine on a low E following the completion of an instruction.

Figure 11 is a flowchart describing the major decision paths and interrupt vectors of the microprocessor. Table 1 gives the memory map for interrupt vectors.

TABLE 1 — MEMORY MAP FOR INTERRUPT VECTORS

Vector		Description
MS	LS	
$FFFE	$FFFF	Restart
$FFFC	$FFFD	Non-Maskable Interrupt
$FFFA	$FFFB	Software Interrupt
$FFF8	$FFF9	Interrupt Request

FIGURE 10 — POWER-DOWN SEQUENCE

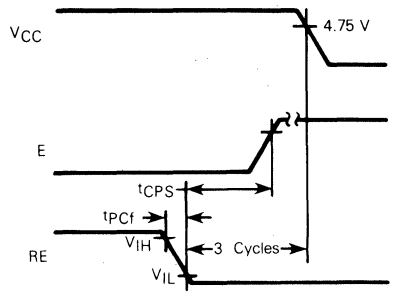


FIGURE 11 — MPU FLOWCHART

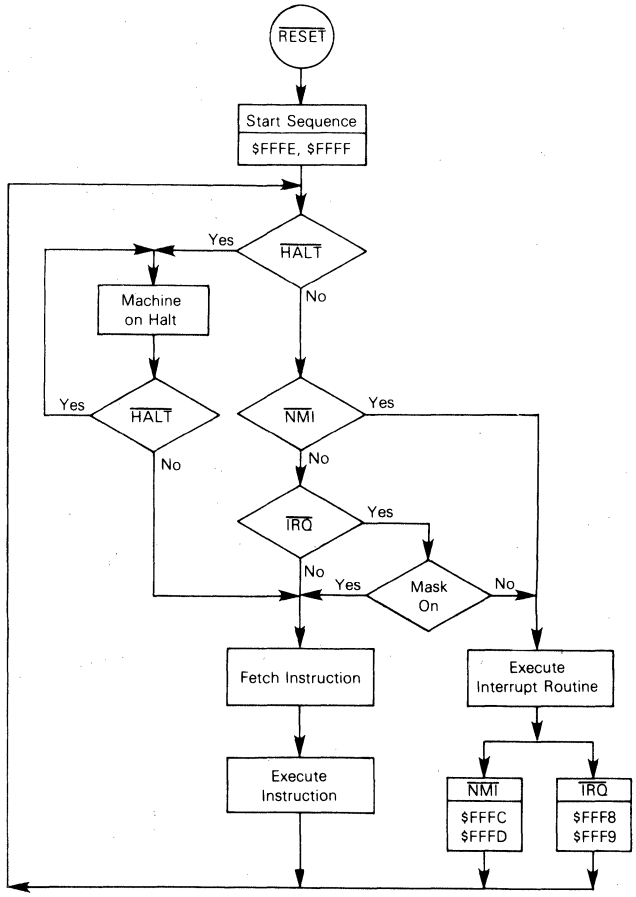
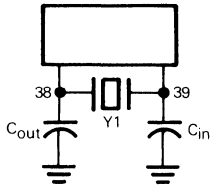
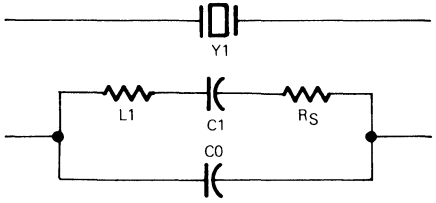


FIGURE 12 — CRYSTAL SPECIFICATIONS



Y1	C_{in}	C_{out}
3.58 MHz	27 pF	27 pF
4 MHz	27 pF	27 pF
6 MHz	20 pF	20 pF
8 MHz	18 pF	18 pF

Crystal Loading



Nominal Crystal Parameters*

	3.58 MHz	4.0 MHz	6.0 MHz	8.0 MHz
R_S	60 Ω	50 Ω	30-50 Ω	20-40 Ω
C0	3.5 pF	6.5 pF	4-6 pF	4-6 pF
C1	0.015 pF	0.025 pF	0.01-0.02 pF	0.01-0.02 pF
Q	>40K	>30K	>20K	>20K

*These are representative AT-cut parallel resonance crystal parameters only. Crystals of other types of cuts may also be used.

Figure 13 — SUGGESTED PC BOARD LAYOUT

Example of Board Design Using the Crystal Oscillator

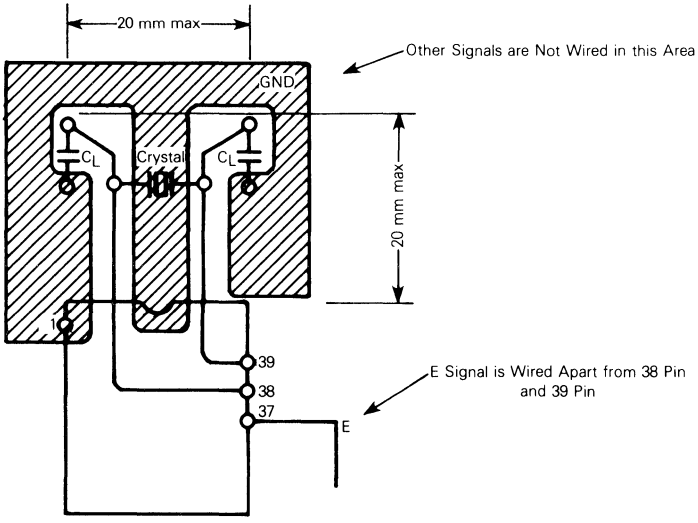
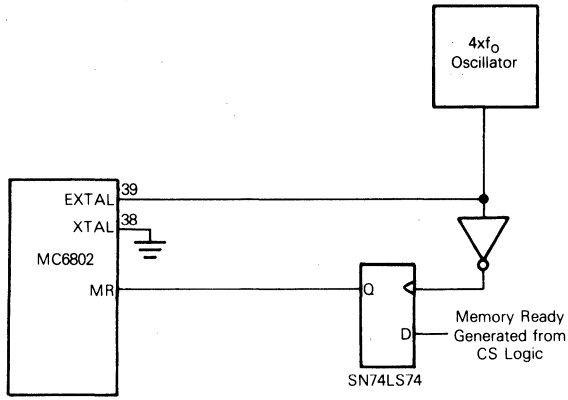


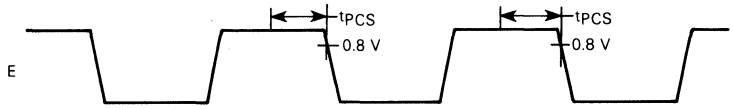
FIGURE 14 — MEMORY READY SYNCHRONIZATION



3

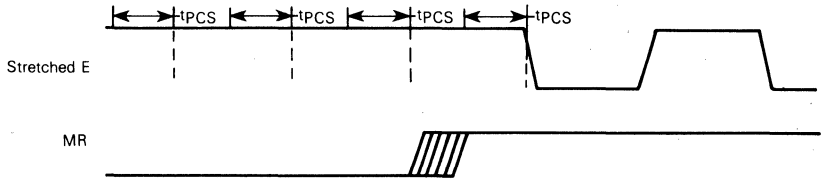
FIGURE 15 — MR NEGATIVE SETUP TIME REQUIREMENT

E Clock Stretch



The E clock will be stretched at end of E high of the cycle during which MR negative meets the t_{PCS} setup time. The t_{PCS} setup time is referenced to the fall of E. If the t_{PCS} setup time is not met, E will be stretched at the end of the next E-high ½ cycle. E will be stretched in integral multiples of ½ cycles.

Resuming E Clocking



The E clock will resume normal operation at the end of the ½ cycle during which MR assertion meets the t_{PCS} setup time. The t_{PCS} setup time is referenced to transitions of E were it not stretched. If t_{PCS} setup time is not met, E will fall at the second possible transition time after MR is asserted. There is no direct means of determining when the t_{PCS} references occur, unless the synchronizing circuit of Figure 14 is used.

RAM ENABLE (RE)

A TTL-compatible RAM enable input controls the on-chip RAM of the MC6802. When placed in the high state, the on-chip memory is enabled to respond to the MPU controls. In the low state, RAM is disabled. This pin may also be utilized to disable reading and writing the on-chip RAM during a powerdown situation. RAM Enable must be low three cycles before V_{CC} goes below 4.75 V during powerdown. RE should be tied to the correct high or low state if not used.

EXTAL AND XTAL

These inputs are used for the internal oscillator that may be crystal controlled. These connections are for a parallel resonant fundamental crystal (see Figure 12). (AT-cut.) A divide-by-four circuit has been added so a 4 MHz crystal may be used in lieu of a 1 MHz crystal for a more cost-effective system. An example of the crystal circuit layout is shown in Figure 13. Pin 39 may be driven externally by a TTL input signal four times the required E clock frequency. Pin 38 is to be grounded.

An RC network is not directly usable as a frequency source on pins 38 and 39. An RC network type TTL or CMOS oscillator will work well as long as the TTL or CMOS output drives the on-chip oscillator.

LC networks are not recommended to be used in place of the crystal.

If an external clock is used, it may not be halted for more than $t_{PW\phi L}$. The MC6802 is a dynamic part except for the internal RAM, and requires the external clock to retain information.

MEMORY READY (MR)

MR is a TTL-compatible input signal controlling the stretching of E. Use of MR requires synchronization with the $4xf_O$ signal, as shown in Figure 14. When MR is high, E will be in normal operation. When MR is low, E will be stretched integral numbers of half periods, thus allowing interface to slow memories. Memory Ready timing is shown in Figure 15.

MR should be tied high (connected directly to V_{CC}) if not used. This is necessary to ensure proper operation of the part. A maximum stretch is t_{cyc}.

ENABLE (E)

This pin supplies the clock for the MPU and the rest of the system. This is a single-phase, TTL-compatible clock. This clock may be conditioned by a memory read signal. This is equivalent to $\phi 2$ on the MC6800. This output is capable of driving one standard TTL load and 130 pF.

V_{CC} STANDBY

This pin supplies the dc voltage to the first 32 bytes of RAM as well as the RAM Enable (RE) control logic. Thus, retention of data in this portion of the RAM on a power-up, power-down, or standby condition is guaranteed. Maximum current drain at V_{SB} maximum is I_{SBB}.

MPU INSTRUCTION SET

The instruction set has 72 different instructions. Included are binary and decimal arithmetic, logical, shift, rotate, load, store, conditional or unconditional branch, interrupt and stack manipulation instructions (Tables 2 through 6). The instruction set is the same as that for the MC6800.

MPU ADDRESSING MODES

There are seven address modes that can be used by a programmer, with the addressing mode a function of both the type of instruction and the coding within the instruction. A summary of the addressing modes for a particular instruction can be found in Table 7 along with the associated instruction execution time that is given in machine cycles. With a bus frequency of 1 MHz, these times would be microseconds.

ACCUMULATOR (ACCX) ADDRESSING

In accumulator only addressing, either accumulator A or accumulator B is specified. These are one-byte instructions.

IMMEDIATE ADDRESSING

In immediate addressing, the operand is contained in the second byte of the instruction except LDS and LDX which have the operand in the second and third bytes of the instruction. The MPU addresses this location when it fetches the immediate instruction for execution. These are two- or three-byte instructions.

DIRECT ADDRESSING

In direct addressing, the address of the operand is contained in the second byte of the instruction. Direct addressing allows the user to directly address the lowest 256 bytes in the machine, i.e., locations zero through 255. Enhanced execution times are achieved by storing data in these locations. In most configurations, it should be a random-access memory. These are two-byte instructions.

EXTENDED ADDRESSING

In extended addressing, the address contained in the second byte of the instruction is used as the higher eight bits of the address of the operand. The third byte of the instruction is used as the lower eight bits of the address for the operand. This is an absolute address in memory. These are three-byte instructions.

INDEXED ADDRESSING

In indexed addressing, the address contained in the second byte of the instruction is added to the index register's lowest eight bits in the MPU. The carry is then added to the higher order eight bits of the index register. This result is then used to address memory. The modified address is held in a temporary address register so there is no change to the index register. These are two-byte instructions.

IMPLIED ADDRESSING

In the implied addressing mode, the instruction gives the address (i.e., stack pointer, index register, etc.). These are one-byte instructions.

RELATIVE ADDRESSING

In relative addressing, the address contained in the second byte of the instruction is added to the program counter's lowest eight bits plus two. The carry or borrow is then added to the high eight bits. This allows the user to address data within a range of −125 to +129 bytes of the present instruction. These are two-byte instructions.

TABLE 2 — MICROPROCESSOR INSTRUCTION SET — ALPHABETIC SEQUENCE

ABA	Add Accumulators
ADC	Add with Carry
ADD	Add
AND	Logical And
ASL	Arithmetic Shift Left
ASR	Arithmetic Shift Right
BCC	Branch if Carry Clear
BCS	Branch if Carry Set
BEQ	Branch if Equal to Zero
BGE	Branch if Greater or Equal Zero
BGT	Branch if Greater than Zero
BHI	Branch if Higher
BIT	Bit Test
BLE	Branch if Less or Equal
BLS	Branch if Lower or Same
BLT	Branch if Less than Zero
BMI	Branch if Minus
BNE	Branch if Not Equal to Zero
BPL	Branch if Plus
BRA	Branch Always
BSR	Branch to Subroutine
BVC	Branch if Overflow Clear
BVS	Branch if Overflow Set
CBA	Compare Accumulators
CLC	Clear Carry
CLI	Clear Interrupt Mask
CLR	Clear
CLV	Clear Overflow
CMP	Compare
COM	Complement
CPX	Compare Index Register
DAA	Decimal Adjust
DEC	Decrement
DES	Decrement Stack Pointer
DEX	Decrement Index Register
EOR	Exclusive OR
INC	Increment
INS	Increment Stack Pointer
INX	Increment Index Register
JMP	Jump
JSR	Jump to Subroutine
LDA	Load Accumulator
LDS	Load Stack Pointer
LDX	Load Index Register
LSR	Logical Shift Right
NEG	Negate
NOP	No Operation
ORA	Inclusive OR Accumulator
PSH	Push Data
PUL	Pull Data
ROL	Rotate Left
ROR	Rotate Right
RTI	Return from Interrupt
RTS	Return from Subroutine
SBA	Subtract Accumulators
SBC	Subtract with Carry
SEC	Set Carry
SEI	Set Interrupt Mask
SEV	Set Overflow
STA	Store Accumulator
STS	Store Stack Register
STX	Store Index Register
SUB	Subtract
SWI	Software Interrupt
TAB	Transfer Accumulators
TAP	Transfer Accumulators to Condition Code Reg.
TBA	Transfer Accumulators
TPA	Transfer Condition Code Reg. to Accumulator
TST	Test
TSX	Transfer Stack Pointer to Index Register
TXS	Transfer Index Register to Stack Pointer
WAI	Wait for Interrupt

TABLE 3 — ACCUMULATOR AND MEMORY INSTRUCTIONS

OPERATIONS	MNEMONIC	IMMED OP	IMMED ~	IMMED =	DIRECT OP	DIRECT ~	DIRECT =	INDEX OP	INDEX ~	INDEX =	EXTND OP	EXTND ~	EXTND =	IMPLIED OP	IMPLIED ~	IMPLIED =	BOOLEAN/ARITHMETIC OPERATION (All register labels refer to contents)	5 H	4 I	3 N	2 Z	1 V	0 C
Add	ADDA	3B	2	2	9B	3	2	AB	5	2	BB	4	3				A + M → A	↕	•	↕	↕	↕	↕
	ADDB	CB	2	2	DB	3	2	EB	5	2	FB	4	3				B + M → B	↕	•	↕	↕	↕	↕
Add Acmltrs	ABA													1B	2	1	A + B → A	↕	•	↕	↕	↕	↕
Add with Carry	ADCA	89	2	2	99	3	2	A9	5	2	B9	4	3				A + M + C → A	↕	•	↕	↕	↕	↕
	ADCB	C9	2	2	D9	3	2	E9	5	2	F9	4	3				B + M + C → B	↕	•	↕	↕	↕	↕
And	ANDA	84	2	2	94	3	2	A4	5	2	B4	4	3				A · M → A	•	•	↕	↕	R	•
	ANDB	C4	2	2	D4	3	2	E4	5	2	F4	4	3				B · M → B	•	•	↕	↕	R	•
Bit Test	BITA	85	2	2	95	3	2	A5	5	2	B5	4	3				A · M	•	•	↕	↕	R	•
	BITB	C5	2	2	D5	3	2	E5	5	2	F5	4	3				B · M	•	•	↕	↕	R	•
Clear	CLR							6F	7	2	7F	6	3				00 → M	•	•	R	S	R	R
	CLRA													4F	2	1	00 → A	•	•	R	S	R	R
	CLRB													5F	2	1	00 → B	•	•	R	S	R	R
Compare	CMPA	81	2	2	91	3	2	A1	5	2	B1	4	3				A - M	•	•	↕	↕	↕	↕
	CMPB	C1	2	2	D1	3	2	E1	5	2	F1	4	3				B - M	•	•	↕	↕	↕	↕
Compare Acmltrs	CBA													11	2	1	A - B	•	•	↕	↕	↕	↕
Complement, 1's	COM							63	7	2	73	6	3				M̄ → M	•	•	↕	↕	R	S
	COMA													43	2	1	Ā → A	•	•	↕	↕	R	S
	COMB													53	2	1	B̄ → B	•	•	↕	↕	R	S
Complement, 2's (Negate)	NEG							60	7	2	70	6	3				00 - M → M	•	•	↕	↕	①	②
	NEGA													40	2	1	00 - A → A	•	•	↕	↕	①	②
	NEGB													50	2	1	00 - B → B	•	•	↕	↕	①	②
Decimal Adjust, A	DAA													19	2	1	Converts Binary Add. of BCD Characters into BCD Format	•	•	↕	↕	↕	④
Decrement	DEC							6A	7	2	7A	6	3				M - 1 → M	•	•	↕	↕	④	•
	DECA													4A	2	1	A - 1 → A	•	•	↕	↕	④	•
	DECB													5A	2	1	B - 1 → B	•	•	↕	↕	④	•
Exclusive OR	EORA	88	2	2	98	3	2	A8	5	2	B8	4	3				A⊕M → A	•	•	↕	↕	R	•
	EORB	C8	2	2	D8	3	2	E8	5	2	F8	4	3				B⊕M → B	•	•	↕	↕	R	•
Increment	INC							6C	7	2	7C	6	3				M + 1 → M	•	•	↕	↕	⑤	•
	INCA													4C	2	1	A + 1 → A	•	•	↕	↕	⑤	•
	INCB													5C	2	1	B + 1 → B	•	•	↕	↕	⑤	•
Load Acmltr	LDAA	86	2	2	96	3	2	A6	5	2	B6	4	3				M → A	•	•	↕	↕	R	•
	LDAB	C6	2	2	D6	3	2	E6	5	2	F6	4	3				M → B	•	•	↕	↕	R	•
Or, Inclusive	ORAA	8A	2	2	9A	3	2	AA	5	2	BA	4	3				A + M → A	•	•	↕	↕	R	•
	ORAB	CA	2	2	DA	3	2	EA	5	2	FA	4	3				B + M → B	•	•	↕	↕	R	•
Push Data	PSHA													36	4	1	A → M_{SP}, SP - 1 → SP	•	•	•	•	•	•
	PSHB													37	4	1	B → M_{SP}, SP - 1 → SP	•	•	•	•	•	•
Pull Data	PULA													32	4	1	SP + 1 → SP, M_{SP} → A	•	•	•	•	•	•
	PULB													33	4	1	SP + 1 → SP, M_{SP} → B	•	•	•	•	•	•
Rotate Left	ROL							69	7	2	79	6	3				M	•	•	↕	↕	⑥	↕
	ROLA													49	2	1	A (C b7 b0)	•	•	↕	↕	⑥	↕
	ROLB													59	2	1	B (C b7 b0)	•	•	↕	↕	⑥	↕
Rotate Right	ROR							66	7	2	76	6	3				M	•	•	↕	↕	⑥	↕
	RORA													46	2	1	A (C b7 b0)	•	•	↕	↕	⑥	↕
	RORB													56	2	1	B (C b7 b0)	•	•	↕	↕	⑥	↕
Shift Left, Arithmetic	ASL							68	7	2	78	6	3				M	•	•	↕	↕	⑥	↕
	ASLA													48	2	1	A (b7 b0 → 0)	•	•	↕	↕	⑥	↕
	ASLB													58	2	1	B (b7 b0 → 0)	•	•	↕	↕	⑥	↕
Shift Right, Arithmetic	ASR							67	7	2	77	6	3				M	•	•	↕	↕	⑥	↕
	ASRA													47	2	1	A (b7 b0 → C)	•	•	↕	↕	⑥	↕
	ASRB													57	2	1	B (b7 b0 → C)	•	•	↕	↕	⑥	↕
Shift Right, Logic	LSR							64	7	2	74	6	3				M	•	•	R	↕	⑥	↕
	LSRA													44	2	1	A (0 → b7 b0 C)	•	•	R	↕	⑥	↕
	LSRB													54	2	1	B (0 → b7 b0 C)	•	•	R	↕	⑥	↕
Store Acmltr	STAA				97	4	2	A7	6	2	B7	5	3				A → M	•	•	↕	↕	R	•
	STAB				D7	4	2	E7	6	2	F7	5	3				B → M	•	•	↕	↕	R	•
Subtract	SUBA	80	2	2	90	3	2	A0	5	2	B0	4	3				A - M → A	•	•	↕	↕	↕	↕
	SUBB	C0	2	2	D0	3	2	E0	5	2	F0	4	3				B - M → B	•	•	↕	↕	↕	↕
Subtract Acmltrs.	SBA													10	2	1	A - B → A	•	•	↕	↕	↕	↕
Subtr. with Carry	SBCA	82	2	2	92	3	2	A2	5	2	B2	4	3				A - M - C → A	•	•	↕	↕	↕	↕
	SBCB	C2	2	2	D2	3	2	E2	5	2	F2	4	3				B - M - C → B	•	•	↕	↕	↕	↕
Transfer Acmltrs	TAB													16	2	1	A → B	•	•	↕	↕	R	•
	TBA													17	2	1	B → A	•	•	↕	↕	R	•
Test, Zero or Minus	TST							6D	7	2	7D	6	3				M - 00	•	•	↕	↕	R	R
	TSTA													4D	2	1	A - 00	•	•	↕	↕	R	R
	TSTB													5D	2	1	B - 00	•	•	↕	↕	R	R
																		H	I	N	Z	V	C

LEGEND:

OP	Operation Code (Hexadecimal);	+	Boolean Inclusive OR;
~	Number of MPU Cycles;	⊙	Boolean Exclusive OR;
=	Number of Program Bytes;	M̄	Complement of M;
+	Arithmetic Plus;	→	Transfer Into;
-	Arithmetic Minus;	0	Bit = Zero;
·	Boolean AND;	00	Byte = Zero;
M_{SP}	Contents of memory location pointed to be Stack Pointer;		

Note – Accumulator addressing mode instructions are included in the column for IMPLIED addressing

CONDITION CODE SYMBOLS:

H	Half-carry from bit 3;
I	Interrupt mask
N	Negative (sign bit)
Z	Zero (byte)
V	Overflow, 2's complement
C	Carry from bit 7
R	Reset Always
S	Set Always
↕	Test and set if true, cleared otherwise
•	Not Affected

3

TABLE 4 — INDEX REGISTER AND STACK MANIPULATION INSTRUCTIONS

POINTER OPERATIONS	MNEMONIC	IMMED OP	IMMED ~	IMMED =	DIRECT OP	DIRECT ~	DIRECT =	INDEX OP	INDEX ~	INDEX =	EXTND OP	EXTND ~	EXTND =	IMPLIED OP	IMPLIED ~	IMPLIED =	BOOLEAN/ARITHMETIC OPERATION	COND. CODE REG. 5 H	4 I	3 N	2 Z	1 V	0 C
Compare Index Reg	CPX	8C	3	3	9C	4	2	AC	6	2	BC	5	3				$X_H - M, X_L - (M+1)$	•	•	⑦	↕	⑧	•
Decrement Index Reg	DEX													09	4	1	$X - 1 \rightarrow X$	•	•	•	↕	•	•
Decrement Stack Pntr	DES													34	4	1	$SP - 1 \rightarrow SP$	•	•	•	•	•	•
Increment Index Reg	INX													08	4	1	$X + 1 \rightarrow X$	•	•	•	↕	•	•
Increment Stack Pntr	INS													31	4	1	$SP + 1 \rightarrow SP$	•	•	•	•	•	•
Load Index Reg	LDX	CE	3	3	DE	4	2	EE	6	2	FE	5	3				$M \rightarrow X_H, (M+1) \rightarrow X_L$	•	•	⑨	↕	R	•
Load Stack Pntr	LDS	8E	3	3	9E	4	2	AE	6	2	BE	5	3				$M \rightarrow SP_H, (M+1) \rightarrow SP_L$	•	•	⑨	↕	R	•
Store Index Reg	STX				DF	5	2	EF	7	2	FF	6	3				$X_H \rightarrow M, X_L \rightarrow (M+1)$	•	•	⑨	↕	R	•
Store Stack Pntr	STS				9F	5	2	AF	7	2	BF	6	3				$SP_H \rightarrow M, SP_L \rightarrow (M+1)$	•	•	⑨	↕	R	•
Indx Reg → Stack Pntr	TXS													35	4	1	$X - 1 \rightarrow SP$	•	•	•	•	•	•
Stack Pntr → Indx Reg	TSX													30	4	1	$SP + 1 \rightarrow X$	•	•	•	•	•	•

3

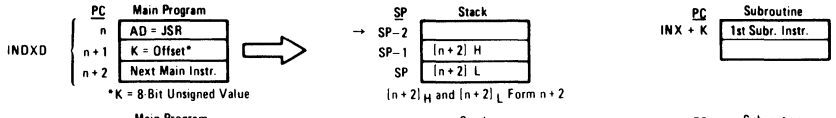
TABLE 5 — JUMP AND BRANCH INSTRUCTIONS

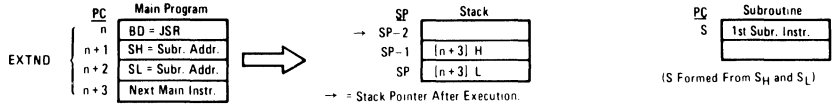
OPERATIONS	MNEMONIC	RELATIVE OP	RELATIVE ~	RELATIVE =	INDEX OP	INDEX ~	INDEX =	EXTND OP	EXTND ~	EXTND =	IMPLIED OP	IMPLIED ~	IMPLIED =	BRANCH TEST	COND. CODE REG. 5 H	4 I	3 N	2 Z	1 V	0 C
Branch Always	BRA	20	4	2										None	•	•	•	•	•	•
Branch If Carry Clear	BCC	24	4	2										C = 0	•	•	•	•	•	•
Branch If Carry Set	BCS	25	4	2										C = 1	•	•	•	•	•	•
Branch If = Zero	BEQ	27	4	2										Z = 1	•	•	•	•	•	•
Branch If ≥ Zero	BGE	2C	4	2										$N \oplus V = 0$	•	•	•	•	•	•
Branch If > Zero	BGT	2E	4	2										$Z + (N \oplus V) = 0$	•	•	•	•	•	•
Branch If Higher	BHI	22	4	2										C + Z = 0	•	•	•	•	•	•
Branch If ≤ Zero	BLE	2F	4	2										$Z + (N \oplus V) = 1$	•	•	•	•	•	•
Branch If Lower Or Same	BLS	23	4	2										C + Z = 1	•	•	•	•	•	•
Branch If < Zero	BLT	2D	4	2										$N \oplus V = 1$	•	•	•	•	•	•
Branch If Minus	BMI	2B	4	2										N = 1	•	•	•	•	•	•
Branch If Not Equal Zero	BNE	26	4	2										Z = 0	•	•	•	•	•	•
Branch If Overflow Clear	BVC	28	4	2										V = 0	•	•	•	•	•	•
Branch If Overflow Set	BVS	29	4	2										V = 1	•	•	•	•	•	•
Branch If Plus	BPL	2A	4	2										N = 0	•	•	•	•	•	•
Branch To Subroutine	BSR	8D	8	2											•	•	•	•	•	•
Jump	JMP				6E	4	2	7E	3	3				See Special Operations	•	•	•	•	•	•
Jump To Subroutine	JSR				AD	8	2	BD	9	3				(Figure 16)	•	•	•	•	•	•
No Operation	NOP										01	2	1	Advances Prog. Cntr. Only	•	•	•	•	•	•
Return From Interrupt	RTI										3B	10	1		⑩					
Return From Subroutine	RTS										39	5	1		•	•	•	•	•	•
Software Interrupt	SWI										3F	12	1	See Special Operations	•	•	•	•	•	•
Wait for Interrupt	WAI										3E	9	1	(Figure 16)	•	⑪	•	•	•	•

FIGURE 16 — SPECIAL OPERATIONS

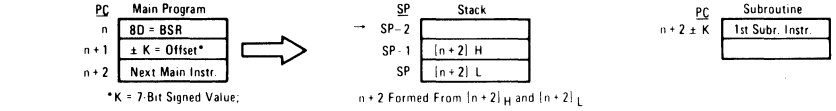
SPECIAL OPERATIONS

JSR, JUMP TO SUBROUTINE:



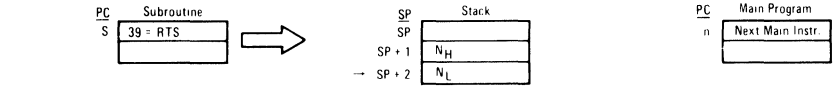


BSR, BRANCH TO SUBROUTINE:



JMP, JUMP:

RTS, RETURN FROM SUBROUTINE:



RTI, RETURN FROM INTERRUPT:

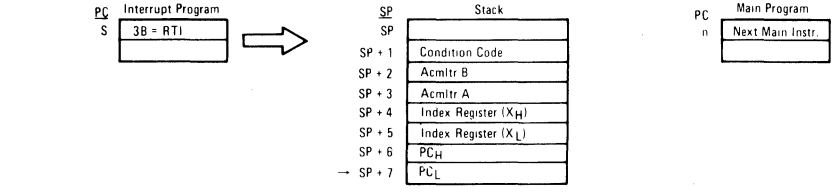


TABLE 6 — CONDITION CODE REGISTER MANIPULATION INSTRUCTIONS

OPERATIONS	MNEMONIC	IMPLIED OP	~	=	BOOLEAN OPERATION	COND. CODE REG. 5 H	4 I	3 N	2 Z	1 V	0 C
Clear Carry	CLC	0C	2	1	0 → C	•	•	•	•	•	R
Clear Interrupt Mask	CLI	0E	2	1	0 → I	•	R	•	•	•	•
Clear Overflow	CLV	0A	2	1	0 → V	•	•	•	•	R	•
Set Carry	SEC	0D	2	1	1 → C	•	•	•	•	•	S
Set Interrupt Mask	SEI	0F	2	1	1 → I	•	S	•	•	•	•
Set Overflow	SEV	0B	2	1	1 → V	•	•	•	•	S	•
Acmltr A → CCR	TAP	06	2	1	A → CCR	⑫					
CCR → Acmltr A	TPA	07	2	1	CCR → A	•	•	•	•	•	•

CONDITION CODE REGISTER NOTES: (Bit set if test is true and cleared otherwise)

1 (Bit V) Test: Result = 10000000?
2 (Bit C) Test: Result ≠ 00000000?
3 (Bit C) Test: Decimal value of most significant BCD Character greater than nine? (Not cleared if previously set.)
4 (Bit V) Test: Operand = 10000000 prior to execution?
5 (Bit V) Test: Operand = 01111111 prior to execution?
6 (Bit V) Test: Set equal to result of N⊕C after shift has occurred.
7 (Bit N) Test: Sign bit of most significant (MS) byte = 1?
8 (Bit V) Test: 2's complement overflow from subtraction of MS bytes?
9 (Bit N) Test: Result less than zero? (Bit 15 = 1)
10 (All) Load Condition Code Register from Stack. (See Special Operations)
11 (Bit I) Set when interrupt occurs. If previously set, a Non-Maskable Interrupt is required to exit the wait state.
12 (All) Set according to the contents of Accumulator A.

TABLE 7 — INSTRUCTION ADDRESSING MODES AND ASSOCIATED EXECUTION TIMES
(Times in Machine Cycle)

	(Dual Operand)	ACCX	Immediate	Direct	Extended	Indexed	Implied	Relative
ABA		•	•	•	•	•	2	•
ADC	x	•	2	3	4	5	•	•
ADD	x	•	2	3	4	5	•	•
AND	x	•	2	3	4	5	•	•
ASL		2	•	•	6	7	•	•
ASR		2	•	•	6	7	•	•
BCC		•	•	•	•	•	•	4
BCS		•	•	•	•	•	•	4
BEA		•	•	•	•	•	•	4
BGE		•	•	•	•	•	•	4
BGT		•	•	•	•	•	•	4
BHI		•	•	•	•	•	•	4
BIT	x	•	2	3	4	5	•	•
BLE		•	•	•	•	•	•	4
BLS		•	•	•	•	•	•	4
BLT		•	•	•	•	•	•	4
BMI		•	•	•	•	•	•	4
BNE		•	•	•	•	•	•	4
BPL		•	•	•	•	•	•	4
BRA		•	•	•	•	•	•	4
BSR		•	•	•	•	•	•	8
BVC		•	•	•	•	•	•	4
BVS		•	•	•	•	•	•	4
CBA		•	•	•	•	•	2	•
CLC		•	•	•	•	•	2	•
CLI		•	•	•	•	•	2	•
CLR		2	•	•	6	7	•	•
CLV		•	•	•	•	•	2	•
CMP	x	•	2	3	4	5	•	•
COM		2	•	•	6	7	•	•
CPX		•	3	4	5	6	•	•
DAA		•	•	•	•	•	2	•
DEC		2	•	•	6	7	•	•
DES		•	•	•	•	•	4	•
DEX		•	•	•	•	•	4	•
EOR	x	•	2	3	4	5	•	•

	(Dual Operand)	ACCX	Immediate	Direct	Extended	Indexed	Implied
INC		2	•	•	6	7	•
INS		•	•	•	•	•	4
INX		•	•	•	•	•	4
JMP		•	•	•	3	4	•
JSR		•	•	•	9	8	•
LDA	x	•	2	3	4	5	•
LDS		•	3	4	5	6	•
LDX		•	3	4	5	6	•
LSR		2	•	•	6	7	•
NEG		2	•	•	6	7	•
NOP		•	•	•	•	•	2
ORA	x	•	2	3	4	5	•
PSH		•	•	•	•	•	4
PUL		•	•	•	•	•	4
ROL		2	•	•	6	7	•
ROR		2	•	•	6	7	•
RTI		•	•	•	•	•	10
RTS		•	•	•	•	•	5
SBA		•	•	•	•	•	2
SBC	x	•	2	3	4	5	•
SEC		•	•	•	•	•	2
SEI		•	•	•	•	•	2
SEV		•	•	•	•	•	2
STA	x	•	•	4	5	6	•
STS		•	•	5	6	7	•
STX		•	•	5	6	7	•
SUB	x	•	2	3	4	5	•
SWI		•	•	•	•	•	12
TAB		•	•	•	•	•	2
TAP		•	•	•	•	•	2
TBA		•	•	•	•	•	2
TPA		•	•	•	•	•	2
TST		2	•	•	6	7	•
TSX		•	•	•	•	•	4
TSX		•	•	•	•	•	4
WAI		•	•	•	•	•	9

NOTE Interrupt time is 12 cycles from the end of the instruction being executed, except following a WAI instruction. Then it is 4 cycles

SUMMARY OF CYCLE-BY-CYCLE OPERATION

Table 8 provides a detailed description of the information present on the address bus, data bus, valid memory address line (VMA), and the read/write line (R/$\overline{W}$) during each cycle for each instruction.

This information is useful in comparing actual with expected results during debug of both software and hardware as the control program is executed. The information is categorized in groups according to addressing modes and number of cycles per instruction. (In general, instructions with the same addressing mode and number of cycles execute in the same manner; exceptions are indicated in the table.)

TABLE 8 — OPERATIONS SUMMARY

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
IMMEDIATE						
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	2	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Operand Data
CPX LDS LDX	3	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Operand Data (High Order Byte)
		3	1	Op Code Address + 2	1	Operand Data (Low Order Byte)
DIRECT						
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	3	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand
		3	1	Address of Operand	1	Operand Data
CPX LDS LDX	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand
		3	1	Address of Operand	1	Operand Data (High Order Byte)
		4	1	Operand Address + 1	1	Operand Data (Low Order Byte)
STA	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Destination Address
		3	0	Destination Address	1	Irrelevant Data (Note 1)
		4	1	Destination Address	0	Data from Accumulator
STS STX	5	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand
		3	0	Address of Operand	1	Irrelevant Data (Note 1)
		4	1	Address of Operand	0	Register Data (High Order Byte)
		5	1	Address of Operand + 1	0	Register Data (Low Order Byte)
INDEXED						
JMP	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	5	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
		5	1	Index Register Plus Offset	1	Operand Data
CPX LDS LDX	6	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
		5	1	Index Register Plus Offset	1	Operand Data (High Order Byte)
		6	1	Index Register Plus Offset + 1	1	Operand Data (Low Order Byte)

TABLE 8 — OPERATIONS SUMMARY (CONTINUED)

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
INDEXED (Continued)						
STA	6	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
		5	0	Index Register Plus Offset	1	Irrelevant Data (Note 1)
		6	1	Index Register Plus Offset	0	Operand Data
ASL LSR ASR NEG CLR ROL COM ROR DEC TST INC	7	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
		5	1	Index Register Plus Offset	1	Current Operand Data
		6	0	Index Register Plus Offset	1	Irrelevant Data (Note 1)
		7	1/0 (Note 3)	Index Register Plus Offset	0	New Operand Data (Note 3)
STS STX	7	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
		5	0	Index Register Plus Offset	1	Irrelevant Data (Note 1)
		6	1	Index Register Plus Offset	0	Operand Data (High Order Byte)
		7	1	Index Register Plus Offset + 1	0	Operand Data (Low Order Byte)
JSR	8	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Offset
		3	0	Index Register	1	Irrelevant Data (Note 1)
		4	1	Stack Pointer	0	Return Address (Low Order Byte)
		5	1	Stack Pointer − 1	0	Return Address (High Order Byte)
		6	0	Stack Pointer − 2	1	Irrelevant Data (Note 1)
		7	0	Index Register	1	Irrelevant Data (Note 1)
		8	0	Index Register Plus Offset (w/o Carry)	1	Irrelevant Data (Note 1)
EXTENDED						
JMP	3	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Jump Address (High Order Byte)
		3	1	Op Code Address + 2	1	Jump Address (Low Order Byte)
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	1	Address of Operand	1	Operand Data
CPX LDS LDX	5	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	1	Address of Operand	1	Operand Data (High Order Byte)
		5	1	Address of Operand + 1	1	Operand Data (Low Order Byte)
STA A STA B	5	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Destination Address (High Order Byte)
		3	1	Op Code Address + 2	1	Destination Address (Low Order Byte)
		4	0	Operand Destination Address	1	Irrelevant Data (Note 1)
		5	1	Operand Destination Address	0	Data from Accumulator
ASL LSR ASR NEG CLR ROL COM ROR DEC TST INC	6	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	1	Address of Operand	1	Current Operand Data
		5	0	Address of Operand	1	Irrelevant Data (Note 1)
		6	1/0 (Note 3)	Address of Operand	0	New Operand Data (Note 3)

TABLE 8 — OPERATIONS SUMMARY (CONTINUED)

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
EXTENDED (Continued)						
STS STX	6	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Operand (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Operand (Low Order Byte)
		4	0	Address of Operand	1	Irrelevant Data (Note 1)
		5	1	Address of Operand	0	Operand Data (High Order Byte)
		6	1	Address of Operand + 1	0	Operand Data (Low Order Byte)
JSR	9	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Address of Subroutine (High Order Byte)
		3	1	Op Code Address + 2	1	Address of Subroutine (Low Order Byte)
		4	1	Subroutine Starting Address	1	Op Code of Next Instruction
		5	1	Stack Pointer	0	Return Address (Low Order Byte)
		6	1	Stack Pointer − 1	0	Return Address (High Order Byte)
		7	0	Stack Pointer − 2	1	Irrelevant Data (Note 1)
		8	0	Op Code Address + 2	1	Irrelevant Data (Note 1)
		9	1	Op Code Address + 2	1	Address of Subroutine (Low Order Byte)
INHERENT						
ABA DAA SEC ASL DEC SEI ASR INC SEV CBA LSR TAB CLC NEG TAP CLI NOP TBA CLR ROL TPA CLV ROR TST COM SBA	2	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
DES DEX INS INX	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
		3	0	Previous Register Contents	1	Irrelevant Data (Note 1)
		4	0	New Register Contents	1	Irrelevant Data (Note 1)
PSH	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
		3	1	Stack Pointer	0	Accumulator Data
		4	0	Stack Pointer − 1	1	Accumulator Data
PUL	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
		3	0	Stack Pointer	1	Irrelevant Data (Note 1)
		4	1	Stack Pointer + 1	1	Operand Data from Stack
TSX	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
		3	0	Stack Pointer	1	Irrelevant Data (Note 1)
		4	0	New Index Register	1	Irrelevant Data (Note 1)
TXS	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
		3	0	Index Register	1	Irrelevant Data
		4	0	New Stack Pointer	1	Irrelevant Data
RTS	5	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Irrelevant Data (Note 2)
		3	0	Stack Pointer	1	Irrelevant Data (Note 1)
		4	1	Stack Pointer + 1	1	Address of Next Instruction (High Order Byte)
		5	1	Stack Pointer + 2	1	Address of Next Instruction (Low Order Byte)

TABLE 8 — OPERATIONS SUMMARY (CONCLUDED)

Address Mode and Instructions	Cycles	Cycle #	VMA Line	Address Bus	R/$\overline{W}$ Line	Data Bus
INHERENT (Continued)						
WAI	9	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Op Code of Next Instruction
		3	1	Stack Pointer	0	Return Address (Low Order Byte)
		4	1	Stack Pointer – 1	0	Return Address (High Order Byte)
		5	1	Stack Pointer – 2	0	Index Register (Low Order Byte)
		6	1	Stack Pointer – 3	0	Index Register (High Order Byte)
		7	1	Stack Pointer – 4	0	Contents of Accumulator A
		8	1	Stack Pointer – 5	0	Contents of Accumulator B
		9	1	Stack Pointer – 6	1	Contents of Cond. Code Register
RTI	10	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Irrelevant Data (Note 2)
		3	0	Stack Pointer	1	Irrelevant Data (Note 1)
		4	1	Stack Pointer + 1	1	Contents of Cond. Code Register from Stack
		5	1	Stack Pointer + 2	1	Contents of Accumulator B from Stack
		6	1	Stack Pointer + 3	1	Contents of Accumulator A from Stack
		7	1	Stack Pointer + 4	1	Index Register from Stack (High Order Byte)
		8	1	Stack Pointer + 5	1	Index Register from Stack (Low Order Byte)
		9	1	Stack Pointer + 6	1	Next Instruction Address from Stack (High Order Byte)
		10	1	Stack Pointer + 7	1	Next Instruction Address from Stack (Low Order Byte)
SWI	12	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Irrelevant Data (Note 1)
		3	1	Stack Pointer	0	Return Address (Low Order Byte)
		4	1	Stack Pointer – 1	0	Return Address (High Order Byte)
		5	1	Stack Pointer – 2	0	Index Register (Low Order Byte)
		6	1	Stack Pointer – 3	0	Index Register (High Order Byte)
		7	1	Stack Pointer – 4	0	Contents of Accumulator A
		8	1	Stack Pointer – 5	0	Contents of Accumulator B
		9	1	Stack Pointer – 6	0	Contents of Cond. Code Register
		10	0	Stack Pointer – 7	1	Irrelevant Data (Note 1)
		11	1	Vector Address FFFA (Hex)	1	Address of Subroutine (High Order Byte)
		12	1	Vector Address FFFB (Hex)	1	Address of Subroutine (Low Order Byte)
RELATIVE						
BCC BHI BNE BCS BLE BPL BEQ BLS BRA BGE BLT BVC BGT BMI BVS	4	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Branch Offset
		3	0	Op Code Address + 2	1	Irrelevant Data (Note 1)
		4	0	Branch Address	1	Irrelevant Data (Note 1)
BSR	8	1	1	Op Code Address	1	Op Code
		2	1	Op Code Address + 1	1	Branch Offset
		3	0	Return Address of Main Program	1	Irrelevant Data (Note 1)
		4	1	Stack Pointer	0	Return Address (Low Order Byte)
		5	1	Stack Pointer – 1	0	Return Address (High Order Byte)
		6	0	Stack Pointer – 2	1	Irrelevant Data (Note 1)
		7	0	Return Address of Main Program	1	Irrelevant Data (Note 1)
		8	0	Subroutine Address (Note 4)	1	Irrelevant Data (Note 1)

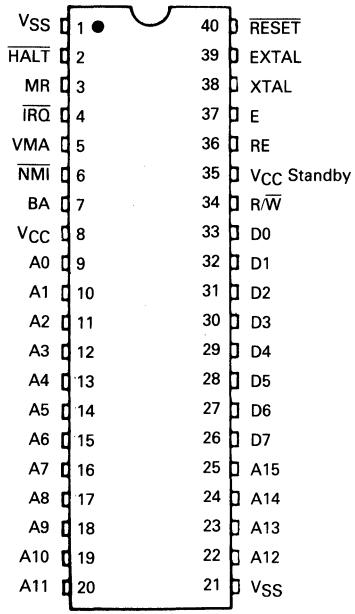
NOTES:
1. If device which is addressed during this cycle uses VMA, then the Data Bus will go to the high-impedance three-state condition. Depending on bus capacitance, data from the previous cycle may be retained on the Data Bus.
2. Data is ignored by the MPU.
3. For TST, VMA = 0 and Operand data does not change.
4. MS Byte of Address Bus = MS Byte of Address of BSR instruction and LS Byte of Address Bus = LS Byte of Sub-Routine Address.

MECHANICAL DATA AND ORDERING INFORMATION

ORDERING INFORMATION

Package Type	Frequency MHz	Temperature	Order Number
Plastic	1.0	0°C to 70°C	MC6802P
P Suffix	1.0	−40°C to +85°C	MC6802CP
	1.5	0°C to 70°C	MC68A02P
	1.5	−40°C to +85°C	MC68A02CP
	2.0	0°C to 70°C	MC68B02P
Cerdip	1.0	0°C to 70°C	MC6802S
S Suffix	1.0	−40°C to +85°C	MC6802CS
	1.5	0°C to 70°C	MC68A02S
	1.5	−40°C to +85°C	MC68A02CS
	2.0	0°C to 70°C	MC68B02S

PIN ASSIGNMENT



MC6804J1

Technical Summary
8-Bit Microcomputer Unit

MC6804J1 HMOS (high-density NMOS) microcomputer unit (MCU) is a member of the M6804 Family of serial processing microcomputers. This device displays all the versatility of an MCU whose design-ability to process 8-bit variables one bit at a time already makes it tremendously cost effective.

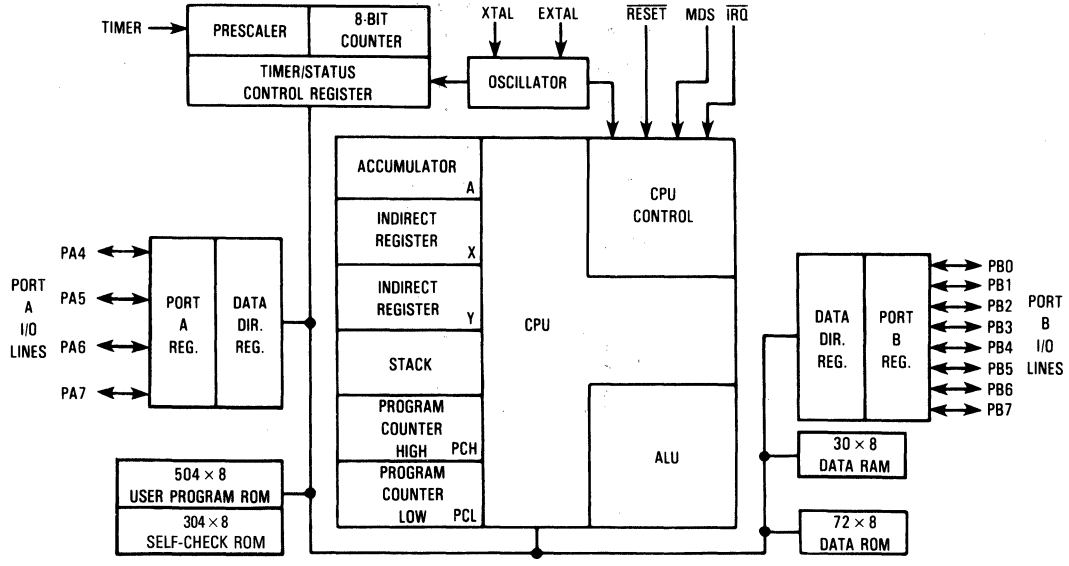
This technical summary contains limited information on the MC6804J1. For detailed information, refer to the advanced information data sheet for the MC6804J1, MC6804J2, MC6804P2, and MC68704P2 8-bit microcomputers (MC6804J1/D) or to the *M6804 MCU Manual* (DLE404/D).

Major hardware and software features of the MC6804P2 MCU are:

- On-Chip Clock Generator
- Memory Mapped I/O
- Software Programmable 8-Bit Timer with 7-Bit Prescaler
- Single Instruction Memory Examine/ Change
- 30 Bytes of Data RAM
- True Bit Manipulation
- Bit Test and Branch Instruction
- 304 Bytes Self-Check ROM
- Conditional Branches
- Timer Pin is Software Programmable as Clock Input or Timer Output
- 504 Bytes of User Program Space ROM
- User Selectable Constant Current Pullup Devices available on LSTTL and Open-Drain Interface Ports
- Mask Selectable Edge- or Level-Sensitive Interrupt Pin

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

V_{CC} AND V_{SS}

Power is supplied to the microcomputer using these two pins. V_{CC} is +5 volts (±0.5 V) power, and V_{SS} is ground.

$\overline{IRQ}$

This pin provides the capability for asynchronously applying an external interrupt to the microcomputer.

EXTAL AND XTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a resistor/capacitor combination, or an external signal is connected to these pins to provide a system clock. Selection is made by a manufacturing mask option. The different clock generator options are shown in Figure 1, along with crystal specifications.

Internal Clock Options

The crystal oscillator start-up time is a function of many variables. To ensure rapid oscillator start-up, neither the crystal characteristics nor load capacitances should exceed recommendations. When using the on-board oscillator, the MCU should remain in a reset condition, with the $\overline{RESET}$ pin voltage below V_{IRES+}, until the oscillator has stabilized at its operating frequency. See Figure 2 for resistor/capacitor oscillator options.

TIMER

The TIMER pin can be configured to operate in either the input or output mode. As input, this pin is connected to the prescaler input and serves as the timer clock. As output, the timer pin reflects the contents of the DOUT bit of the timer status/control register, the last time the TMZ bit was logic high.

$\overline{RESET}$

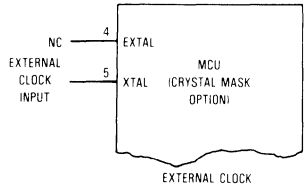
The $\overline{RESET}$ pin is used to restart the processor to the beginning of a program. The program counter is loaded with the address of the restart vector. This should be a jump instruction to the first instruction of the main program. Together with the MDS pin, the $\overline{RESET}$ pin selects the operating mode of the MCU.

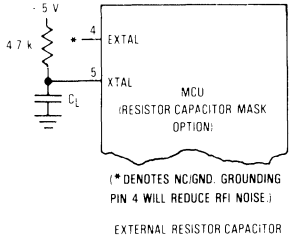
MDS

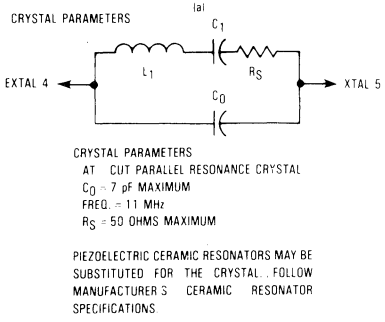
The mode select (MDS) pin places the MCU into special operating modes. When this pin is logic high at the exit of the reset state, the decoded state of PA6 and PA7 is latched to determine the operating mode. This choice can be either the single-chip, self-check, or EPROM programming. However, if MDS is logic low at the end of the reset state, the single-chip operating mode is automatically selected. No external diodes, switches, transistors, etc. are required for single-chip mode selection.

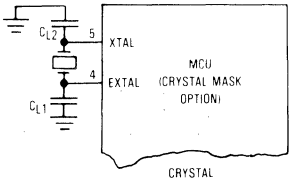
INPUT/OUTPUT LINES (PA4-PA7, PB0-PB7)

These 12 lines are arranged into one 4-bit port (A) and one 8-bit port (B). All lines are programmable as either



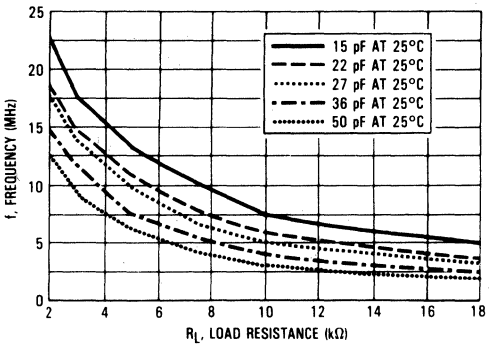




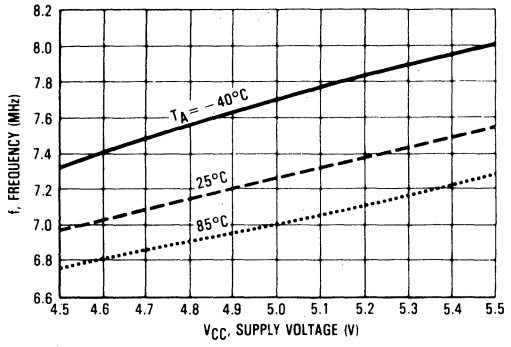


NOTE Keep crystal leads and circuit connections as short as possible

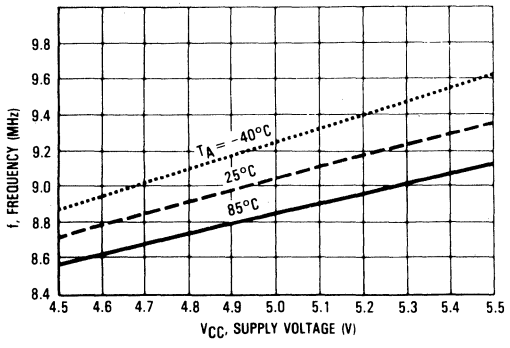
Figure 1. Clock Generator Options and Crystal Parameters



(a) TYPICAL FREQUENCY VS RESISTANCE



(b) TYPICAL FREQUENCY VARIATIONS @ C_L = 15 pF, 10 kΩ



(c) TYPICAL FREQUENCY VARIATIONS @ C_L = 50 pF, 3 kΩ

Figure 2. Typical Frequency Selection for Resistor/Capacitor Oscillator Options

inputs or outputs under software control of the data direction registers.

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

There are 12 input/output pins. All pins of each port are programmable as inputs or outputs under the control of the data direction registers (DDR).

The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic one for output, or a logic zero for input, as shown in Figure 3. When the registers are programmed as outputs, the latched data is readable regardless of the logic levels at the output pin due to output loading.

All the I/O pins are LSTTL compatible as both inputs and outputs. In addition, both ports may use either or both of two manufacturing mask options; open drain output, or internal pull-up resistor for CMOS compatibility.

Any write to a port writes to all of its data bits even though the port DDR may be set to input. This can be used as a tool to initialize the data registers and avoid undefined outputs. However, care must be exercised when using read-modify-write instructions. The data read corresponds to the pin level if the DDR is an input or to the latched output data when the DDR is an output.

The 12 bidirectional lines may be configured by port to be the standard configuration (LSTTL), or either mask option; LSTTL/CMOS, or open drain. Port B outputs are LED compatible.

Port Data Registers ($00, $01)

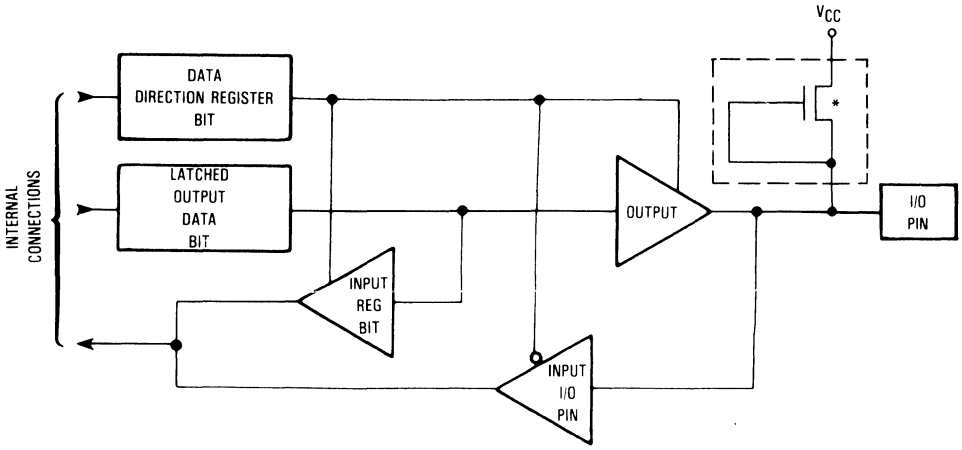
The port data registers are not initialized on reset. These registers should be initialized before changing the DDR bits to avoid undefined levels.

Port A ($00)

7	6	5	4	3	2	1	0
				X	X	X	X

Port B ($01)

7	6	5	4	3	2	1	0



DATA DIRECTION REGISTER BIT	OUTPUT DATA BIT	OUTPUT STATE	INPUT TO MCU
1	0	0	0
1	1	1	1
0	X	HI-Z	PIN

*For CMOS option transistor acts as resistor (approximately 40 kΩ) to V_{CC}.
For LSTTL/open-drain options transistor acts as low current clamping diode to V_{CC}.

Figure 3. Typical I/O Port Circuitry

With regard to Port A only, the four LSB bits are unused. These bits are "don't care" (X) bits when written to but are always logic high when read.

Port Data Direction Registers ($04, $05)

Port DDRs configure the port pins as either outputs or inputs. Each port pin can be programmed individually to be an input or an output. A zero in the pin's corresponding DDR bit programs it as an input; a logic one programs it as an output. On reset, all the DDRs are initialized to a logic zero state to put the ports in the input mode.

Port A ($04)

7	6	5	4	3	2	1	0
				0	0	0	0

Port B ($05)

7	6	5	4	3	2	1	0

With regard to Port A, the four LSB bits are cleared (logic zero) by reset. These bits must not be set (logic one).

MEMORY

The MCU memory map (Figure 4) consists of 4352 bytes of addressable memory, I/O register locations, and four levels of stack space. This MCU has three separate memory spaces: program space, data space, and stack space.

The MCU is capable of addressing 4096 bytes of program space memory with its program counter and 256 bytes of data space memory with its instructions. Program space memory includes self-check ROM, program ROM, self-check and user program vectors, and reserved memory locations.

A non-accessible subroutine stack space RAM is provided. This stack space consists of a last-in-first-out (LIFO) register. This register is used with inherent addressing to stack the return address for subroutines and interrupts.

Indirect X and Y register locations $80 and $81 are generally used as pointers for such tasks as indirect addressing to data space locations. Short direct addressing allows access to the four data space addresses $80-$83 with single byte opcodes. The operations allowed are increment, decrement, load, and store. Data space locations $82 and $83 can be used for 8-bit counter locations.

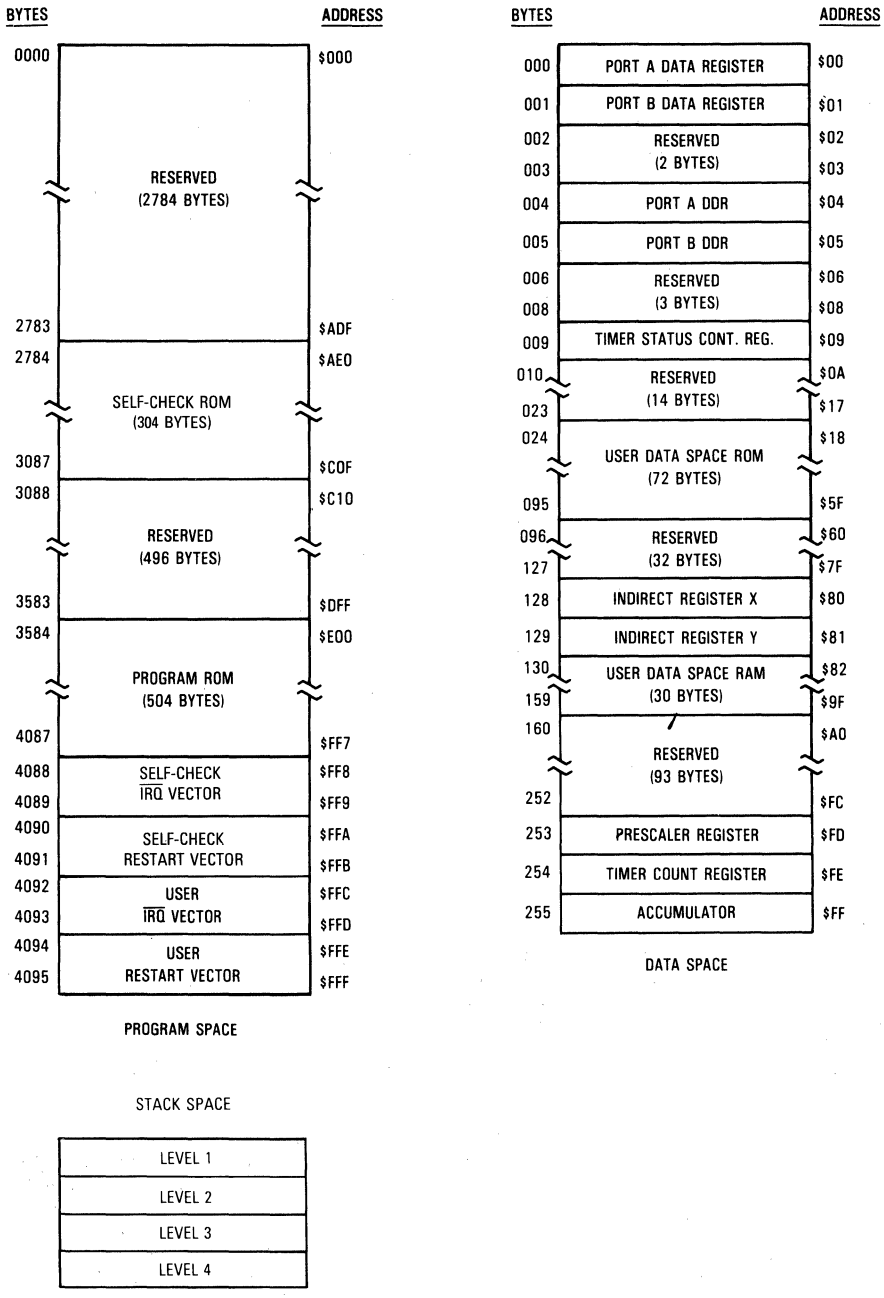
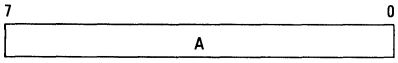


Figure 4. Memory Map

REGISTERS

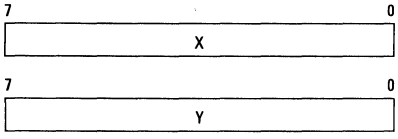
ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



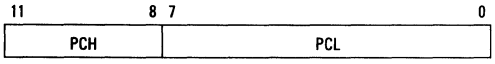
INDIRECT REGISTERS (X,Y)

These two registers are used to maintain pointers to other memory locations in data space. They are used in the register-indirect addressing mode and can be accessed with the direct, indirect, short direct, or bit set/clear modes.



PROGRAM COUNTER (PC)

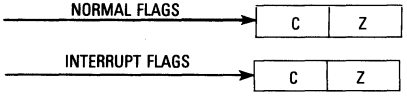
The program counter is a 12-bit register that contains the address of the next byte to be fetched. The program counter is contained in low byte (PCL) and high nibble (PCH).



FLAGS (C,Z)

The first flag, the carry (C) bit, is set on a carry or borrow out of the arithmetic logic unit (ALU). It is cleared if the arithmetic operation does not result in a carry or borrow. The C bit is also set to the value of the bit tested in a bit test instruction. It participates in the rotate left (ROLA) instruction, as well.

The second flag, the zero (Z) bit, is set if the result of the last arithmetic or logic operation was equal to zero. Otherwise, it is cleared. Bit test instructions do not affect the Z bit.



There are two sets of these flags. One set is for interrupt processing (interrupt mode flags). The other set is for normal operations (program mode flags). When an interrupt occurs, a context switch is made from the program flags to the interrupt flags. An RTI forces the context switch back. While in either mode, only the flags for that mode are available. A context switch does not affect the value of the C or Z bits. Both sets of flags are cleared by $\overline{\text{RESET}}$.

STACK

A last-in-first-out (LIFO) stack is incorporated in the MCU that eliminates the need for a stack pointer. This non-accessible subroutine stack space is implemented in separate RAM, 12 bits wide. Whenever a subroutine call or interrupt occurs, the contents of the PC are shifted into the top register of the stack. At the same time, the top register is shifted one level deeper. This happens to all registers, with the bottom register falling out of the stack.

Whenever a return from subroutine or interrupt occurs, the top register is shifted into the PC and all lower registers are shifted one level higher. The stack RAM is four levels deep. If the stack is pulled more than four times with no pushes, then the address that was stored in the bottom level of the stack is shifted into the PC.

SELF CHECK

The MCU implements two forms of internal check: self check and ROM verify. Self check performs an extensive functional check of the MCU using a signature analysis technique. ROM verify uses a similar method to check the contents of program ROM.

Self-check mode is selected by holding the MDS and PA7 pins logic high and the PA6 pin logic low as $\overline{\text{RESET}}$ goes low to high. ROM verify mode is entered by holding MDS, PA7, and PA6 logic high as $\overline{\text{RESET}}$* goes low to high. Unimplemented program space ROM locations are also tested. Monitoring the self-check mode's stages for successful completion requires external circuitry, see *M6804 MCU Manual* (DLE404/D).

RESET

$\overline{\text{RESET}}$

All resets of the MC6804J1 are caused by the external reset input ($\overline{\text{RESET}}$). A reset can be achieved by pulling the $\overline{\text{RESET}}$ pin to logic low for a minimum of 96 oscillator cycles.

During reset, a delay of 96 oscillator cycles is needed before allowing the $\overline{\text{RESET}}$ input to go high. If power is being applied, $\overline{\text{RESET}}$ must be held low long enough for the oscillator to stabilize and then provide the 96 clocks. Connecting a capacitor and resistor to the $\overline{\text{RESET}}$ input, as shown in Figure 5 below, typically provides sufficient delay.

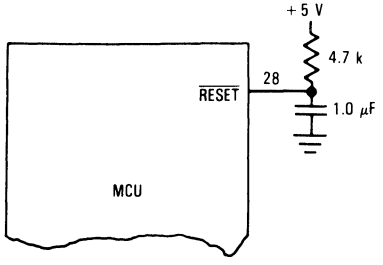


Figure 5. Powerup RESET Delay Circuit

INTERRUPT

The MCU can be interrupted by applying a logic low signal to the $\overline{IRQ}$ pin. However, a manufacturing mask option determines whether the falling edge or the actual low level of the $\overline{IRQ}$ pin is sensed to indicate an interrupt.

EDGE-SENSITIVE OPTION

When the $\overline{IRQ}$ pin is pulled low, the internal interrupt request latch is set. Prior to each instruction fetch, this interrupt request latch is tested. If its output is low, an interrupt sequence is initiated at the end of the current instruction, provided the interrupt mask is cleared. Figure 6 contains a flowchart that illustrates both the reset and interrupt sequences.

The interrupt sequence consists of one cycle during which:

The interrupt request latch is cleared;

The interrupt mode flags are selected;

The program counter (PC) is saved on the stack;

The interrupt mask is set; and

The $\overline{IRQ}$ vector jump address is loaded into the PC.

The $\overline{IRQ}$ vector jump address is $FFC-$FFD in the single-chip mode and $FF8-$FF9 in the self-check mode. The contents of these locations are not decoded as an address to which the PC should jump. Instead, they are decoded like any other EPROM program word. So, it is essential that the vector contents specify a JMP instruction in addition to the starting address of the interrupt service routine. If required, this routine should save the values of the accumulator and the X and Y registers, since these values are not stored on the stack.

Internal processing of the interrupt continues until a return from interrupt (RTI) instruction is processed. During RTI the interrupt mask is cleared and the program mode flags are selected. The next instruction of the program is then fetched and executed.

When the interrupt was initially detected and the interrupt sequence started, the interrupt request latch was cleared so that the next interrupt could be detected. These steps occurred even as the first interrupt was being serviced. However, even though the second interrupt edge set the interrupt request latch during the first interrupt's processing, the second interrupt's sequence can not begin until completion of the interrupt service routine for the first interrupt. Completion of an interrupt service routine is always accomplished using an RTI instruction to return to the main program. The interrupt mask, which is not directly available to the programmer, is cleared during the last cycle of the RTI instruction.

LEVEL-SENSITIVE OPTION

Actual operation of the level-sensitive and edge-sensitive options are similar. However, the level-sensitive option does not have an interrupt request latch. Since there is no interrupt request latch, the logic level of the $\overline{IRQ}$ pin is checked to detect the interrupt. Also, in the interrupt sequence there is no need to clear the interrupt request latch. These differences are shown in Figure 6.

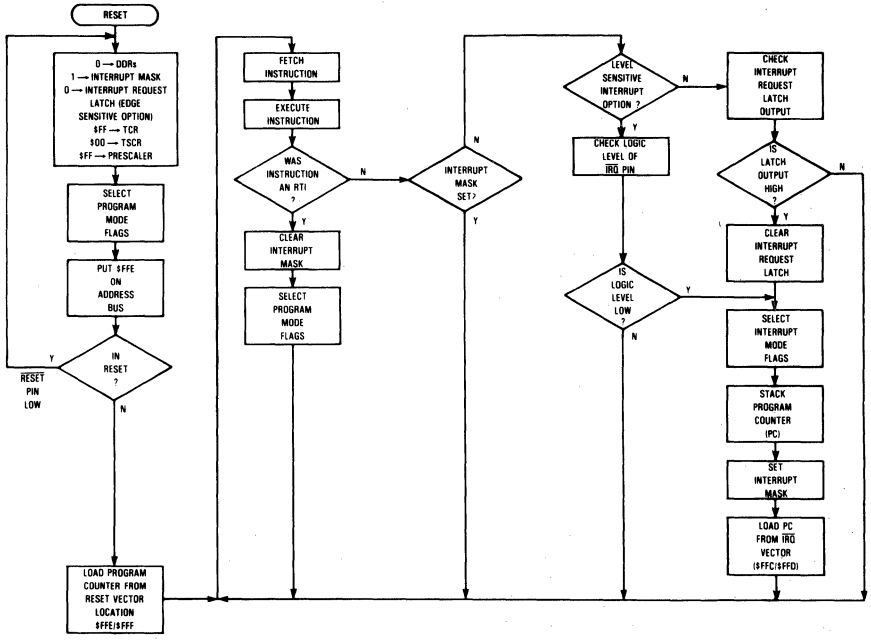


Figure 6. Reset and Interrupt Flowchart

POWERUP AND TIMING

During the powerup sequence, the interrupt mask is closed. This precludes any false interrupts. The PC is also loaded with the appropriate restart vector (jump instruction).

To open the interrupt mask, the user should do a JSR to an initialization subroutine that ends with an RTI instead of an RTS. The RTI opens the interrupt mask. Typical $\overline{\text{RESET}}$ and $\overline{\text{IRQ}}$ processes and their relationship to the interrupt mask are shown in Figure 7.

Maximum interrupt response time is six machine cycles. This includes five cycles for the longest instruction plus one for stacking the PC and switching flags.

TIMER

A block diagram of the MC6804J1 timer circuitry is shown in Figure 8. The timer logic in the MCU is comprised of a simple 8-bit counter called the timer counter. This counter is decremented by a 7-bit prescaler at a rate determined by the timer status/control register (TSCR).

PRESCALER

The prescaler is a 7-bit counter used to extend the maximum interval of the overall timer. This counter is clocked by a signal from the TIMER pin or by the internal sync pulse. It divides the frequency received by some factor

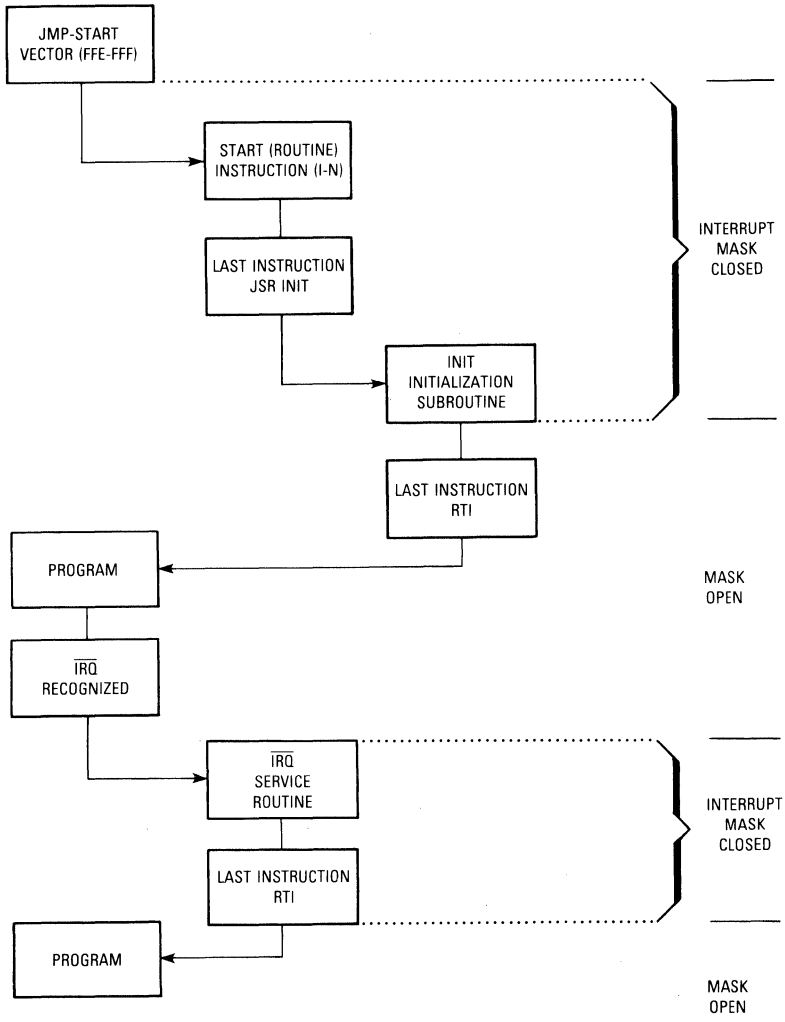
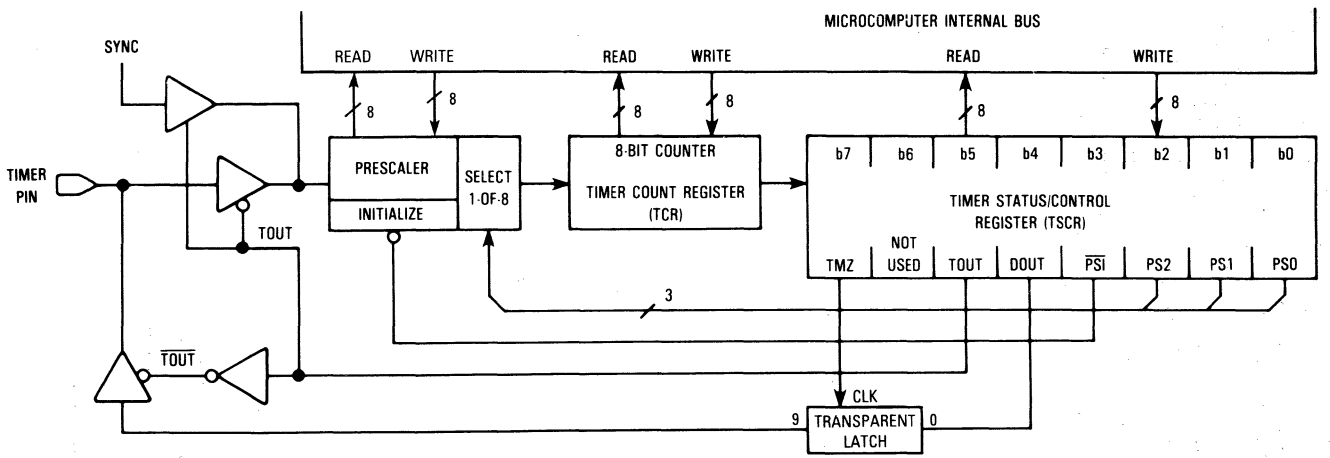


Figure 7. Interrupt Mask



TIMER PIN STATUS

TOUT	PRESCALER CLOCK	TIMER PIN
0	TIMER PIN	INPUT MODE
1	SYNC	OUTPUT MODE

Figure 8. Timer Block Diagram

to create the prescaler output. The factor by which the TIMER pin signal is divided is called the prescaler tap. The value of this tap is selected by three bits of the TSCR (PS0-PS2). These bits control the division of the prescaler input within the range of divide-by-2^0, to divide-by-2^7.

TIMER COUNTER

The timer counter, which may be read or loaded under program control, is decremented from a maximum value of 256 toward zero by the prescaler output. Both are decremented on rising clock edges.

The prescaler register and timer count register are readable and writeable. A write to either one will take precedence over the normal counter function. For example, if a value is written to the timer count register, and this write and a decrement-to-zero occur at the same time, the write takes precedence. TSCR bit one (TMZ) is not set until the next timer time out.

TIMER PIN

The TIMER pin may be programmed as either an input or an output. Its status depends on the value of TSCR bit 5 (TOUT). This relationship is shown in the TIMER pin status section of Figure 8. The frequency of the internal clock applied to the TIMER pin must be less than t_{byte}, which is (f_{OSC}/48).

TIMER INPUT MODE

In the timer input mode, TOUT is logic zero and the TIMER pin is connected directly to prescaler input. So, the prescaler is clocked by the signal from the TIMER pin. The prescaler divides the TIMER pin clock input by the prescaler tap. The prescaler output then clocks the 8-bit timer count register. When this register is decremented to zero, it sets TSCR bit one (TMZ). This TMZ bit can be tested under program control.

TIMER OUTPUT MODE

In the output mode, the TIMER pin is output. TOUT is a logic one. The prescaler is clocked by the internal sync pulse. This pulse is a divide-by-48 of the internal oscillator (f_{OSC}/48). From this point on, operation is similar to that described for the input mode. However, in the output mode, once the prescaler decrements the timer counter to zero, the high TMZ bit state is used to latch the data at TSCR bit 4 (DOUT), onto the TIMER pin.

NOTE

TMZ is normally set to logic one when TCR decrements to zero and the timer times out. However, it may be set by a write of $00 to the timer counter or by a write to bit 7 of TSCR.

TIMER COUNT REGISTER ($FE)

The timer count register reflects the current count in the internal 8-bit counter. The register is the timer counter and can be read or written.

7							0
MSB							LSB

RESET:
1 1 1 1 1 1 1 1

TIMER STATUS/CONTROL REGISTER (TSCR) ($09)

7	6	5	4	3	2	1	0
TMZ	—	TOUT	DOUT	$\overline{PSI}$	PS2	PS1	PS0

RESET:
0 0 0 0 0 0 0 0

TMZ — Timer Zero
1 = Timer count register has decremented to zero since the last time the TMZ bit was read.
0 = This bit is cleared by a read of the TSCR if TMZ is read as logic one.

Bit 6
Not used by this register.

TOUT — Timer Output
1 = Output mode is selected for the timer.
0 = Input mode is selected for the timer.

DOUT — Data Output
Latched data at this bit is sent to the TIMER pin when both the TMZ and TOUT bits are logic high.

$\overline{PSI}$ — Prescaler Initialization
1 = Prescaler begins to decrement.
0 = Prescaler is initialized and counting is inhibited.

PS0-PS2
These bits are used to select the prescaler tap. The coding of the bits is shown below:

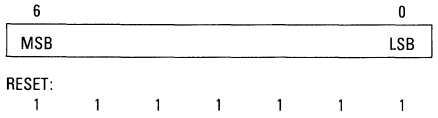
PS2	PS1	PS0	Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

It is recommended that MVI or loading and storing instructions be used when changing bit values in the TSCR. Read-modify-write instructions can cause the TMZ to assume an unexpected state.

During reset, the TSCR is set to all zeroes; the TIMER pin is in the high impedance input mode; and DOUT LATCH is forced to a logic high. At the same time, PS0-PS2 coding sets the prescaler tap at divide-by-one, and bit 3 initializes the prescaler.

TIMER PRESCALER REGISTER ($FD)

The timer prescaler register reflects the current count of the 7-bit prescaler. This register is the prescaler counter and can be read or written.



INSTRUCTION SET

The MCU has a set of 42 basic instructions. They can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

3

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is the accumulator; the other is obtained from memory using one of the addressing modes. Refer to the following list of instructions.

Function	Mnemonic
Load A from Memory	LDA
Load XP from Memory	LDX
Load YP from Memory	LDY
Store A in Memory	STA
Add to A	ADD
Subtract from A	SUB
AND Memory to A	AND
Transfer A to XP	TAX
Transfer A to YP	TAY
Transfer YP to A	TYA
Transfer XP to A	TPA
Clear A	CLRA
Clear XP	CLRX
Clear YP	CLRY
Arithmetic Compare with Memory	CMP
Move Immediate Value to Memory	MVI
Arithmetic Left Shift of A	ASLA
Complement A	COMA
Rotate A Left and Carry	ROLA

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. All INC and DEC forms along with all bit manipulation instructions use this method. Refer to the following list of instructions.

Function	Mnemonic
Increment Memory Location	INC
Increment A	INCA
Increment XP	INCX
Increment YP	INCY
Decrement Memory Location	DEC
Decrement A	DECA
Decrement XP	DECX
Decrement YP	DECY

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two byte instructions. Refer to the following list of instructions.

Function	Mnemonic
Branch if Carry Clear	BCC
Branch if Higher or Same	(BHS)
Branch if Carry Set	BCS
Branch if Lower	(BLO)
Branch if Not Equal	BNE
Branch if Equal	BEQ

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the 256 bytes of data space, where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear, and bit test and branch functions are all implemented with a single instruction. For the test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list of instructions.

Function	Mnemonic
Branch If Bit n is Set	BRSET n(n=0 . . . 7)
Branch If Bit n is Clear	BRCLR n(n=0 . . . 7)
Set Bit n	BSET n(n=0 . . . 7)
Clear Bit n	BCLR n(n=0 . . . 7)

CONTROL INSTRUCTIONS

These instructions are used to control processor operation during program execution. The jump conditional

(JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following list of instructions.

Function	Mnemonic
Return from Subroutine	RTS
Return from Interrupt	RTI
No Operation	NOP
Jump to Subroutine	JSR
Jump Unconditional	JMP

IMPLIED INSTRUCTIONS

Since the accumulator and all other registers are located in RAM, many implied instructions exist. Some of the instructions recognized and translated by the assembler are shown below:

Mnemonic	Becomes	Mnemonic	Becomes
ASLA	ADD $FF	INCX	INC $80
BHS	BCC	INCY	INC $81
BLO	BCS	LDXI	MVI $80 DATA
CLRA	SUB $FF	LDYI	MVI $81 DATA
CLRX	MVI $80 #0	NOP	BEQ (PC) +1
CLRY	MVI $81 #0	TAX	STA $80
DECA	DEC $FF	TAY	STA $81
DECX	DEC $80	TXA	LDA $80
DECY	DEC $81	TYA	LDA $81
INCA	INC $FF		

Some examples of valuable instructions not specifically recognized by the assembler are shown below:

Mnemonic	Meaning
BCLR 7,$FF	Ensures A is plus
BSET 7, $FF	Ensures A is minus
BRCLR 7, $FF	Branch if A is plus
BRSET 7, $FF	Branch if A is minus
BRCLR 7, $80	Branch if X is plus (BXPL)
BRSET 7, $80	Branch if X is minus (BXMI)
BRCLR 7, $81	Branch if Y is plus (BYPL)
BRSET 7, $81	Branch if Y is minus (BYMI)

OPCODE MAP

Table 1 is a listing of all the instruction set opcodes applicable to the MC6804J1 MCU.

ADDRESSING MODES

The MCU has nine different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. It deals with objects in three different address spaces: program space, data space, and stack space. The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is located in program ROM. It is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution, such as a constant used to initialize a loop counter.

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the 256 bytes of data space with a single two-byte instruction.

SHORT DIRECT

In the short direct addressing mode, the MCU has four locations in data space RAM it can use, ($80, $81, $82, and $83). The opcode determines the data space RAM location, and the instruction is only one byte. Short direct addressing is a subset of the direct addressing mode. The X and Y registers are at locations $80 and $81, respectively.

EXTENDED

In the extended addressing mode, the effective address of the argument is obtained by concatenating the four least-significant bits of the opcode with the byte following the opcode to form a 12-bit address. Instructions using the extended addressing mode, such as JMP or JSR, are capable of branching anywhere in program space. An extended addressing mode instruction is two bytes long.

RELATIVE

The relative addressing mode is only used in conditional branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −15 to +16 from the opcode address. The programmer need not calculate the offset when using the Motorola assembler, since it calculates the proper offset and checks to see that it is within the span of the branch.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte in which the specified bit is to be set or cleared. Thus, any bit in the 256 locations of data space memory that can be written to can be set or cleared with a single two-byte instruction.

3

Table 1. Opcode Map

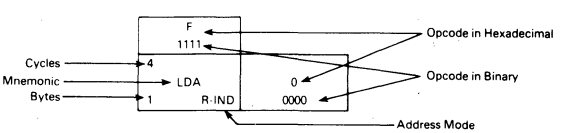
	Branch Instructions								Register/Memory, Control, and Read/Modify/Write Instructions				Bit Manipulation Instructions		Register/Memory and Read/Modify/Write		
Hi / Low	0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111	8 1000	9 1001	A 1010	B 1011	C 1100	D 1101	E 1110	F 1111	Hi / Low
0 0000	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	4 MVI 3 IMM	5 BRCLR0 3 B T B	4 BCLR0 2 BSC	4 LDA 1 R IND	4 LDA 1 R IND	0 0000
1 0001	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	*	5 BRCLR1 3 B T B	4 BCLR1 2 BSC	4 STA 1 R IND	4 STA 1 R IND	1 0001
2 0010	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	2 RTI 1 INH	5 BRCLR2 3 BT B	4 BCLR2 2 BSC	4 ADD 1 R IND	4 ADD 1 R IND	2 0010
3 0011	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	2 RTS 1 INH	5 BRCLR3 3 B T B	4 BCLR3 2 BSC	4 SUB 1 R IND	4 SUB 1 R IND	3 0011
4 0100	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	4 COMA 1 INH	5 BRCLR4 3 B T B	4 BCLR4 2 BSC	4 CMP 1 R IND	4 CMP 1 R IND	4 0100
5 0101	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	4 ROLA 1 INH	5 BRCLR5 3 B T B	4 BCLR5 2 BSC	4 AND 1 R IND	4 AND 1 R IND	5 0101
6 0110	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	*	5 BRCLR6 3 B T B	4 BCLR6 2 BSC	4 INC 1 R IND	4 INC 1 R IND	6 0110
7 0111	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	*	5 BRCLR7 3 B T B	4 BCLR7 2 BSC	4 DEC 1 R IND	4 DEC 1 R IND	7 0111
8 1000	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	4 INC 1 S D	4 DEC 1 S D	5 BRSET0 3 B T B	4 BSET0 2 BSC	4 LDA 2 IMM	4 LDA 2 DIR	8 1000
9 1001	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	4 INC 1 S D	4 DEC 1 S D	5 BRSET1 3 B T B	4 BSET1 2 BSC	#	4 STA 2 DIR	9 1001
A 1010	2 BNE 1 REL	2 BNE 2 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	4 INC 1 S D	4 DEC 1 S D	5 BRSET2 3 B T B	4 BSET2 2 BSC	4 ADD 2 IMM	4 ADD 2 DIR	A 1010
B 1011	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	4 INC 1 S D	4 DEC 1 S D	5 BRSET3 3 B T B	4 BSET3 2 BSC	SUB 2 IMM	4 SUB 2 DIR	B 1011
C 1100	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	4 LDA 1 S D	4 STA 1 S D	5 BRSET4 3 B T B	4 BSET4 2 BSC	4 CMP 2 IMM	CMP DIR	C 1100
D 1101	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	4 LDA 1 S D	4 STA 1 S D	5 BRSET5 3 B T B	4 BSET5 2 BSC	4 AND 2 IMM	4 AND 2 DIR	D 1101
E 1110	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JRSn 2 EXT	4 JMPn 2 EXT	4 LDA 1 S D	4 STA 1 S D	5 BRSET6 3 B T B	4 BSET6 2 BSC	#	4 INC 2 DIR	E 1110
F 1111	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	4 LDA 1 S D	4 STA 1 S D	5 BRSET7 3 B T B	4 BSET7 2 BSC	#	4 DEC 2 DIR	F 1111

Abbreviations for Address Modes

INH	Inherent
S-D	Short Direct
B-T-B	Bit Test and Branch
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
R-IND	Register Indirect

*	Indicates Instruction Reserved for Future Use
#	Indicates Illegal Instruction

LEGEND



CAUTION

The corresponding DDRs for ports A and B are write only registers (registers at $04 and $05). A read operation on these registers is undefined. Since BSET and BCLR are read-modify-write functions, they cannot be used to set or clear a DDR bit; all "unaffected" bits would be set. Write all DDR bits in a port using a single-store instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit that is to be tested and its condition (set or clear) is included in the opcode. The data space address of the byte to be tested is in the single byte immediately following the opcode byte. The third byte is sign extended to 12 bits and becomes the offset added to the PC if the condition is true. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the 256 locations of data space. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry flag.

REGISTER-INDIRECT

In the register-indirect addressing mode, the operand is at the address in data space pointed to by the contents of one of the indirect registers, X or Y. The particular indirect register is selected by bit 4 of the opcode. Bit 4 decodes into an address that represents the register, $80 or $81. A register-indirect instruction is one byte long.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. These instructions are one byte long.

3

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range (Comm.)	T_A	0 to 70	°C
Operating Temperature Range (Ind.)	T_A	−40 to +85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature	T_J	150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{CC}$. Reliability of operation is enhanced if unused inputs except EXTAL are connected to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance	θ_{JA}	70	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.0 Vdc ± 0.5 Vdc, V_{SS} = GND, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Internal Power Dissipation — No Port Loading	P_{INT}	—	120	165	mW
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Input Capacitance	C_{in}	—	10	—	pF
Input Current ($\overline{IRQ}$, $\overline{RESET}$)	I_{in}	—	2	20	μA

SWITCHING CHARACTERISTICS

(V_{CC} = +5.0 Vdc ± 0.5 Vdc, V_{SS} = GND, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Oscillator Frequency	f_{osc}	4.0	—	11.0	MHz
Bit Time	t_{bit}	0.364	—	1.0	μs
Byte Cycle Time	t_{byte}	4.36	—	12.0	μs
$\overline{IRQ}$ and TIMER Pulse Width	t_{WL}, t_{WH}	2 × t_{byte}	—	—	—
$\overline{RESET}$ Pulse Width	t_{RWL}	2 × t_{byte}	—	—	—
$\overline{RESET}$ Delay Time (External Capacitance = 1.0 μF)	t_{RHL}	100	—	—	ms

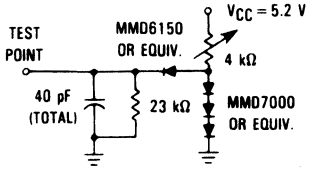


Figure 9. LSTTL Equivalent Test Load (Port B)

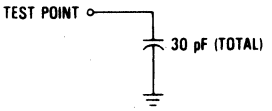


Figure 10. CMOS Equivalent Test Load (Ports A, B, C)

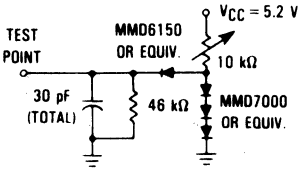


Figure 11. LSTTL Equivalent Test Load (Ports A, C, and TIMER)

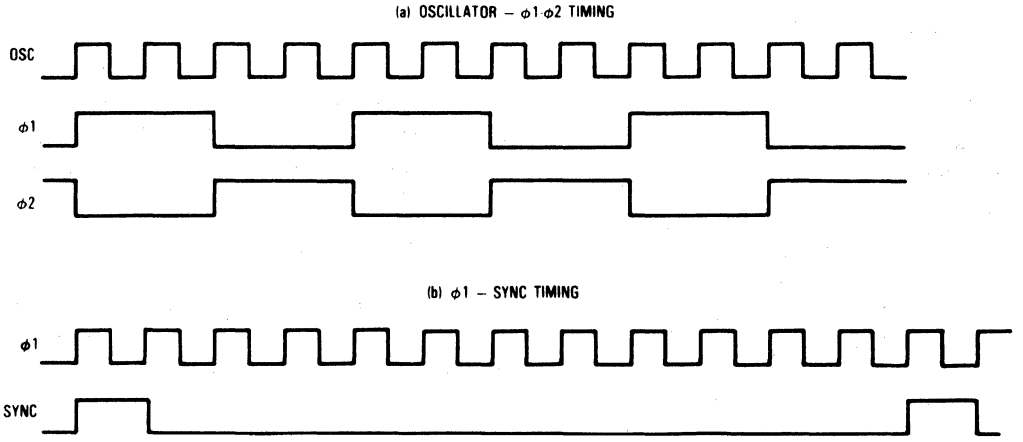


Figure 12. Clock Generator Timing Diagram

PORT DC ELECTRICAL CHARACTERISTICS
(V_{CC} = +5.0 Vdc ± 0.5 Vdc, V_{SS} = GND, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Ports A and Timer (Standard)					
Output Low Voltage, I_{Load} = 0.4 mA	V_{OL}	—	—	0.5	V
Output High Voltage, I_{Load} = −50 μA	V_{OH}	2.3	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	4	40	μA
Port A (Open Drain)					
Output Low Voltage, I_{Load} = 0.4 mA	V_{OL}	—	—	0.5	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	4	40	μA
Open Drain Leakage (V_{out} = V_{CC})	I_{LOD}	—	4	40	μA
Port A (CMOS Drive)					
Output Low Voltage, I_{Load} = 0.4 mA (Sink)	V_{OL}	—	—	0.5	V
Output High Voltage, I_{Load} = −10 μA	V_{OH}	V_{CC} − 1.0	—	—	V
Output High Voltage, I_{Load} = −50 μA	V_{OH}	2.3	—	—	V
Input High Voltage, I_{Load} = −300 μA Max	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage, I_{Load} = −300 μA Max	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current (V_{in} = 0.4 V to V_{CC})	I_{TSI}	—	—	−300	μA
Port B (Standard)					
Output Low Voltage, I_{Load} = 1.0 mA	V_{OL}	—	—	0.5	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.5	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.3	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	8	80	μA
Port B (Open Drain)					
Output Low Voltage, I_{Load} = 1.0 mA	V_{OL}	—	—	0.5	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.5	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	8	80	μA
Open Drain Leakage (V_{out} = V_{CC})	I_{LOD}	—	8	80	μA
Port B (CMOS Drive)					
Output Low Voltage, I_{Load} = 1.0 mA	V_{OL}	—	—	0.5	V
Output High Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.5	V
Output High Voltage, I_{Load} = −10 μA	V_{OH}	V_{CC} − 1.0	—	—	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.3	—	—	V
Input High Voltage, I_{Load} = −300 μA Max	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage, I_{Load} = −300 μA Max	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current (V_{in} = 0.4 V to V_{CC})	I_{TSI}	—	—	−300	μA
Ports A and B (Low Current Clamping Diode*)					
Input High Current V_{IH} = V_{CC} + 1.0 V	I_{IH}	—	—	100	μA
Input Low Current V_{IL} = 0.8 V	I_{IL}	—	—	−4.0	μA

*Denotes not tested unless specified on ordering form.

3

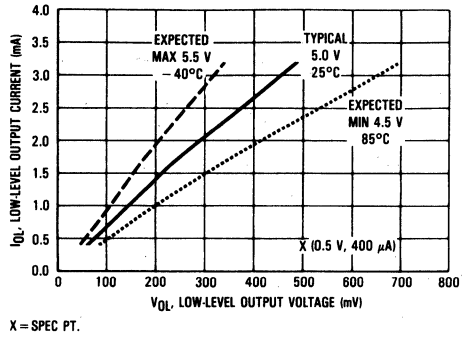


Figure 13. Typical V_{OL} vs I_{OL} for Port A and TIMER

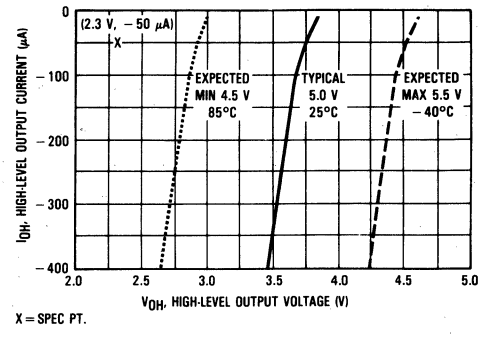


Figure 14. Typical V_{OH} vs I_{OH} for Port A and TIMER

3

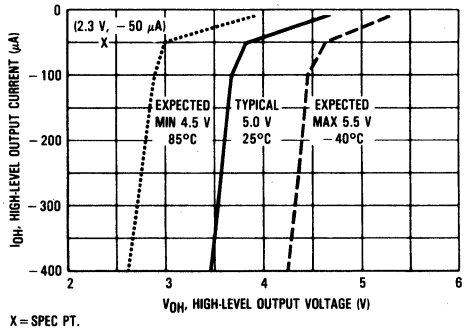


Figure 15. Typical V_{OH} vs I_{OH} for Port A with CMOS Pullups

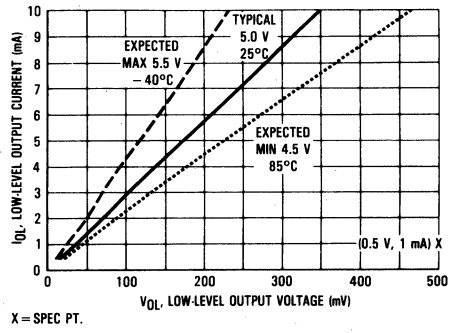


Figure 16. Typical V_{OL} vs I_{OL} for Port B

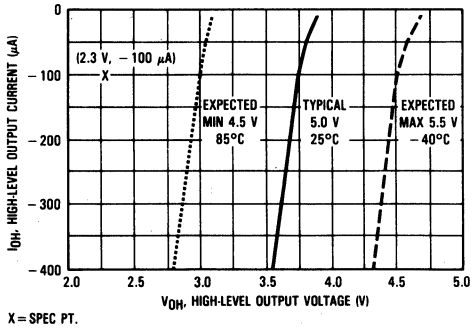


Figure 17. Typical V_{OH} vs I_{OH} for Port B

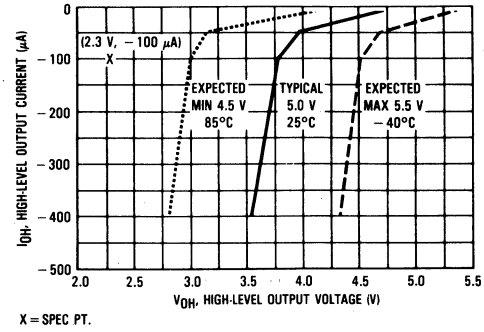


Figure 18. Typical V_{OH} vs I_{OH} for Port B with CMOS Pullups

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola using the following media:

MDOS™, disk file
MS™-DOS/PC-DOS disk file (360K)
EPROM(s) 2516, 2716, 2532, 2732

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, sales person, or Motorola representative.

FLEXIBLE DISKS

Several types of flexible disks (MDOS or MS-DOS/PC-DOS disk file) may be submitted for pattern generation. They should be programmed with the customer program, using positive logic sense for address and data. The diskette should be clearly labeled with the customer's name, date, project or product name, and the filename containing the pattern.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to expedite the process in case of any difficulty with the pattern file.

MDOS Disk File

MDOS is Motorola's Disk Operating System available on the EXORciser® development system. The disk media submitted must be a single-sided, single-density, 8-inch MDOS compatible floppy diskette. The diskette must contain the minimum set of MDOS system files in addition to the pattern file.

The .LO output of the M6804 cross assembler should be furnished. In addition, the file must be produced using the ROLLOUT command, so that it contains the absolute image of the M6804 memory. It is necessary to include the entire memory image of both program and data space. All unused bytes, including those in the user space, must be set to logic zero.

MS-DOS/PC-DOS Disk File

MS-DOS is Microsoft's Disk Operating System. PC-DOS is IBM®'s Personal Computer Disk Operating System. Disk media submitted must be standard density (360K), double-sided 5 1/4 inch compatible floppy diskette. The diskette must contain the object file code in Motorola's S-record format. The S-record format is a character-based object file format generated by M6804 cross assemblers and linkers on IBM PC style machines.

EPROMS

Four K of EPROM are necessary to contain the entire MC6804J1 program. Two 2516 or 2716 type EPROMs or a single 2532 or 2732 type EPROM can be submitted for pattern generation. The EPROM is programmed with the customer program using positive logic sense for address and data. Submissions on two EPROMs must be clearly marked. All unused bytes, including the user's space, must be set to zero.

If the MC6804J1 MCU ROM pattern is submitted on one 2532 or 2732 EPROM, or on two 2516 or 2716 type EPROMs, memory map addressing is one-for-one. The data space ROM runs from EPROM address $018 to $05F and program space ROM runs from EPROM address $E00 to $FF7, with vectors from $FFC to $FFF.

For shipment to Motorola, EPROMs should be placed in a conductive IC carrier and packed securely. Styrofoam is not acceptable for shipment.

Verification Media

All original pattern media, EPROMs or floppy disks, are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked and the verification form should be completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for the creation of the customer mask. To aid in the verification process, Motorola will program customer supplied blank EPROM(s) or DOS disks from the data file used to create the custom mask.

ROM Verification Units (RVUs)

Ten MCUs containing the customers ROM pattern will be sent for program verification. These units will have been made using the custom mask, but are for the purpose of ROM verification only. For expediency, the MCUs are unmarked, packaged in ceramic, and tested with five volts at room temperature. These RVUs are free with the minimum order quantity, but are not production parts. These RVUs are not guaranteed by Motorola Quality Assurance.

Ordering Information

The following table provides generic information pertaining to the package type and temperature for the MC6804J1. This MCU device is available in the 20-pin dual-in-line (DIP) package.

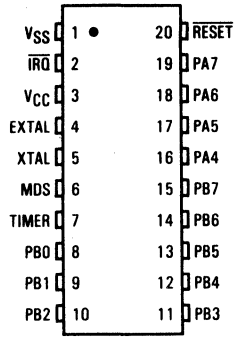
Generic Information

Package Type	Temperature	Order Number
Plastic (P Suffix)	0°C to 70°C	MC6804J1P
	−40°C to +85°C	MC6804J1CP

MDOS is a trademark of Motorola Inc.
MS-DOS is a trademark of Microsoft, Inc.
EXORciser is a registered trademark of Motorola Inc.
IBM is a registered trademark of International Business Machines Corporation.

MECHANICAL DATA

PIN ASSIGNMENTS



3

MC6804J2

Technical Summary
8-Bit Microcomputer Unit

MC6804J2 HMOS (high-density NMOS) microcomputer unit (MCU) is a member of the M6804 Family of serial processing microcomputers. This device displays all the versatility of an MCU whose design-ability to process 8-bit variables one bit at a time already makes it tremendously cost effective.

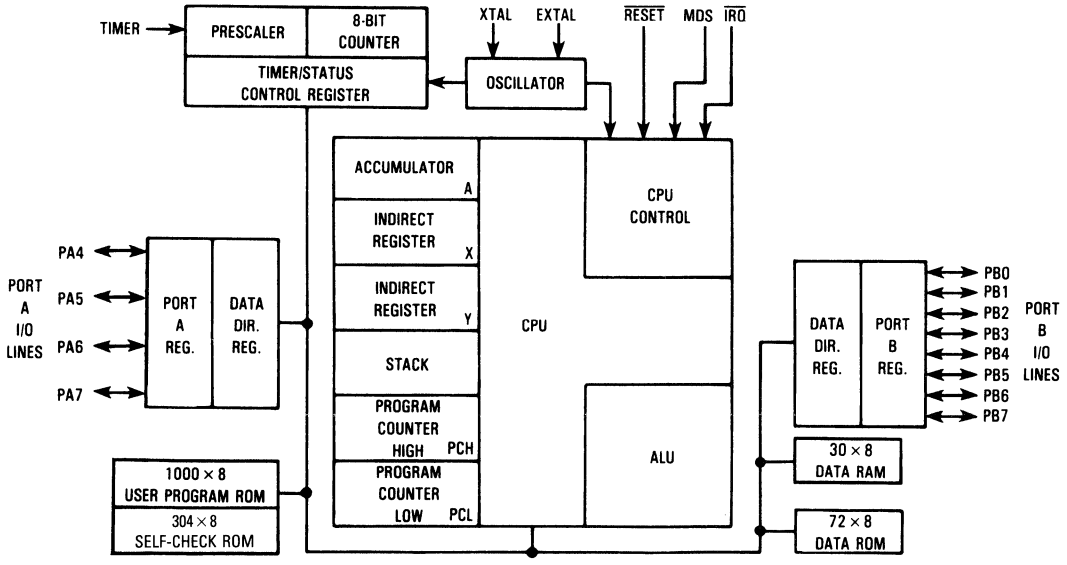
This technical summary contains limited information on the MC6804J2. For detailed information, refer to the advanced information data sheet for the MC6804J1, MC6804J2, MC6804P2, and MC68704P2 8-bit microcomputers (MC6804J1/D) or to the *M6804 MCU Manual* (DLE404/D).

Major hardware and software features of the MC6804J2 MCU are:

- On-Chip Clock Generator
- Memory Mapped I/O
- Software Programmable 8-Bit Timer with 7-Bit Prescaler
- Single Instruction Memory Examine/ Change
- 30 Bytes of Data RAM
- True Bit Manipulation
- Bit Test and Branch Instruction
- 304 Bytes Self-Check ROM
- Conditional Branches
- Timer Pin is Software Programmable as Clock Input or Timer Output
- 1000 Bytes of User Program Space ROM
- User Selectable Output Drive Options, LSTTL, LSTTL/CMOS, and Open-Drain Interface Ports
- Mask Selectable Edge- or Level-Sensitive Interrupt Pin

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

V_{CC} AND V_{SS}

Power is supplied to the microcomputer using these two pins. V_{CC} is +5 volts (±0.5 V) power, and V_{SS} is ground.

$\overline{IRQ}$

This pin provides the capability for asynchronously applying an external interrupt to the microcomputer.

EXTAL AND XTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a resistor/capacitor combination, or an external signal is connected to these pins to provide a system clock. Selection is made by a manufacturing mask option. The different clock generator options are shown in Figure 1, along with crystal specifications.

3

Internal Clock Options

The crystal oscillator start-up time is a function of many variables. To ensure rapid oscillator start-up, neither the crystal characteristics nor load capacitances should exceed recommendations. When using the on-board oscillator, the MCU should remain in a reset condition, with the $\overline{RESET}$ pin voltage below V_{IRES+}, until the oscillator has stabilized at its operating frequency. See Figure 2 for resistor/capacitor oscillator options.

TIMER

The TIMER pin can be configured to operate in either the input or output mode. As input, this pin is connected to the prescaler input and serves as the timer clock. As output, the timer pin reflects the contents of the DOUT bit of the timer status/control register, the last time the TMZ bit was logic high.

$\overline{RESET}$

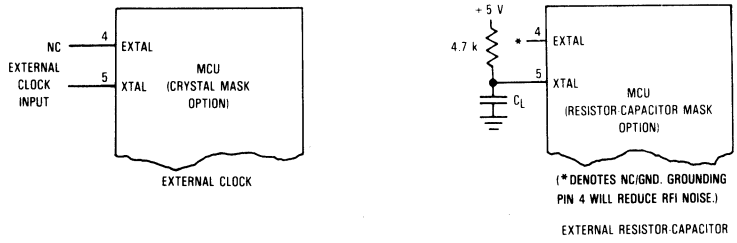
The $\overline{RESET}$ pin is used to restart the processor to the beginning of a program. The program counter is loaded with the address of the restart vector. This should be a jump instruction to the first instruction of the main program. Together with the MDS pin, the $\overline{RESET}$ pin selects the operating mode of the MCU.

MDS

The mode select (MDS) pin places the MCU into special operating modes. When this pin is logic high at the exit of the reset state, the decoded state of PA6 and PA7 is latched to determine the operating mode. This choice can be either the single-chip, self-check, or EPROM programming. However, if MDS is logic low at the end of the reset state, the single-chip operating mode is automatically selected. No external diodes, switches, transistors, etc. are required for single-chip mode selection.

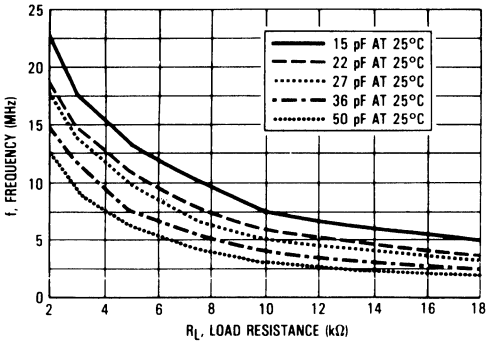
INPUT/OUTPUT LINES (PA4-PA7, PB0-PB7)

These 12 lines are arranged into one 4-bit port (A) and one 8-bit port (B). All lines are programmable as either

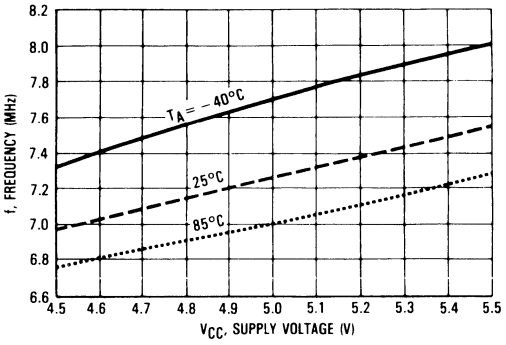


NOTE: Keep crystal leads and circuit connections as short as possible.

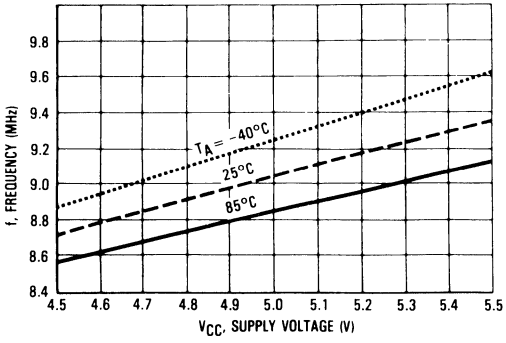
Figure 1. Clock Generator Options and Crystal Parameters



(a) TYPICAL FREQUENCY VS RESISTANCE



(b) TYPICAL FREQUENCY VARIATIONS @ C_L = 15 pF, 10 kΩ



(c) TYPICAL FREQUENCY VARIATIONS @ C_L = 50 pF, 3 kΩ

Figure 2. Typical Frequency Selection for Resistor/Capacitor Oscillator Options

inputs or outputs under software control of the data direction registers.

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

There are 12 input/output pins. All pins of each port are programmable as inputs or outputs under the control of the data direction registers (DDR).

The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic one for output, or a logic zero for input, as shown in Figure 3. When the registers are programmed as outputs, the latched data is readable regardless of the logic levels at the output pin due to output loading.

All the I/O pins are LSTTL compatible as both inputs and outputs. In addition, both ports may use either or both of two manufacturing mask options; open drain output, or internal pull-up resistor for CMOS compatibility.

Any write to a port writes to all of its data bits even though the port DDR may be set to input. This can be used as a tool to initialize the data registers and avoid undefined outputs. However, care must be exercised when using read-modify-write instructions. The data read corresponds to the pin level if the DDR is an input or to the latched output data when the DDR is an output.

The 12 bidirectional lines may be configured by port to be the standard configuration (LSTTL), or either mask option; LSTTL/CMOS, or open drain. Port B outputs are LED compatible.

Port Data Registers ($00, $01)

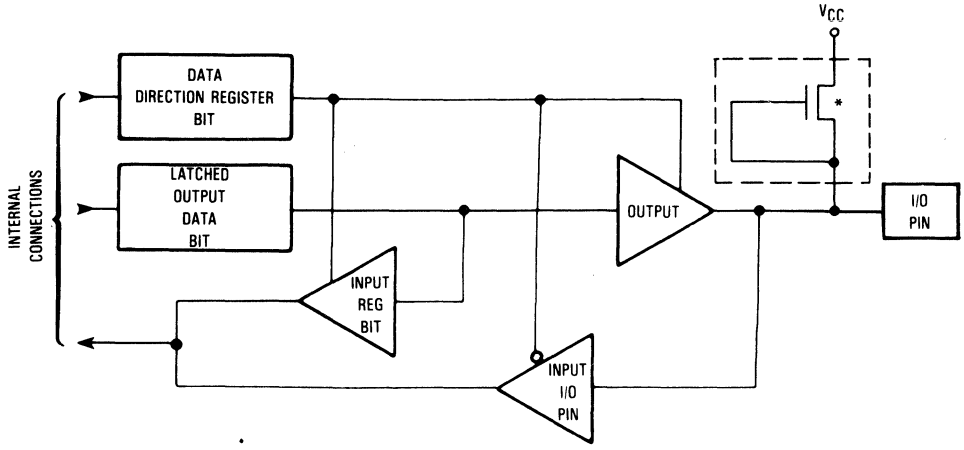
The port data registers are not initialized on reset. These registers should be initialized before changing the DDR bits to avoid undefined levels.

Port A ($00)

7	6	5	4	3	2	1	0
				X	X	X	X

Port B ($01)

7	6	5	4	3	2	1	0



DATA DIRECTION REGISTER BIT	OUTPUT DATA BIT	OUTPUT STATE	INPUT TO MCU
1	0	0	0
1	1	1	1
0	X	HI-Z	PIN

*For CMOS option transistor acts as resistor (approximately 40 kΩ) to V_{CC}.
For LSTTL/open-drain options transistor acts as low current clamping diode to V_{CC}.

Figure 3. Typical I/O Port Circuitry

With regard to Port A only, the four LSB bits are unused. These bits are "don't care" (X) bits when written to but are always logic high when read.

Port Data Direction Registers ($04, $05)

Port DDRs configure the port pins as either outputs or inputs. Each port pin can be programmed individually to be an input or an output. A zero in the pin's corresponding DDR bit programs it as an input; a logic one programs it as an output. On reset, all the DDRs are initialized to a logic zero state to put the ports in the input mode.

Port A ($04)

7	6	5	4	3	2	1	0
				0	0	0	0

Port B ($05)

7	6	5	4	3	2	1	0

With regard to Port A only, the four LSB bits are cleared (logic zero) by reset. These bits must not be set (logic one).

MEMORY

The MCU memory map (Figure 4) consists of 4352 bytes of addressable memory, I/O register locations, and four levels of stack space. This MCU has three separate memory spaces: program space, data space, and stack space.

The MCU is capable of addressing 4096 bytes of program space memory with its program counter and 256 bytes of data space memory with its instructions. Program space memory includes self-check ROM, program ROM, self-check and user program vectors, and reserved memory locations.

A non-accessible subroutine stack space RAM is provided. This stack space consists of a last-in-first-out (LIFO) register. This register is used with inherent addressing to stack the return address for subroutines.

Indirect X and Y register locations $80 and $81 are generally used as pointers for such tasks as indirect addressing to data space locations. Short direct addressing allows access to the four data space addresses $80-$83 with single-byte opcodes. The operations allowed are increment, decrement, load, and store. Data space locations $82 and $83 can be used for 8-bit counter locations.

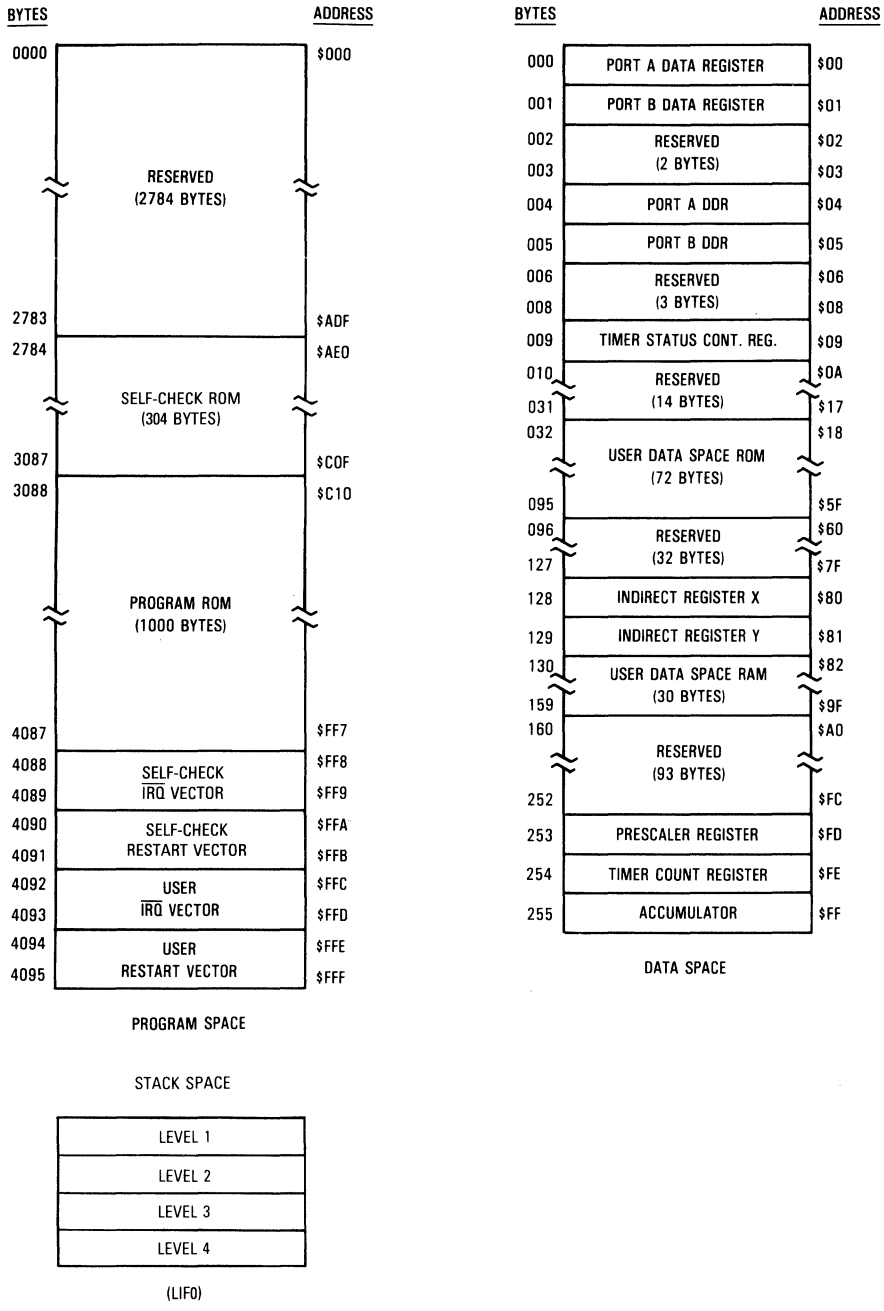


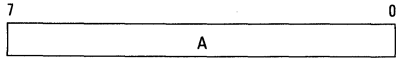
Figure 4. Memory Map

3

REGISTERS

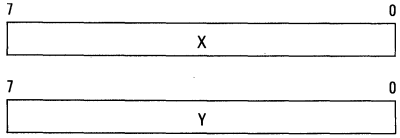
ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



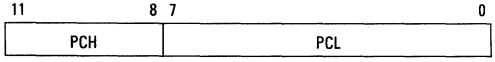
INDIRECT REGISTERS (X,Y)

These two registers are used to maintain pointers to other memory locations in data space. They are used in the register-indirect addressing mode and can be accessed with the direct, indirect, short direct, or bit set/clear modes.



PROGRAM COUNTER (PC)

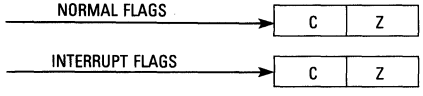
The program counter is a 12-bit register that contains the address of the next byte to be fetched. The program counter is contained in low byte (PCL) and high nibble (PCH).



FLAGS (C,Z)

The first flag, the carry (C) bit, is set on a carry or borrow out of the arithmetic logic unit (ALU). It is cleared if the arithmetic operation does not result in a carry or borrow. The C bit is also set to the value of the bit tested in a bit test instruction. It participates in the rotate left (ROLA) instruction, as well.

The second flag, the zero (Z) bit, is set if the result of the last arithmetic or logic operation was equal to zero. Otherwise, it is cleared. Bit test instructions do not affect the Z bit.



There are two sets of these flags. One set is for interrupt processing (interrupt mode flags). The other set is for normal operations (program mode flags). When an interrupt occurs, a context switch is made from the program flags to the interrupt flags. An RTI forces the context switch back. While in either mode, only the flags for that mode are available. A context switch does not affect the value of the C or Z bits. Both sets of flags are cleared by $\overline{\text{RESET}}$.

STACK

A last-in-first-out (LIFO) stack is incorporated in the MCU that eliminates the need for a stack pointer. This non-accessible subroutine stack space is implemented in separate RAM, 12 bits wide. Whenever a subroutine call or interrupt occurs, the contents of the PC are shifted into the top register of the stack. At the same time, the top register is shifted one level deeper. This happens to all registers, with the bottom register falling out of the stack.

Whenever a return from subroutine or interrupt occurs, the top register is shifted into the PC and all lower registers are shifted one level higher. The stack RAM is four levels deep. If the stack is pulled more than four times with no pushes, then the address that was stored in the bottom level of the stack is shifted into the PC.

SELF CHECK

The MCU implements two forms of internal check: self check and ROM verify. Self check performs an extensive functional check of the MCU using a signature analysis technique. ROM verify uses a similar method to check the contents of program ROM.

Self-check mode is selected by holding the MDS and PA7 pins logic high and the PA6 pin logic low as $\overline{\text{RESET}}$ goes low to high. ROM verify mode is entered by holding MDS, PA7, and PA6 logic high as $\overline{\text{RESET}}$ goes low to high. Unimplemented program space ROM locations are also tested. Monitoring the self-check mode's stages for successful completion requires external circuitry, see *M6804 MCU Manual* (DLE404/D).

RESET

$\overline{\text{RESET}}$

All resets of the MC6804J2 are caused by the external reset input ($\overline{\text{RESET}}$). A reset can be achieved by pulling the $\overline{\text{RESET}}$ pin to logic low for a minimum of 96 oscillator cycles.

During reset, a delay of 96 oscillator cycles is needed before allowing the $\overline{\text{RESET}}$ input to go high. If power is being applied, $\overline{\text{RESET}}$ must be held low long enough for the oscillator to stabilize and then provide the 96 clocks. Connecting a capacitor and resistor to the $\overline{\text{RESET}}$ input, as shown in Figure 5 below, typically provides sufficient delay.

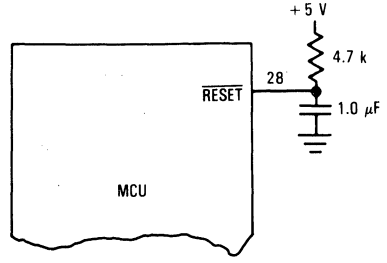


Figure 5. Powerup RESET Delay Circuit

INTERRUPT

The MCU can be interrupted by applying a logic low signal to the $\overline{IRQ}$ pin. However, a manufacturing mask option determines whether the falling edge or the actual low level of the $\overline{IRQ}$ pin is sensed to indicate an interrupt.

EDGE-SENSITIVE OPTION

When the $\overline{IRQ}$ pin is pulled low, the internal interrupt request latch is set. Prior to each instruction fetch, this interrupt request latch is tested. If its output is low, an interrupt sequence is initiated at the end of the current instruction, provided the interrupt mask is cleared. Figure 6 contains a flowchart that illustrates both the reset and interrupt sequences.

The interrupt sequence consists of one cycle during which:

The interrupt request latch is cleared;

The interrupt mode flags are selected;

The program counter (PC) is saved on the stack;

The interrupt mask is set; and

The $\overline{IRQ}$ vector jump address is loaded into the PC.

The $\overline{IRQ}$ vector jump address is $FFC-$FFD in the single-chip mode and $FF8-$FF9 in the self-check mode. The contents of these locations are not decoded as an address to which the PC should jump. Instead, they are decoded like any other EPROM program word. So, it is essential that the vector contents specify a JMP instruction in addition to the starting address of the interrupt service routine. If required, this routine should save the values of the accumulator and the X and Y registers, since these values are not stored on the stack.

Internal processing of the interrupt continues until a return from interrupt (RTI) instruction is processed. During RTI the interrupt mask is cleared and the program mode flags are selected. The next instruction of the program is then fetched and executed.

When the interrupt was initially detected and the interrupt sequence started, the interrupt request latch was cleared so that the next interrupt could be detected. These steps occurred even as the first interrupt was being serviced. However, even though the second interrupt edge set the interrupt request latch during the first interrupt's processing, the second interrupt's sequence can not begin until completion of the interrupt service routine for the first interrupt. Completion of an interrupt service routine is always accomplished using an RTI instruction to return to the main program. The interrupt mask, which is not directly available to the programmer, is cleared during the last cycle of the RTI instruction.

LEVEL-SENSITIVE OPTION

Actual operation of the level-sensitive and edge-sensitive options are similar. However, the level-sensitive option does not have an interrupt request latch. Since there is no interrupt request latch, the logic level of the

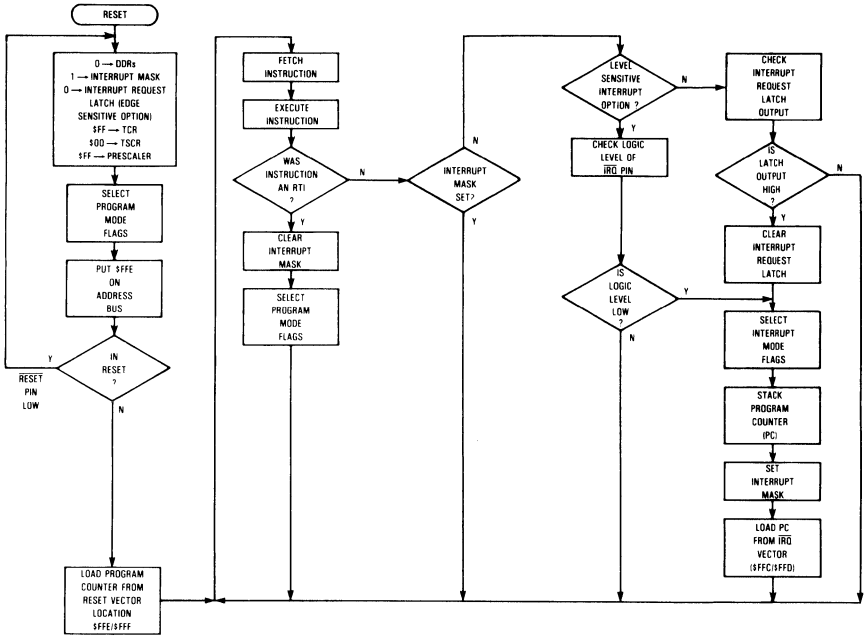


Figure 6. Reset and Interrupt Flowchart

$\overline{IRQ}$ pin is checked to detect the interrupt. Also, in the interrupt sequence there is no need to clear the interrupt request latch. These differences are shown in Figure 6.

POWERUP AND TIMING

During the powerup sequence, the interrupt mask is closed. This precludes any false interrupts. The PC is also loaded with the appropriate restart vector (jump instruction).

To open the interrupt mask, the user should do a JSR to an initialization subroutine that ends with an RTI instead of an RTS. The RTI opens the interrupt mask. Typical $\overline{RESET}$ and $\overline{IRQ}$ processes and their relationship to the interrupt mask are shown in Figure 7.

Maximum interrupt response time is six machine cycles. This includes five cycles for the longest instruction plus one for stacking the PC and switching flags.

TIMER

A block diagram of the MC6804J2 timer circuitry is shown in Figure 8. The timer logic in the MCU is comprised of a simple 8-bit counter called the timer counter. This counter is decremented by a 7-bit prescaler at a rate determined by the timer status/control register (TSCR).

PRESCALER

The prescaler is a 7-bit counter used to extend the maximum interval of the overall timer. This counter is clocked

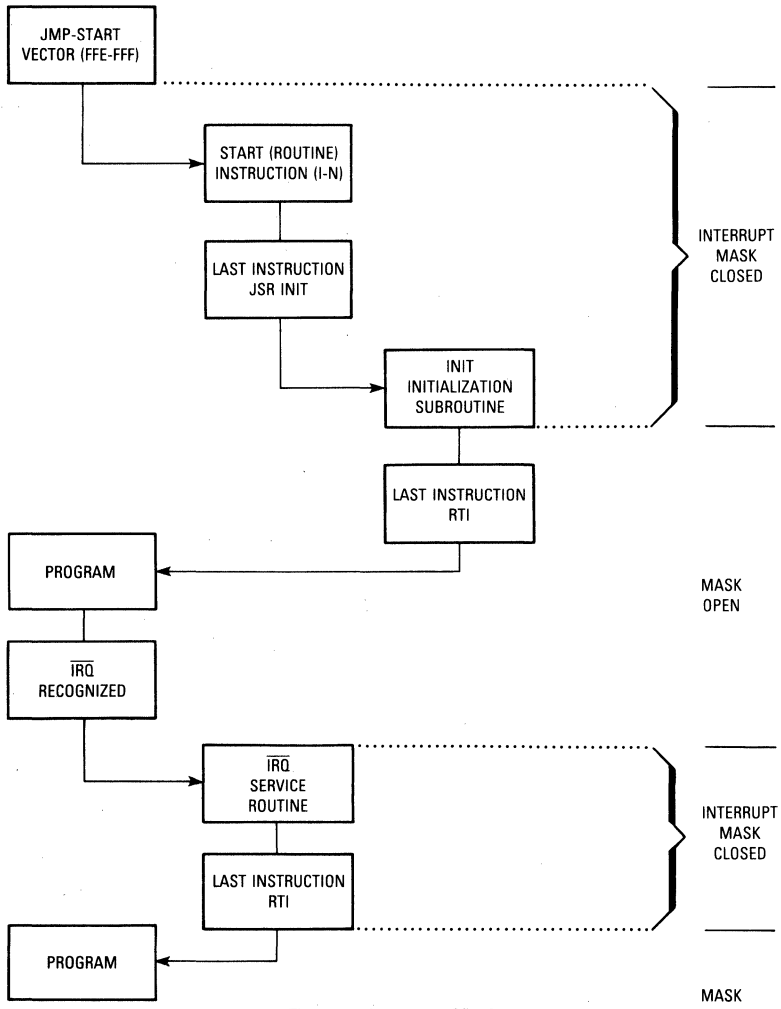
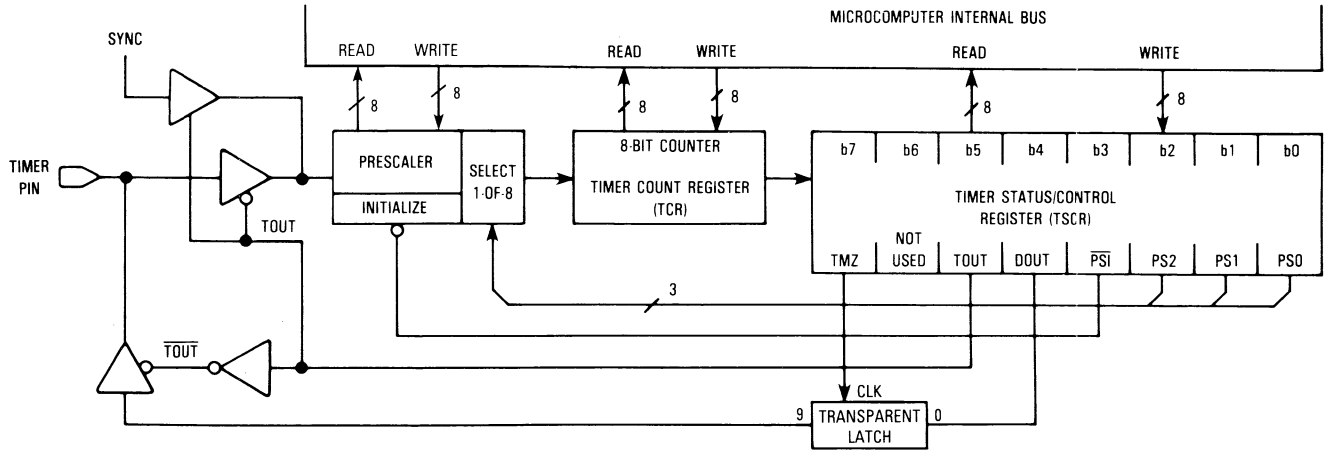


Figure 7. Interrupt Mask



TIMER PIN STATUS

TOUT	PRESCALER CLOCK	TIMER PIN
0	TIMER PIN	INPUT MODE
1	SYNC	OUTPUT MODE

Figure 8. Timer Block Diagram

by a signal from the TIMER pin or by the internal sync pulse. It divides the frequency received by some factor to create the prescaler output. The factor by which the TIMER pin signal is divided is called the prescaler tap. The value of this tap is selected by three bits of the TSCR (PS0-PS2). These bits control the division of the prescaler input within the range of divide-by-2^0, to divide-by-2^7.

TIMER COUNTER

The timer counter, which may be read or loaded under program control, is decremented from a maximum value of 256 toward zero by the prescaler output. Both are decremented on rising clock edges.

The prescaler register and timer count register are readable and writeable. A write to either one will take precedence over the normal counter function. For example, if a value is written to the timer count register, and this write and a decrement-to-zero occur at the same time, the write takes precedence. TSCR bit one (TMZ) is not set until the next timer time out.

3

TIMER PIN

The TIMER pin may be programmed as either an input or an output. Its status depends on the value of TSCR bit 5 (TOUT). This relationship is shown in the TIMER pin status section of Figure 8. The frequency of the internal clock applied to the TIMER pin must be less than t_{byte}, which is (f_{osc}/48).

TIMER INPUT MODE

In the timer input mode, TOUT is logic zero and the TIMER pin is connected directly to prescaler input. So, the prescaler is clocked by the signal from the TIMER pin. The prescaler divides the TIMER pin clock input by the prescaler tap. The prescaler output then clocks the 8-bit timer count register. When this register is decremented to zero, it sets TSCR bit one (TMZ). This TMZ bit can be tested under program control to tell when the counter register has reached zero.

TIMER OUTPUT MODE

In the output mode, the TIMER pin is output. TOUT is a logic one. The prescaler is clocked by the internal sync pulse. This pulse is a divide-by-48 of the internal oscillator (f_{osc}/48). From this point on, operation is similar to that described for the input mode. However, in the output mode, once the prescaler decrements the timer counter to zero, the high TMZ bit state allows TSCR bit 4 (DOUT) to become direct input to the TIMER pin.

NOTE

TMZ is normally set to logic one when TCR decrements to zero and the timer times out. However, it may be set by a write of $00 to TCR or by a write to bit 7 of TSCR.

TIMER COUNT REGISTER ($FE)

The timer count register reflects the current count in the internal 8-bit counter. The register is the timer counter and can be read or written.

7							0
MSB							LSB

RESET:

1	1	1	1	1	1	1	1

TIMER STATUS/CONTROL REGISTER (TSCR) ($09)

7	6	5	4	3	2	1	0
TMZ	—	TOUT	DOUT	$\overline{PSI}$	PS2	PS1	PS0

RESET:

0	0	0	0	0	0	0	0

TMZ — Timer Zero
1 = Timer count register has decremented to zero since the last time the TMZ bit was read.
0 = This bit is cleared by a read of the TSCR if TMZ is read as logic one.

Bit 6
Not used by this register.

TOUT — Timer Output
1 = Output mode is selected for the timer.
0 = Input mode is selected for the timer.

DOUT — Data Output
Latched data at this bit is sent to the TIMER pin when both the TMZ and TOUT bits are logic high.

$\overline{PSI}$ — Prescaler Initialization
1 = Prescaler begins to decrement.
0 = Prescaler is initialized and counting is inhibited.

PS0-PS2
These bits are used to select the prescaler tap. The coding of the bits is shown below:

PS2	PS1	PS0	Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

It is recommended that MVI or loading and storing instructions be used when changing bit values in the TSCR. Read-modify-write instructions can cause the TMZ to assume an unexpected state.

During reset, the TSCR is set to all zeroes; the TIMER pin is in the high impedance input mode; and DOUT LATCH is forced to a logic high. At the same time, PS0-PS2 coding sets the prescaler tap at divide-by-one, and bit 3 initializes the prescaler.

TIMER PRESCALER REGISTER ($FD)

The timer prescaler register reflects the current count of the 7-bit prescaler. This register is the prescaler counter and can be read or written.

6						0
MSB						LSB

RESET:

1	1	1	1	1	1	1

INSTRUCTION SET

The MCU has a set of 42 basic instructions. They can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is the accumulator; the other is obtained from memory using one of the addressing modes. Refer to the following list of instructions.

Function	Mnemonic
Load A from Memory	LDA
Load XP from Memory	LDX
Load YP from Memory	LDY
Store A in Memory	STA
Add to A	ADD
Subtract from A	SUB
AND Memory to A	AND
Transfer A to XP	TAX
Transfer A to YP	TAY
Transfer YP to A	TYA
Transfer XP to A	TPA
Clear A	CLRA
Clear XP	CLRX
Clear YP	CLRY
Arithmetic Compare with Memory	CMP
Move Immediate Value to Memory	MVI
Arithmetic Left Shift of A	ASLA
Complement A	COMA
Rotate A Left and Carry	ROLA

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. All INC and DEC forms along with all bit manipulation instructions use this method. Refer to the following list of instructions.

Function	Mnemonic
Increment Memory Location	INC
Increment A	INCA
Increment XP	INCX
Increment YP	INCY
Decrement Memory Location	DEC
Decrement A	DECA
Decrement XP	DECX
Decrement YP	DECY

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two byte instructions. Refer to the following list of instructions.

Function	Mnemonic
Branch if Carry Clear	BCC
Branch if Higher or Same	(BHS)
Branch if Carry Set	BCS
Branch if Lower	(BLO)
Branch if Not Equal	BNE
Branch if Equal	BEQ

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the 256 bytes of data space, where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear, and bit test and branch functions are all implemented with a single instruction. For the test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list of instructions.

Function	Mnemonic
Branch If Bit n is Set	BRSET n(n=0 . . . 7)
Branch If Bit n is Clear	BRCLR n(n=0 . . . 7)
Set Bit n	BSET n(n=0 . . . 7)
Clear Bit n	BCLR n(n=0 . . . 7)

CONTROL INSTRUCTIONS

These instructions are used to control processor operation during program execution. The jump conditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following list of instructions.

Function	Mnemonic
Return from Subroutine	RTS
Return from Interrupt	RTI
No Operation	NOP
Jump to Subroutine	JSR
Jump Unconditional	JMP

IMPLIED INSTRUCTIONS

Since the accumulator and all other registers are located in RAM, many implied instructions exist. Some of the instructions recognized and translated by the assembler are shown below:

Mnemonic	Becomes	Mnemonic	Becomes
ASLA	ADD $FF	INCX	INC $80
BHS	BCC	INCY	INC $81
BLO	BCS	LDXI	MVI $80 DATA
CLRA	SUB $FF	LDYI	MVI $81 DATA
CLRX	MVI $80 #0	NOP	BEQ (PC) +1
CLRY	MVI $81 #0	TAX	STA $80
DECA	DEC $FF	TAY	STA $81
DECX	DEC $80	TXA	LDA $80
DECY	DEC $81	TYA	LDA $81
INCA	INC $FF		

Some examples of valuable instructions not specifically recognized by the assembler are shown below:

Mnemonic	Meaning
BCLR 7,$FF	Ensures A is plus
BSET 7, $FF	Ensures A is minus
BRCLR 7, $FF	Branch if A is plus
BRSET 7, $FF	Branch if A is minus
BRCLR 7, $80	Branch if X is plus (BXPL)
BRSET 7, $80	Branch if X is minus (BXMI)
BRCLR 7, $81	Branch if Y is plus (BYPL)
BRSET 7, $81	Branch if Y is minus (BYMI)

OPCODE MAP

Table 1 is a listing of all the instruction set opcodes applicable to the MC6804J2 MCU.

ADDRESSING MODES

The MCU has nine different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. It deals with objects in three different address spaces: program space, data space, and stack space. The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is located in program ROM. It is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution, such as a constant used to initialize a loop counter.

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the 256 bytes of data space with a single two-byte instruction.

SHORT DIRECT

In the short direct addressing mode, the MCU has four locations in data space RAM it can use, ($80, $81, $82, and $83). The opcode determines the data space RAM location, and the instruction is only one byte. Short direct addressing is a subset of the direct addressing mode. The X and Y registers are at locations $80 and $81, respectively.

EXTENDED

In the extended addressing mode, the effective address of the argument is obtained by concatenating the four least-significant bits of the opcode with the byte following the opcode to form a 12-bit address. Instructions using the extended addressing mode, such as JMP or JSR, are capable of branching anywhere in program space. An extended addressing mode instruction is two bytes long.

RELATIVE

The relative addressing mode is only used in conditional branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −15 to +16 from the opcode address. The programmer need not calculate the offset when using the Motorola assembler, since it calculates the proper offset and checks to see that it is within the span of the branch.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte in which the specified bit is to be set or cleared. Thus, any bit in the 256 locations of data space memory that can be written to can be set or cleared with a single two-byte instruction.

CAUTION

The corresponding DDRs for ports A and B are write only registers (registers at $04 and $05). A read operation on these registers is undefined. Since BSET and BCLR are read-modify-write functions, they cannot be used to set or clear a DDR bit; all "unaffected" bits would be set. Write all DDR bits in a port using a single-store instruction.

Table 1. Opcode Map

Hi / Low	Branch Instructions								Register/Memory, Control, and Read/Modify/Write Instructions				Bit Manipulation Instructions		Register/Memory and Read Modify Write		Hi / Low
	0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111	8 1000	9 1001	A 1010	B 1011	C 1100	D 1101	E 1110	F 1111	
0 0000	2 1 REL BNE	2 1 REL BNE	2 1 REL BEQ	2 1 REL BEQ	2 1 REL BCC	2 1 REL BCC	2 1 REL BCS	2 1 REL BCS	4 2 EXT JSRn	4 2 EXT JMPn	•	4 3 IMM MVI	5 3 B-T-B BRCLR0	4 2 BSC BCLR0	4 1 R-IND LDA	4 1 R-IND LDA	0 0000
1 0001	2 1 REL BNE	2 1 REL BNE	2 1 REL BEQ	2 1 REL BEQ	2 1 REL BCC	2 1 REL BCC	2 1 REL BCS	2 1 REL BCS	4 2 EXT JSRn	4 2 EXT JMPn	•	•	5 3 B-T-B BRCLR1	4 2 BSC BCLR1	4 1 R-IND STA	4 1 R-IND STA	1 0001
2 0010	2 1 REL BNE	2 1 REL BNE	2 1 REL BEQ	2 1 REL BEQ	2 1 REL BCC	2 1 REL BCC	2 1 REL BCS	2 1 REL BCS	4 2 EXT JSRn	4 2 EXT JMPn	•	2 1 INH RTI	5 3 B-T-B BRCLR2	4 2 BSC BCLR2	4 1 R-IND ADD	4 1 R-IND ADD	2 0010
3 0011	2 1 REL BNE	2 1 REL BNE	2 1 REL BEQ	2 1 REL BEQ	2 1 REL BCC	2 1 REL BCC	2 1 REL BCS	2 1 REL BCS	4 2 EXT JSRn	4 2 EXT JMPn	•	2 1 INH RTS	5 3 B-T-B BRCLR3	4 2 BSC BCLR3	4 1 R-IND SUB	4 1 R-IND SUB	3 0011
4 0100	2 1 REL BNE	2 1 REL BNE	2 1 REL BEQ	2 1 REL BEQ	2 1 REL BCC	2 1 REL BCC	2 1 REL BCS	2 1 REL BCS	4 2 EXT JSRn	4 2 EXT JMPn	•	4 1 INH COMA	5 3 B-T-B BRCLR4	4 2 BSC BCLR4	4 1 R-IND CMP	4 1 R-IND CMP	4 0100
5 0101	2 1 REL BNE	2 1 REL BNE	2 1 REL BEQ	2 1 REL BEQ	2 1 REL BCC	2 1 REL BCC	2 1 REL BCS	2 1 REL BCS	4 2 EXT JSRn	4 2 EXT JMPn	•	4 1 INH ROLA	5 3 B-T-B BRCLR5	4 2 BSC BCLR5	4 1 R-IND AND	4 1 R-IND AND	5 0101
6 0110	2 1 REL BNE	2 1 REL BNE	2 1 REL BEQ	2 1 REL BEQ	2 1 REL BCC	2 1 REL BCC	2 1 REL BCS	2 1 REL BCS	4 2 EXT JSRn	4 2 EXT JMPn	•	•	5 3 B-T-B BRCLR6	4 2 BSC BCLR6	4 1 R-IND INC	4 1 R-IND INC	6 0110
7 0111	2 1 REL BNE	2 1 REL BNE	2 1 REL BEQ	2 1 REL BEQ	2 1 REL BCC	2 1 REL BCC	2 1 REL BCS	2 1 REL BCS	4 2 EXT JSRn	4 2 EXT JMPn	•	•	5 3 B-T-B BRCLR7	4 2 BSC BCLR7	4 1 R-IND DEC	4 1 R-IND DEC	7 0111
8 1000	2 1 REL BNE	2 1 REL BNE	2 1 REL BEQ	2 1 REL BEQ	2 1 REL BCC	2 1 REL BCC	2 1 REL BCS	2 1 REL BCS	4 2 EXT JSRn	4 2 EXT JMPn	4 1 S-D INC	4 1 S-D DEC	5 3 B-T-B BRSET0	4 2 BSC BSET0	4 2 IMM LDA	4 2 DIR LDA	8 1000
9 1001	2 1 REL BNE	2 1 REL BNE	2 1 REL BEQ	2 1 REL BEQ	2 1 REL BCC	2 1 REL BCC	2 1 REL BCS	2 1 REL BCS	4 2 EXT JSRn	4 2 EXT JMPn	4 1 S-D INC	4 1 S-D DEC	5 3 B-T-B BRSET1	4 2 BSC BSET1	#	4 2 DIR STA	9 1001
A 1010	2 1 REL BNE	2 2 REL BNE	2 1 REL BEQ	2 1 REL BEQ	2 1 REL BCC	2 1 REL BCC	2 1 REL BCS	2 1 REL BCS	4 2 EXT JSRn	4 2 EXT JMPn	4 1 S-D INC	4 1 S-D DEC	5 3 B-T-B BRSET2	4 2 BSC BSET2	4 2 IMM ADD	4 2 DIR ADD	A 1010
B 1011	2 1 REL BNE	2 1 REL BNE	2 1 REL BEQ	2 1 REL BEQ	2 1 REL BCC	2 1 REL BCC	2 1 REL BCS	2 1 REL BCS	4 2 EXT JSRn	4 2 EXT JMPn	4 1 S-D INC	4 1 S-D DEC	5 3 B-T-B BRSET3	4 2 BSC BSET3	4 2 IMM SUB	4 2 DIR SUB	B 1011
C 1100	2 1 REL BNE	2 1 REL BNE	2 1 REL BEQ	2 1 REL BEQ	2 1 REL BCC	2 1 REL BCC	2 1 REL BCS	2 1 REL BCS	4 2 EXT JSRn	4 2 EXT JMPn	4 1 S-D LDA	4 1 S-D STA	5 3 B-T-B BRSET4	4 2 BSC BSET4	4 2 IMM CMP	4 2 DIR CMP	C 1100
D 1101	2 1 REL BNE	2 1 REL BNE	2 1 REL BEQ	2 1 REL BEQ	2 1 REL BCC	2 1 REL BCC	2 1 REL BCS	2 1 REL BCS	4 2 EXT JSRn	4 2 EXT JMPn	4 1 S-D LDA	4 1 S-D STA	5 3 B-T-B BRSET5	4 2 BSC BSET5	4 2 IMM AND	4 2 DIR AND	D 1101
E 1110	2 1 REL BNE	2 1 REL BNE	2 1 REL BEQ	2 1 REL BEQ	2 1 REL BCC	2 1 REL BCC	2 1 REL BCS	2 1 REL BCS	4 2 EXT JRSn	4 2 EXT JMPn	4 1 S-D LDA	4 1 S-D STA	5 3 B-T-B BRSET6	4 2 BSC BSET6	#	4 2 DIR INC	E 1110
F 1111	2 1 REL BNE	2 1 REL BNE	2 1 REL BEQ	2 1 REL BEQ	2 1 REL BCC	2 1 REL BCC	2 1 REL BCS	2 1 REL BCS	4 2 EXT JSRn	4 2 EXT JMPn	4 1 S-D LDA	4 1 S-D STA	5 3 B-T-B BRSET7	4 2 BSC BSET7	#	4 2 DIR DEC	F 1111

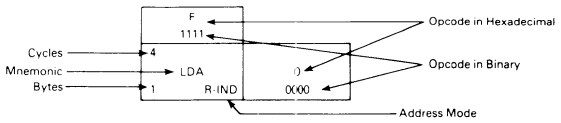
Abbreviations for Address Modes

INH	Inherent
S-D	Short Direct
B-T-B	Bit Test and Branch
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
R-IND	Register Indirect

• Indicates Instruction Reserved for Future Use

# Indicates Illegal Instruction

LEGEND



BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit that is to be tested and its condition (set or clear) is included in the opcode. The data space address of the byte to be tested is in the single byte immediately following the opcode byte. The third byte is sign extended to twelve bits and becomes the offset added to the PC if the condition is true. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the 256 locations of data space. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry flag.

REGISTER-INDIRECT

In the register-indirect addressing mode, the operand is at the address in data space pointed to by the contents of one of the indirect registers, X or Y. The particular indirect register is selected by bit 4 of the opcode. Bit 4 decodes into an address that represents the register, $80 or $81. A register-indirect instruction is one byte long.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. These instructions are one byte long.

ELECTRICAL SPECIFICATIONS

3

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range (Comm.)	T_A	0 to 70	°C
Operating Temperature Range (Ind.)	T_A	−40 to +85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature	T_J	150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{CC}$. Reliability of operation is enhanced if unused inputs except EXTAL are connected to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance	θ_{JA}	70	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.0 Vdc ± 0.5 Vdc, V_{SS} = GND, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Internal Power Dissipation — No Port Loading	P_{INT}	—	120	165	mW
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Input Capacitance	C_{in}	—	10	—	pF
Input Current ($\overline{IRQ}$, $\overline{RESET}$)	I_{in}	—	2	20	μA

SWITCHING CHARACTERISTICS

(V_{CC} = +5.0 Vdc ± 0.5 Vdc, V_{SS} = GND, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Oscillator Frequency	f_{osc}	4.0	—	11.0	MHz
Bit Time	t_{bit}	0.364	—	1.0	μs
Byte Cycle Time	t_{byte}	4.36	—	12.0	μs
$\overline{IRQ}$ and TIMER Pulse Width	t_{WL}, t_{WH}	2 × t_{byte}	—	—	—
$\overline{RESET}$ Pulse Width	t_{RWL}	2 × t_{byte}	—	—	—
$\overline{RESET}$ Delay Time (External Capacitance = 1.0 μF)	t_{RHL}	100	—	—	ms

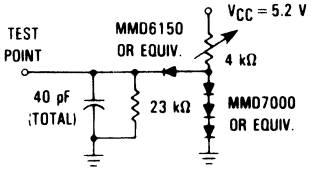


Figure 9. LSTTL Equivalent Test Load (Port B)

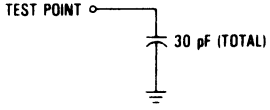


Figure 10. CMOS Equivalent Test Load (Ports A, B, C)

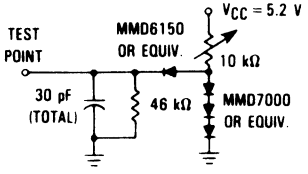


Figure 11. LSTTL Equivalent Test Load (Ports A, C, and TIMER)

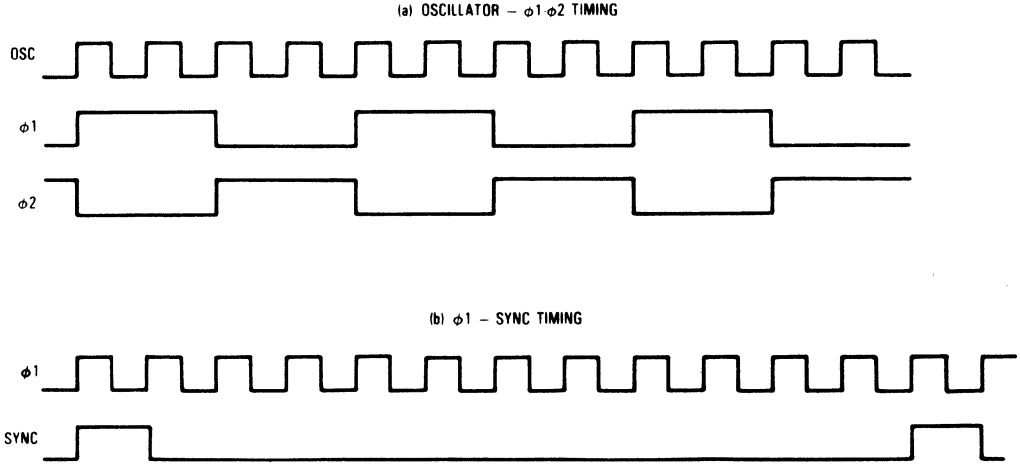


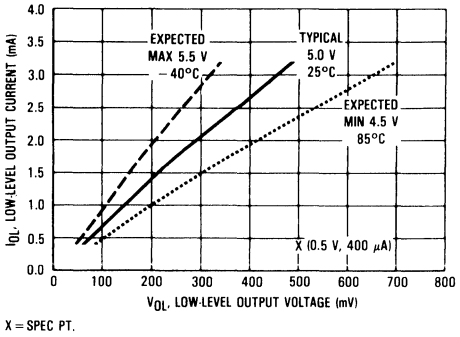
Figure 12. Clock Generator Timing Diagram

PORT DC ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.0 Vdc ± 0.5 Vdc, V_{SS} = GND, T_A = 0°C to 70°C, unless otherwise noted)

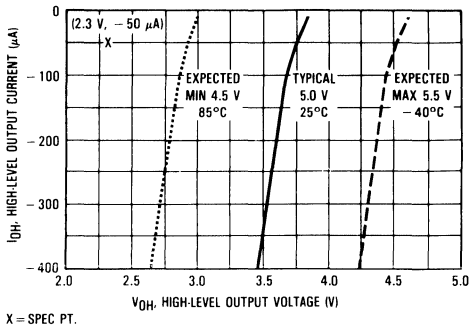
Characteristic	Symbol	Min	Typ	Max	Unit
Ports A and Timer (Standard)					
Output Low Voltage, I_{Load} = 0.4 mA	V_{OL}	—	—	0.5	V
Output High Voltage, I_{Load} = −50 μA	V_{OH}	2.3	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	4	40	μA
Port A (Open Drain)					
Output Low Voltage, I_{Load} = 0.4 mA	V_{OL}	—	—	0.5	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	4	40	μA
Open Drain Leakage (V_{out} = V_{CC})	I_{LOD}	—	4	40	μA
Port A (CMOS Drive)					
Output Low Voltage, ILoad = 0.4 mA (Sink)	V_{OL}	—	—	0.5	V
Output High Voltage, I_{Load} = −10 μA	V_{OH}	V_{CC} − 1.0	—	—	V
Output High Voltage, I_{Load} = −50 μA	V_{OH}	2.3	—	—	V
Input High Voltage, I_{Load} = −300 μA Max	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage, I_{Load} = −300 μA Max	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current (V_{in} = 0.4 V to V_{CC})	I_{TSI}	—	—	−300	μA
Port B (Standard)					
Output Low Voltage, I_{Load} = 1.0 mA	V_{OL}	—	—	0.5	V
Output Low Voltage,I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.5	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.3	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	8	80	μA
Port B (Open Drain)					
Output Low Voltage, I_{Load} = 1.0 mA	V_{OL}	—	—	0.5	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.5	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	8	80	μA
Open Drain Leakage (V_{out} = V_{CC})	I_{LOD}	—	8	80	μA
Port B (CMOS Drive)					
Output Low Voltage, I_{Load} = 1.0 mA	V_{OL}	—	—	0.5	V
Output High Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.5	V
Output High Voltage, I_{Load} = −10 μA	V_{OH}	V_{CC} − 1.0	—	—	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.3	—	—	V
Input High Voltage, I_{Load} = −300 μA Max	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage, I_{Load} = −300 μA Max	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current (V_{in} = 0.4 V to V_{CC})	I_{TSI}	—	—	−300	μA
Ports A and B (Low Current Clamping Diode*)					
Input High Current V_{IH} = V_{CC} + 1.0 V	I_{IH}	—	—	100	μA
Input Low Current V_{IL} = 0.8 V	I_{IL}	—	—	−4.0	μA

*Denotes not tested unless specified on ordering form.



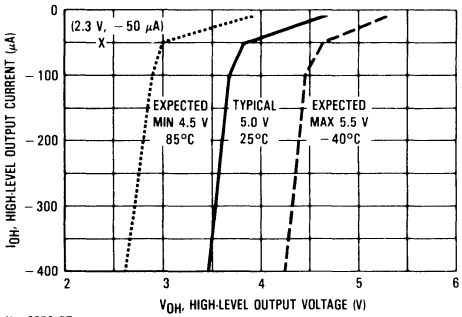
X = SPEC PT.

Figure 13. Typical V_{OL} vs I_{OL} for Port A and TIMER



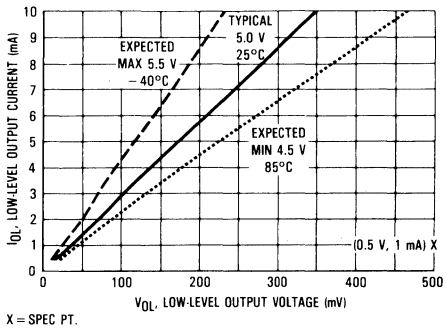
X = SPEC PT.

Figure 14. Typical V_{OH} vs I_{OH} for Port A and TIMER



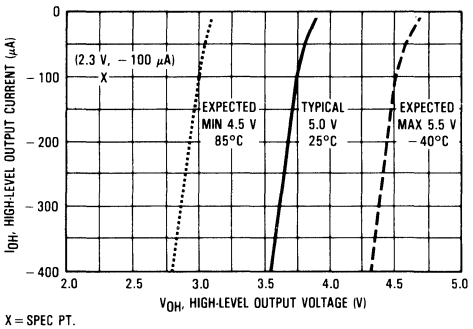
X = SPEC PT.

Figure 15. Typical V_{OH} vs I_{OH} for Port A with CMOS Pullups



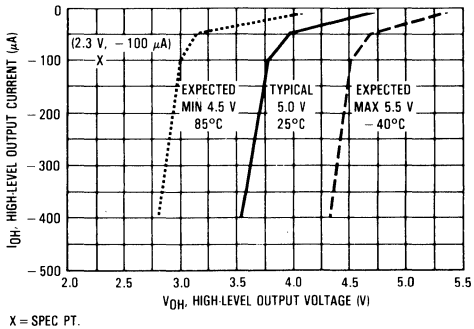
X = SPEC PT.

Figure 16. Typical V_{OL} vs I_{OL} for Port B



X = SPEC PT.

Figure 17. Typical V_{OH} vs I_{OH} for Port B



X = SPEC PT.

Figure 18. Typical V_{OH} vs I_{OH} for Port B with CMOS Pullups

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola using the following media:

MDOS™, disk file
MS™-DOS/PC-DOS disk file (360K)
EPROM(s) 2516, 2716, 2532, 2732

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, sales person, or Motorola representative.

FLEXIBLE DISKS

Several types of flexible disks (MDOS or MS-DOS/PC-DOS disk file) may be submitted for pattern generation. They should be programmed with the customer program, using positive logic sense for address and data. The diskette should be clearly labeled with the customer's name, date, project or product name, and the filename containing the pattern.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to expedite the process in case of any difficulty with the pattern file.

MDOS Disk File

MDOS is Motorola's Disk Operating System available on the EXORciser® development system. The disk media submitted must be a single-sided, single-density, 8-inch MDOS compatible floppy diskette. The diskette must contain the minimum set of MDOS system files in addition to the pattern file.

The .LO output of the M6804 cross assembler should be furnished. In addition, the file must be produced using the ROLLOUT command, so that it contains the absolute image of the M6804 memory. It is necessary to include the entire memory image of both program and data space. All unused bytes, including those in the user space, must be set to logic zero.

MS-DOS/PC-DOS Disk File

MS-DOS is Microsoft's Disk Operating System. PC-DOS is IBM® Personal Computer Disk Operating System. Disk media submitted must be standard density (360K), double-sided 5 1/4 inch compatible floppy diskette. The diskette must contain the object file code in Motorola's S-record format. The S-record format is a character-based object file format generated by M6804 cross assemblers and linkers on IBM PC style machines.

EPROMS

Four K of EPROM are necessary to contain the entire MC6804J2 program. Two 2516 or 2716 type EPROMs or a single 2532 or 2732 type EPROM can be submitted for pattern generation. The EPROM is programmed with the customer program using positive logic sense for address and data. Submissions on two EPROMs must be clearly marked. All unused bytes, including the user's space, must be set to zero.

If the MC6804J2 MCU ROM pattern is submitted on one 2532 or 2732 EPROM, or on two 2516 or 2716 type EPROMs, memory map addressing is one-for-one. The data space ROM runs from EPROM address $018 to $05F and program space ROM runs from EPROM address $C10 to $FF7, with vectors from $FFC to $FFF.

For shipment to Motorola, EPROMs should be placed in a conductive IC carrier and packed securely. Styrofoam is not acceptable for shipment.

Verification Media

All original pattern media, EPROMs or floppy disks, are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked and the verification form should be completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for the creation of the customer mask. To aid in the verification process, Motorola will program customer supplied blank EPROM(s) or DOS disks from the data file used to create the custom mask.

ROM Verification Units (RVUs)

Ten MCUs containing the customers ROM pattern will be sent for program verification. These units will have been made using the custom mask, but are for the purpose of ROM verification only. For expediency, the MCUs are unmarked, packaged in ceramic, and tested with five volts at room temperature. These RVUs are free with the minimum order quantity, but are not production parts. These RVUs are not guaranteed by Motorola Quality Assurance.

Ordering Information

The following table provides generic information pertaining to the package type and temperature for the MC6804J2. This MCU device is available only in the 20-pin plastic dual-in-line (DIP) package.

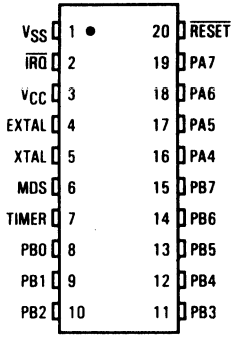
Generic Information

Package Type	Temperature	Order Number
Plastic (P Suffix)	0°C to 70°C −40°C to +85°C	MC6804J2P MC6804J2CP

MDOS is a trademark of Motorola Inc.
MS-DOS is a trademark of Microsoft, Inc.
EXORciser is a registered trademark of Motorola Inc.
IBM is a registered trademark of International Business Machines Corporation.

MECHANICAL DATA

PIN ASSIGNMENTS



MC6804P2

Technical Summary
8-Bit Microcontroller Unit

MC6804P2 HMOS (high-density NMOS) microcontroller unit (MCU) is a member of the M6804 Family of serial processing microcontrollers. This device is extremely versatile and cost effective based on the MCU's simple design and its ability to process 8-bit variables, one bit at a time.

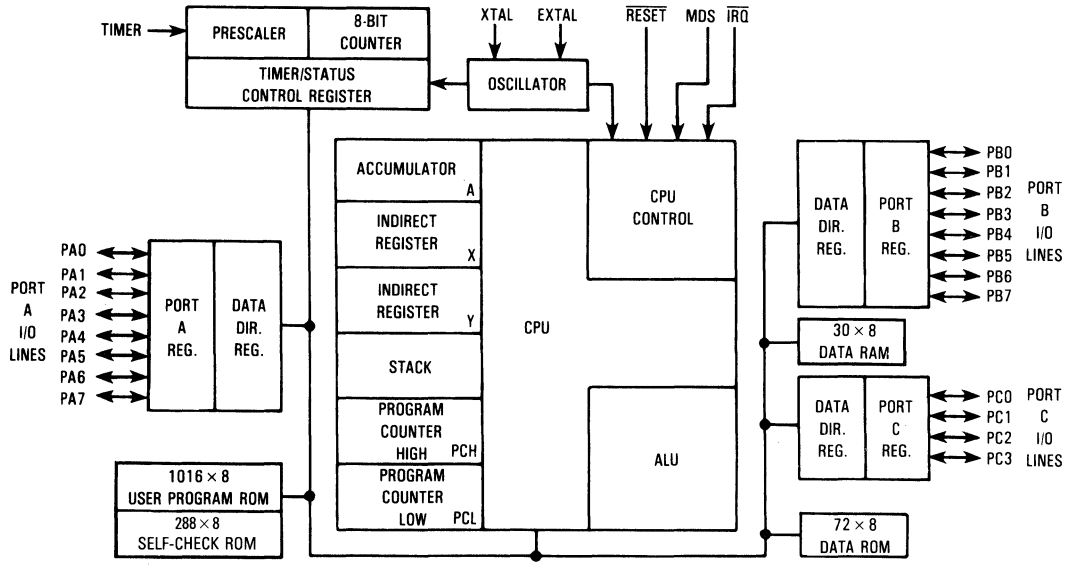
This technical summary contains limited information on the MC6804P2. For detailed information, refer to the advanced information data sheet for the MC6804J1, MC6804J2, MC6804P2 and MC68704P2 8-bit microcontrollers (MC6804 J1/D) or to the *M6804 MCU Manual* (DLE404/D).

Major hardware and software features of the MC6804P2 MCU are:

- On-Chip Clock Generator
- Memory Mapped I/O
- Software Programmable 8-Bit Timer with 7-Bit Prescaler
- Single Instruction Memory Examine/ Change
- 30 Bytes of RAM
- True Bit Manipulation
- Bit Test and Branch Instruction
- 288 Bytes Self-Check ROM
- Conditional Branches
- Timer Pin is Software Programmable as Clock Input or Timer Output
- 1016 Bytes of User Program ROM
- User Selectable Constant Current Pullup Devices available on LSTTL and Open-Drain Interface Ports
- Mask Selectable Edge- or Level-Sensitive Interrupt Pin

3

BLOCK DIAGRAM



SIGNAL DESCRIPTION

V_{CC} AND V_{SS}

Power is supplied to the microcontroller using these two pins. V_{CC} is +5 volts (±0.5 V) power, and V_{SS} is ground.

$\overline{IRQ}$

This pin provides the capability for asynchronously applying an external interrupt to the microcontroller.

EXTAL AND XTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a resistor/capacitor combination, or an external signal is connected to these pins to provide a system clock. Selection is made by a manufacturing mask option. The different clock generator options are shown in Figure 1, along with crystal specifications.

Internal Clock Options

The crystal oscillator start-up time is a function of many variables. To ensure rapid oscillator start-up, neither the crystal characteristics nor load capacitances should exceed recommendations. When using the on-board oscillator, the MCU should remain in a reset condition, with the $\overline{RESET}$ pin voltage below V_{IRES+}, until the oscillator has stabilized at its operating frequency. See Figure 2 for resistor/capacitor oscillator options.

TIMER

The TIMER pin can be configured to operate in either the input or output mode. As input, this pin is connected to the prescaler input and serves as the timer clock. As output, the timer pin reflects the contents of the DOUT bit of the timer status/control register, the last time the TMZ bit was logic high.

$\overline{RESET}$

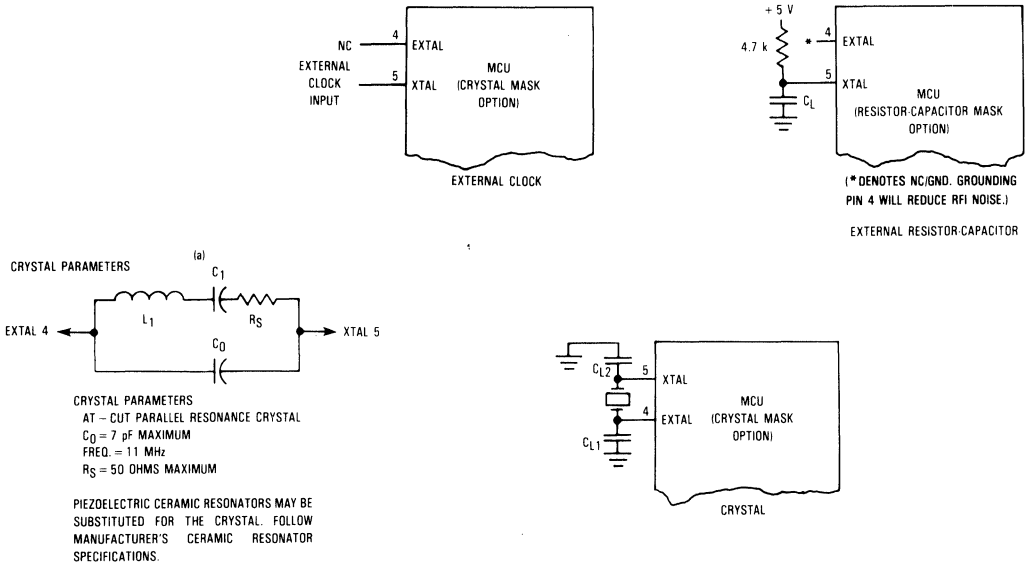
The $\overline{RESET}$ pin is used to restart the processor to the beginning of a program. The program counter is loaded with the address of the restart vector. This should be a jump instruction to the first instruction of the main program. Together with the MDS pin, the $\overline{RESET}$ pin selects the operating mode of the MCU.

MDS

The mode select (MDS) pin places the MCU into special operating modes. When this pin is logic high at the exit of the reset state, the decoded state of PA6 and PA7 is latched to determine the operating mode. This choice can be either the single-chip, self-check, or ROM-verify mode. However, if MDS is logic low at the end of the reset state, the single-chip operating mode is automatically selected. No external diodes, switches, transistors, etc. are required for single-chip mode selection.

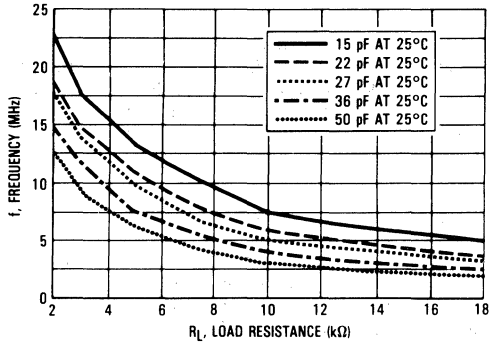
INPUT/OUTPUT LINES (PA0-PA7, PB0-PB7, PC0-PC3)

These 20 lines are arranged into two 8-bit ports (A and B) and one 4-bit port (C). All lines are programmable as

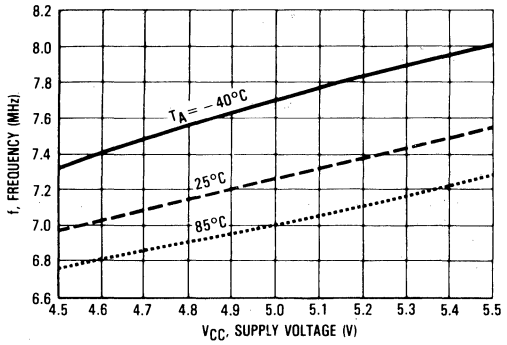


NOTE: Keep crystal leads and circuit connections as short as possible.

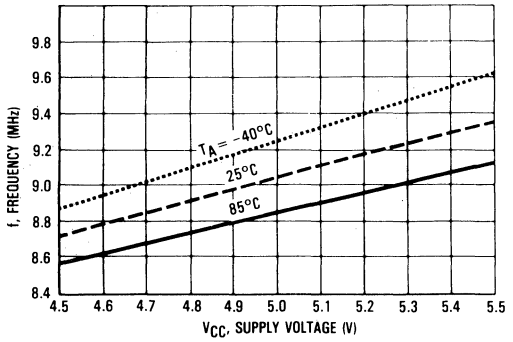
Figure 1. Clock Generator Options and Crystal Parameters



(a) TYPICAL FREQUENCY VS RESISTANCE



(b) TYPICAL FREQUENCY VARIATIONS @ $C_L = 15$ pF, 10 kΩ



(c) TYPICAL FREQUENCY VARIATIONS @ $C_L = 50$ pF, 3 kΩ

Figure 2. Typical Frequency Selection for Resistor/Capacitor Oscillator Options

either inputs or outputs under software control of the data direction registers.

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

There are 20 input/output pins. All pins of each port are programmable as inputs or outputs under the control of the data direction registers (DDR).

The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic one for output, or a logic zero for input, as shown in Figure 3. When the registers are programmed as outputs, the latched data is readable regardless of the logic levels at the output pin due to output loading.

All the I/O pins are LSTTL compatible as both inputs and outputs. In addition, all three ports may use either or both of two manufacturing mask options; open drain output, or internal pull-up resistor for CMOS compatibility.

Any write to a port writes to all of its data bits even though the port DDR may be set to input. This can be used as a tool to initialize the data registers and avoid undefined outputs. However, care must be exercised when using read-modify-write instructions. The data read corresponds to the pin level if the DDR is an input or to the latched output data when the DDR is an output.

The 20 bidirectional lines may be configured by port to be the standard configuration (LSTTL), or either mask option; LSTTL/CMOS, or open drain. Port B outputs are LED compatible.

Port Data Registers ($00, $01, $02)

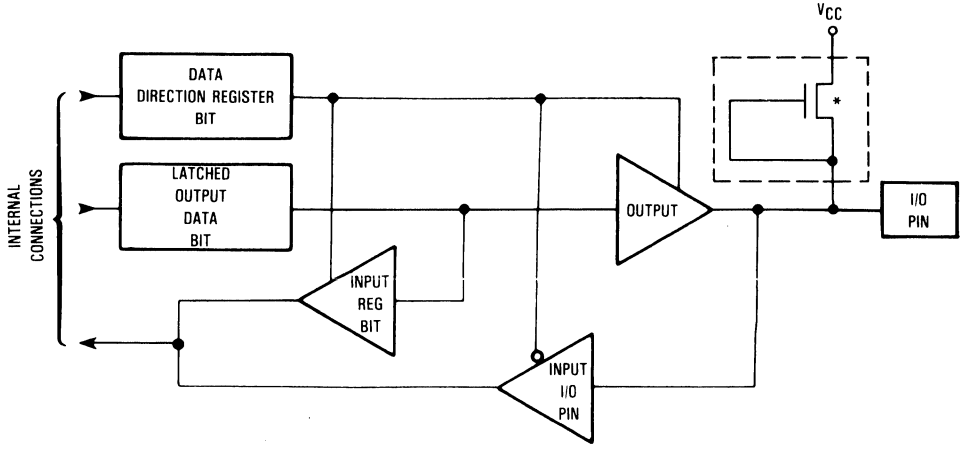
The port data registers are not initialized on reset. These registers should be initialized before changing the DDR bits to avoid undefined levels.

Port A ($00) and Port B ($01)

7	6	5	4	3	2	1	0

Port C ($02)

7	6	5	4	3	2	1	0
X	X	X	X				



DATA DIRECTION REGISTER BIT	OUTPUT DATA BIT	OUTPUT STATE	INPUT TO MCU
1	0	0	0
1	1	1	1
0	X	HI-Z	PIN

*For CMOS option transistor acts as resistor (approximately 40 kΩ) to V_{CC}.
For LSTTL/open-drain options transistor acts as low current clamping diode to V_{CC}.

Figure 3. Typical I/O Port Circuitry

With regard to Port C only, the four MSB bits are unused. These bits are "don't care" (X) bits when written to but are always logic high when read.

Port Data Direction Registers ($04, $05, $06)

Port DDRs configure the port pins as either outputs or inputs. Each port pin can be programmed individually to be an input or an output. A zero in the pin's corresponding DDR bit programs it as an input; a logic one programs it as an output. On reset, all the DDRs are initialized to a logic zero state to put the ports in the input mode.

Port A ($04) and Port B ($05)

7	6	5	4	3	2	1	0

Port C ($06)

7	6	5	4	3	2	1	0
X	X	X	X				

With regard to Port C only, the four MSB bits are unused. These bits are "don't care" (X) bits when written to but are always logic high when read.

MEMORY

The MCU memory map (Figure 4) consists of 4352 bytes of addressable memory, I/O register locations, and four levels of stack space. This MCU has three separate memory spaces: program space, data space, and stack space.

The MCU is capable of addressing 4096 bytes of program space memory with its program counter and 256 bytes of data space memory with its instructions. Program space memory includes self-check ROM, program ROM, self-check and user program vectors, and reserved memory locations.

A non-accessible subroutine stack space RAM is provided. This stack space consists of a last-in-first-out (LIFO) register. This register is used with inherent addressing to stack the return address for subroutines and interrupts.

Indirect X and Y register locations $80 and $81 are generally used as pointers for such tasks as indirect addressing to data space locations. Short direct addressing allows access to the four data space addresses $80-$83 with single byte opcodes. The operations allowed are increment, decrement, load, and store. Data space locations $82 and $83 can be used for 8-bit counter locations.

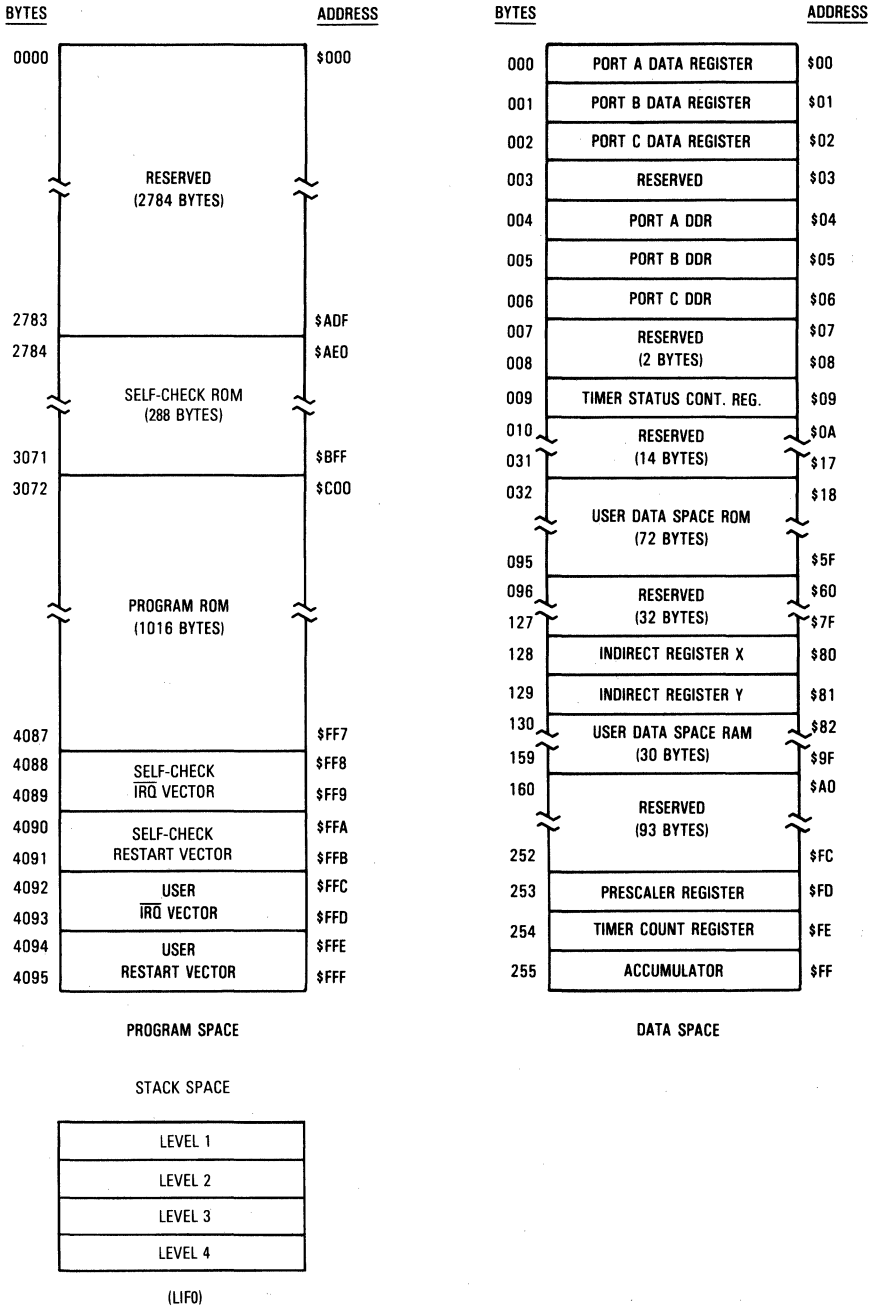
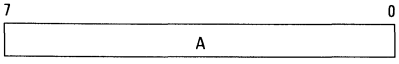


Figure 4. Memory Map

REGISTERS

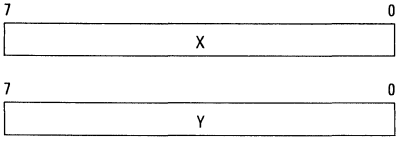
ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



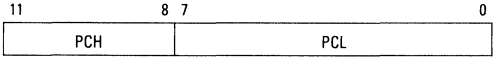
INDIRECT REGISTERS (X,Y)

These two registers are used to maintain pointers to other memory locations in data space. They are used in the register-indirect addressing mode and can be accessed with the direct, indirect, short direct, or bit set/clear modes.



PROGRAM COUNTER (PC)

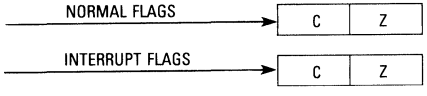
The program counter is a 12-bit register that contains the address of the next byte to be fetched. The program counter is contained in low byte (PCL) and high nibble (PCH).



FLAGS (C,Z)

The first flag, the carry (C) bit, is set on a carry or borrow out of the arithmetic logic unit (ALU). It is cleared if the arithmetic operation does not result in a carry or borrow. The C bit is also set to the value of the bit tested in a bit test instruction. It participates in the rotate left (ROLA) instruction, as well.

The second flag, the zero (Z) bit, is set if the result of the last arithmetic or logic operation was equal to zero. Otherwise, it is cleared. Bit test instructions do not affect the Z bit.



There are two sets of these flags. One set is for interrupt processing (interrupt mode flags). The other set is for normal operations (program mode flags). When an interrupt occurs, a context switch is made from the program flags to the interrupt flags. An RTI forces the context switch back. While in either mode, only the flags for that mode are available. A context switch does not affect the value of the C or Z bits. Both sets of flags are cleared by $\overline{\text{RESET}}$.

STACK

A last-in-first-out (LIFO) stack is incorporated in the MCU that eliminates the need for a stack pointer. This non-accessible subroutine stack space is implemented in separate RAM, 12-bits wide. Whenever a subroutine call or interrupt occurs, the contents of the PC are shifted into the top register of the stack. At the same time, the top register is shifted one level deeper. This happens to all registers, with the bottom register falling out of the stack.

Whenever a return from subroutine or interrupt occurs, the top register is shifted into the PC and all lower registers are shifted one level higher. The stack RAM is four levels deep. If the stack is pulled more than four times with no pushes, then the address that was stored in the bottom level of the stack is shifted into the PC.

SELF CHECK

The MCU implements two forms of internal check: self check and ROM verify. Self check performs an extensive functional check of the MCU using a signature analysis technique. ROM verify uses a similar method to check the contents of program ROM.

Self-check mode is selected by holding the MDS and PA7 pins logic high and the PA6 pin logic low as $\overline{\text{RESET}}$ goes low to high. ROM verify mode is entered by holding MDS, PA7, and PA6 logic high as $\overline{\text{RESET}}$* goes low to high. Unimplemented program space ROM locations are also tested. Monitoring the self-check mode's stages for successful completion requires external circuitry, see *M6804 MCU Manual* (DLE404/D).

RESET

$\overline{\text{RESET}}$

All resets of the MC6804P2 are caused by the external reset input ($\overline{\text{RESET}}$). A reset can be achieved by pulling the $\overline{\text{RESET}}$ pin to logic low for a minimum of 96 oscillator cycles.

During reset, a delay of 96 oscillator cycles is needed before allowing the $\overline{\text{RESET}}$ input to go high. If power is being applied, $\overline{\text{RESET}}$ must be held low long enough for the oscillator to stabilize and then provide the 96 clocks. Connecting a capacitor and resistor to the $\overline{\text{RESET}}$ input, as shown in Figure 5 below typically provides sufficient delay.

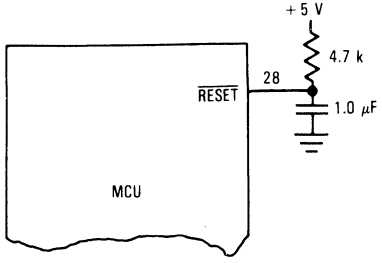


Figure 5. Powerup $\overline{\text{RESET}}$ Delay Circuit

INTERRUPT

The MCU can be interrupted by applying a logic low signal to the $\overline{IRQ}$ pin. However, a manufacturing mask option determines whether the falling edge or the actual low level of the $\overline{IRQ}$ pin is sensed to indicate an interrupt.

EDGE-SENSITIVE OPTION

When the $\overline{IRQ}$ pin is pulled low, the internal interrupt request latch is set. Prior to each instruction fetch, this interrupt request latch is tested. If its output is low, an interrupt sequence is initiated at the end of the current instruction, provided the interrupt mask is cleared. Figure 6 contains a flowchart that illustrates both the reset and interrupt sequences.

The interrupt sequence consists of one cycle during which:

the interrupt request latch is cleared;
the interrupt mode flags are selected;
the program counter (PC) is saved on the stack;
the interrupt mask is set; and
the $\overline{IRQ}$ vector jump address is loaded into the PC.

The $\overline{IRQ}$ vector jump address is $FFC-$FFD in the single-chip mode and $FF8-$FF9 in the self-check mode. The contents of these locations are not decoded as an address to which the PC should jump. Instead, they are decoded like any other ROM word. So, it is essential that the vector contents specify a JMP instruction in addition to the starting address of the interrupt service routine. If required, this routine should save the values of the accumulator and the X and Y registers, since these values are not stored on the stack.

Internal processing of the interrupt continues until a return from interrupt (RTI) instruction is processed. During RTI the interrupt mask is cleared and the program mode flags are selected. The next instruction of the program is then fetched and executed.

When the interrupt was initially detected and the interrupt sequence started, the interrupt request latch was cleared so that the next interrupt could be detected. These steps occurred even as the first interrupt was being serviced. However, even though the second interrupt edge set the interrupt request latch during the first interrupt's processing, the second interrupt's sequence can not begin until completion of the interrupt service routine for

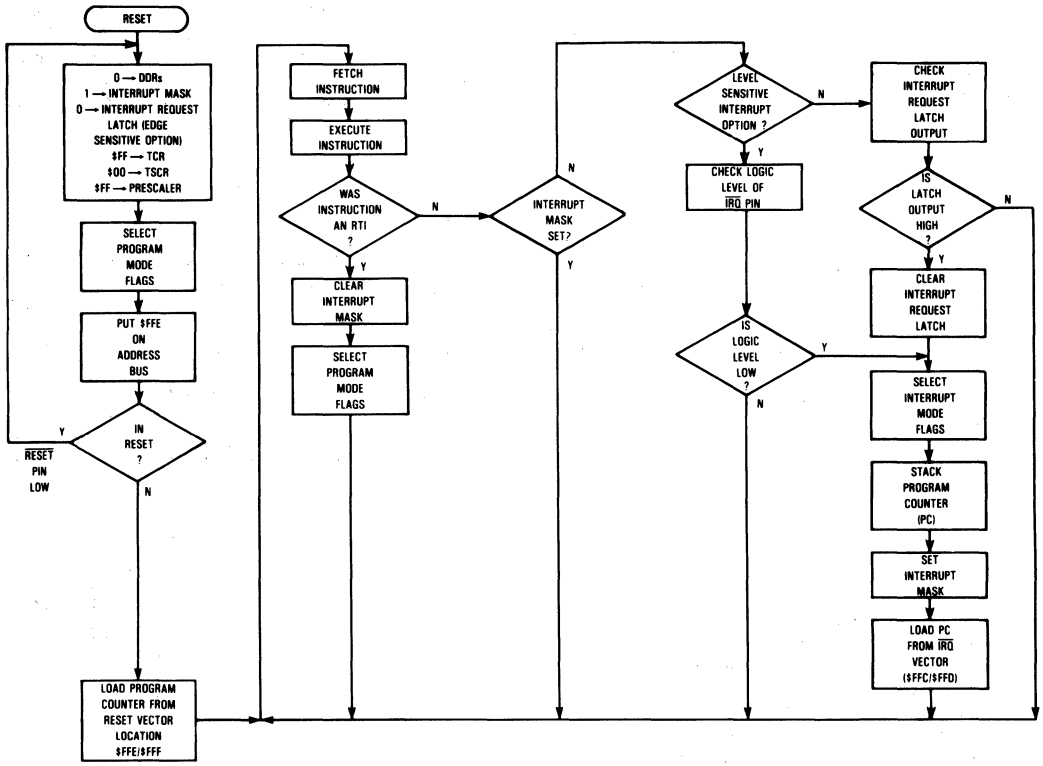


Figure 6. Reset and Interrupt Flowchart

the first interrupt. Completion of an interrupt service routine is always accomplished using an RTI instruction to return to the main program. The interrupt mask, which is not directly available to the programmer, is cleared during the last cycle of the RTI instruction.

LEVEL-SENSITIVE OPTION

Actual operation of the level-sensitive and edge-sensitive options are similar. However, the level-sensitive option does not have an interrupt request latch. Since there is no interrupt request latch, the logic level of the $\overline{IRQ}$ pin is checked to detect the interrupt. Also, in the interrupt sequence there is no need to clear the interrupt request latch. These differences are shown in Figure 6.

POWERUP AND TIMING

During the powerup sequence, the interrupt mask is closed. This precludes any false interrupts. The PC is also loaded with the appropriate restart vector (jump instruction).

To open the interrupt mask, the user should do a JSR to an initialization subroutine that ends with an RTI instead of an RTS. The RTI opens the interrupt mask. Typical $\overline{RESET}$ and $\overline{IRQ}$ processes and their relationship to the interrupt mask are shown in Figure 7.

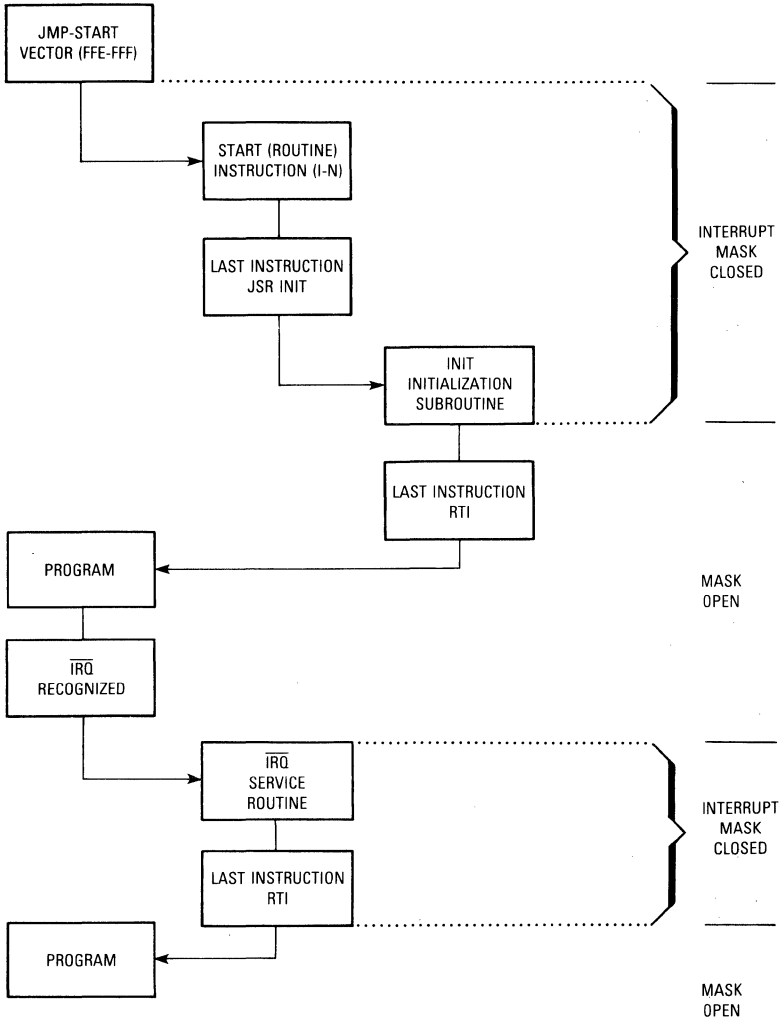


Figure 7. Interrupt Mask

Maximum interrupt response time is six machine cycles. This includes five cycles for the longest instruction plus one for stacking the PC and switching flags.

TIMER

A block diagram of the MC6804P2 timer circuitry is shown in Figure 8. The timer logic in the MCU is comprised of a simple 8-bit counter called the timer counter. This counter is decremented by a 7-bit prescaler at a rate determined by the timer status/control register (TSCR).

PRESCALER

The prescaler is a 7-bit counter used to extend the maximum interval of the overall timer. This counter is clocked by a signal from the TIMER pin or by the internal sync pulse. It divides the frequency received by some factor to create the prescaler output. The factor by which the TIMER pin signal is divided is called the prescaler tap. The value of this tap is selected by three bits of the TSCR (PS0-PS2). These bits control the division of the prescaler input within the range of divide-by-2^0, to divide-by-2^7.

TIMER COUNTER

The timer counter, which may be read or loaded under program control, is decremented from a maximum value of 256 toward zero by the prescaler output. Both are decremented on rising clock edges.

The prescaler register and timer count register are readable and writeable. A write to either one will take precedence over the normal counter function. For example, if a value is written to the timer count register, and this write and a decrement-to-zero occur at the same time, the write takes precedence. TSCR bit one (TMZ) is not set until the next timer time out.

TIMER PIN

The TIMER pin may be programmed as either an input or an output. Its status depends on the value of TSCR bit 5 (TOUT). This relationship is shown in the TIMER pin status section of Figure 8. The frequency of the internal clock applied to the TIMER pin must be less than t_{byte}, which is (f_{osc}/48).

TIMER INPUT MODE

In the timer input mode, TOUT is logic zero and the TIMER pin is connected directly to prescaler input. So, the prescaler is clocked by the signal from the TIMER pin. The prescaler divides the TIMER pin clock input by the prescaler tap. The prescaler output then clocks the 8-bit timer count register. When this register is decremented to zero, it sets TSCR bit one (TMZ). This TMZ bit can be tested under program control to tell when the counter register has reached zero.

TIMER OUTPUT MODE

In the output mode, the TIMER pin is output. TOUT is a logic one. The prescaler is clocked by the internal sync pulse. This pulse is a divide-by-48 of the internal oscillator (f_{osc}/48). From this point on, operation is similar to that described for the input mode. However, in the output mode, once the prescaler decrements the timer counter to zero, the high TMZ bit state allows TSCR bit 4 (DOUT) to become direct input to the TIMER pin.

NOTE

TMZ is normally set to logic one when TCR decrements to zero and the timer times out. However, it may be set by a write of $00 to the timer counter or by a write to bit 7 of TSCR.

TIMER COUNT REGISTER ($FE)

The timer count register reflects the current count in the internal 8-bit counter. The register is the counter and can be read or written.

7							0
MSB							LSB
RESET:							
1	1	1	1	1	1	1	1

TIMER STATUS/CONTROL REGISTER (TSCR) ($09)

7	6	5	4	3	2	1	0
TMZ	—	TOUT	DOUT	$\overline{PSI}$	PS2	PS1	PS0
RESET:							
0	0	0	0	0	0	0	0

TMZ — Timer Zero
- 1 = Timer count register has decremented to zero since the last time the TMZ bit was read.
- 0 = This bit is cleared by a read of the TSCR if TMZ is read as logic one.

Bit 6
- Not used by this register.

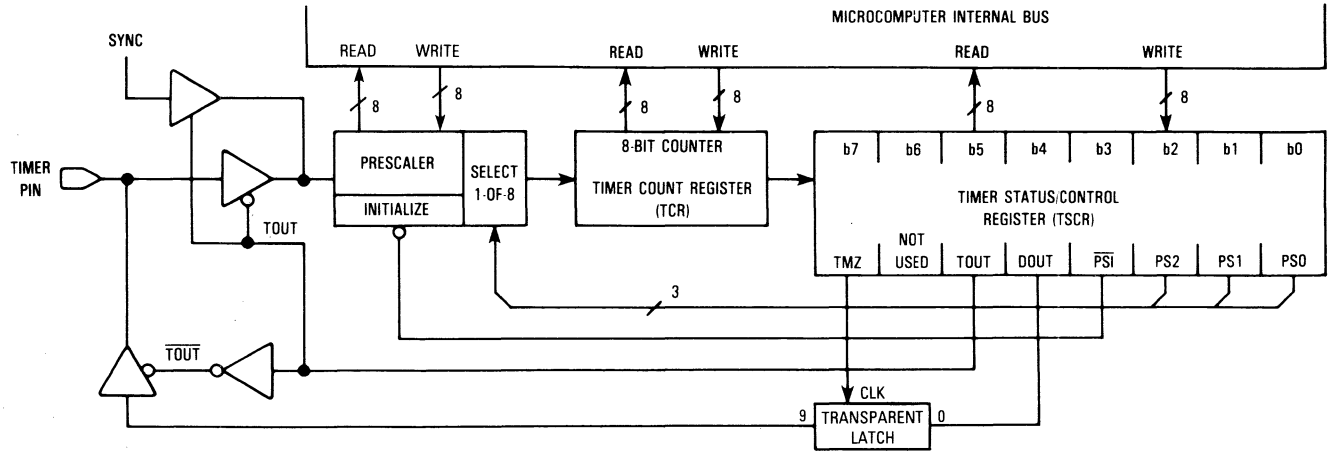
TOUT — Timer Output
- 1 = Output mode is selected for the timer.
- 0 = Input mode is selected for the timer.

DOUT — Data Output
- Latched data at this bit is sent to the TIMER pin when both the TMZ and TOUT bits are logic high.

$\overline{PSI}$ — Prescaler Initialize
- 1 = Prescaler begins to decrement.
- 0 = Prescaler is initialized and counting is inhibited.

PS0-PS2
- These bits are used to select the prescaler tap. The coding of the bits is shown below:

PS2	PS1	PS0	Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128



TIMER PIN STATUS

TOUT	PRESCALER CLOCK	TIMER PIN
0	TIMER PIN	INPUT MODE
1	SYNC	OUTPUT MODE

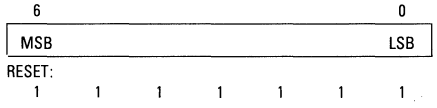
Figure 8. Timer Block Diagram

It is recommended that MVI or load immediate and storing instructions be used when changing bit values in the TSCR. Read-modify-write instructions can cause the TMZ to assume an unexpected state.

During reset, the TSCR is set to all zeros; the TIMER pin is in the high impedance input mode; and DOUT LATCH is forced to a logic high. At the same time, PS0-PS2 coding sets the prescaler tap at divide-by-one, and bit 3 initializes the prescaler.

TIMER PRESCALER REGISTER ($FD)

The timer prescaler register reflects the current count of the 7-bit prescaler. This register is the prescaler counter and can be read or written.



3

INSTRUCTION SET

The MCU has a set of 42 basic instructions. They can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is the accumulator; the other is obtained from memory using one of the addressing modes. Refer to the following list of instructions.

Function	Mnemonic
Load A from Memory	LDA
Load XP from Memory	LDX
Load YP from Memory	LDY
Store A in Memory	STA
Add to A	ADD
Subtract from A	SUB
AND Memory to A	AND
Transfer A to XP	TAX
Transfer A to YP	TAY
Transfer YP to A	TYA
Transfer XP to A	TPA
Clear A	CLRA
Clear XP	CLRX
Clear YP	CLRY
Arithmetic Compare with Memory	CMP
Move Immediate Value to Memory	MVI
Arithmetic Left Shift of A	ASLA
Complement A	COMA
Rotate A Left and Carry	ROLA

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. All INC and DEC forms along with all bit manipulation instructions use this method. Refer to the following list of instructions.

Function	Mnemonic
Increment Memory Location	INC
Increment A	INCA
Increment XP	INCX
Increment YP	INCY
Decrement Memory Location	DEC
Decrement A	DECA
Decrement XP	DECX
Decrement YP	DECY

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two-byte instructions. Refer to the following list of instructions.

Function	Mnemonic
Branch if Carry Clear	BCC
Branch if Higher or Same	(BHS)
Branch if Carry Set	BCS
Branch if Lower	(BLO)
Branch if Not Equal	BNE
Branch if Equal	BEQ

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the 256 bytes of data space where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear, and bit test and branch functions are all implemented with a single instruction. For the test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list of instructions.

Function	Mnemonic
Branch If Bit n is Set	BRSET n(n=0 . . . 7)
Branch If Bit n is Clear	BRCLR n(n=0 . . . 7)
Set Bit n	BSET n(n=0 . . . 7)
Clear Bit n	BCLR n(n=0 . . . 7)

CONTROL INSTRUCTIONS

These instructions are used to control processor operation during program execution. The jump conditional (JMP) and jump to subroutine (JSR) instructions have no

register operand. Refer to the following list of instructions.

Function	Mnemonic
Return from Subroutine	RTS
Return from Interrupt	RTI
No Operation	NOP
Jump to Subroutine	JSR
Jump Unconditional	JMP

IMPLIED INSTRUCTIONS

Since the accumulator and all other registers are located in RAM, many implied instructions exist. Some of the instructions recognized and translated by the assembler are shown below:

Mnemonic	Becomes	Mnemonic	Becomes
ASLA	ADD $FF	INCX	INC $80
BHS	BCC	INCY	INC $81
BLO	BCS	LDXI	MVI $80 DATA
CLRA	SUB $FF	LDYI	MVI $81 DATA
CLRX	MVI $80 #0	NOP	BEQ (PC) +1
CLRY	MVI $81 #0	TAX	STA $80
DECA	DEC $FF	TAY	STA $81
DECX	DEC $80	TXA	LDA $80
DECY	DEC $81	TYA	LDA $81
INCA	INC $FF		

Some examples of valuable instructions not specifically recognized by the assembler are shown below:

Mnemonic	Meaning
BCLR 7,$FF	Ensures A is plus
BSET 7, $FF	Ensures A is minus
BRCLR 7, $FF	Branch if A is plus
BRSET 7, $FF	Branch if A is minus
BRCLR 7, $80	Branch if X is plus (BXPL)
BRSET 7, $80	Branch if X is minus (BXMI)
BRCLR 7, $81	Branch if Y is plus (BYPL)
BRSET 7, $81	Branch if Y is minus (BYMI)

OPCODE MAP

Table 1 is a listing of all the instruction set opcodes applicable to the MC6804P2 MCU.

ADDRESSING MODES

The MCU has nine different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. It deals with objects in three different address spaces: program space, data space, and stack space. The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is located in program ROM. It is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution, such as a constant used to initialize a loop counter.

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the 256 bytes of data space with a single two-byte instruction.

SHORT DIRECT

In the short direct addressing mode, the MCU has four locations in data space RAM it can use, ($80, $81, $82, and $83). The opcode determines the data space RAM location, and the instruction is only one byte. Short direct addressing is a subset of the direct addressing mode. The X and Y registers are at locations $80 and $81, respectively.

EXTENDED

In the extended addressing mode, the effective address of the argument is obtained by concatenating the four least-significant bits of the opcode with the byte following the opcode to form a 12-bit address. Instructions using the extended addressing mode, such as JMP or JSR, are capable of branching anywhere in program space. An extended addressing mode instruction is two bytes long.

RELATIVE

The relative addressing mode is only used in conditional branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −15 to +16 from the opcode address. The programmer need not calculate the offset when using the Motorola assembler, since it calculates the proper offset and checks to see that it is within the span of the branch.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte in which the specified bit is to be set or cleared. Thus, any bit in the 256 locations of data space memory that can be written to can be set or cleared with a single two-byte instruction.

CAUTION

The corresponding DDRs for ports A, B, and C are write only registers (registers at $04, $05, and $06).

Table 1. Opcode Map

Hi / Low	Branch Instructions								Register/Memory, Control, and Read/Modify/Write Instructions				Bit Manipulation Instructions		Register/Memory and Read/Modify/Write		Hi / Low
	0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111	8 1000	9 1001	A 1010	B 1011	C 1100	D 1101	E 1110	F 1111	
0 0000	2 1 BNE REL	2 1 BNE REL	2 1 BEQ REL	2 1 BEQ REL	2 1 BCC REL	2 1 BCC REL	2 1 BCS REL	2 1 BCS REL	4 2 JSRn EXT	4 2 JMPn EXT	*	4 3 MVI IMM	5 3 BRCLR0 B T B	4 2 BCLR0 BSC	4 1 LDA R IND	4 1 LDA R IND	0 0000
1 0001	2 1 BNE REL	2 1 BNE REL	2 1 BEQ REL	2 1 BEQ REL	2 1 BCC REL	2 1 BCC REL	2 1 BCS REL	2 1 BCS REL	4 2 JSRn EXT	4 2 JMPn EXT	*	*	5 3 BRCLR1 B T B	4 2 BCLR1 BSC	4 1 STA R IND	4 1 STA R IND	1 0001
2 0010	2 1 BNE REL	2 1 BNE REL	2 1 BEQ REL	2 1 BEQ REL	2 1 BCC REL	2 1 BCC REL	2 1 BCS REL	2 1 BCS REL	4 2 JSRn EXT	4 2 JMPn EXT	*	2 1 RTI INH	5 3 BRCLR2 B T B	4 2 BCLR2 BSC	4 1 ADD R IND	4 1 ADD R IND	2 0010
3 0011	2 1 BNE REL	2 1 BNE REL	2 1 BEQ REL	2 1 BEQ REL	2 1 BCC REL	2 1 BCC REL	2 1 BCS REL	2 1 BCS REL	4 2 JSRn EXT	4 2 JMPn EXT	*	2 1 RTS INH	5 3 BRCLR3 B T B	4 2 BCLR3 BSC	4 1 SUB R IND	4 1 SUB R IND	3 0011
4 0100	2 1 BNE REL	2 1 BNE REL	2 1 BEQ REL	2 1 BEQ REL	2 1 BCC REL	2 1 BCC REL	2 1 BCS REL	2 1 BCS REL	4 2 JSRn EXT	4 2 JMPn EXT	*	4 1 COMA INH	5 3 BRCLR4 B T B	4 2 BCLR4 BSC	4 1 CMP R IND	4 1 CMP R IND	4 0100
5 0101	2 1 BNE REL	2 1 BNE REL	2 1 BEQ REL	2 1 BEQ REL	2 1 BCC REL	2 1 BCC REL	2 1 BCS REL	2 1 BCS REL	4 2 JSRn EXT	4 2 JMPn EXT	*	4 1 ROLA INH	5 3 BRCLR5 B T B	4 2 BCLR5 BSC	4 1 AND R IND	4 1 AND R IND	5 0101
6 0110	2 1 BNE REL	2 1 BNE REL	2 1 BEQ REL	2 1 BEQ REL	2 1 BCC REL	2 1 BCC REL	2 1 BCS REL	2 1 BCS REL	4 2 JSRn EXT	4 2 JMPn EXT	*	*	5 3 BRCLR6 B T B	4 2 BCLR6 BSC	4 1 INC R IND	4 1 INC R IND	6 0110
7 0111	2 1 BNE REL	2 1 BNE REL	2 1 BEQ REL	2 1 BEQ REL	2 1 BCC REL	2 1 BCC REL	2 1 BCS REL	2 1 BCS REL	4 2 JSRn EXT	4 2 JMPn EXT	*	*	5 3 BRCLR7 B T B	4 2 BCLR7 BSC	4 1 DEC R IND	4 1 DEC R IND	7 0111
8 1000	2 1 BNE REL	2 1 BNE REL	2 1 BEQ REL	2 1 BEQ REL	2 1 BCC REL	2 1 BCC REL	2 1 BCS REL	2 1 BCS REL	4 2 JSRn EXT	4 2 JMPn EXT	4 1 INC S D	4 1 DEC S D	5 3 BRSET0 B T B	4 2 BSET0 BSC	4 2 LDA IMM	4 2 LDA DIR	8 1000
9 1001	2 1 BNE REL	2 1 BNE REL	2 1 BEQ REL	2 1 BEQ REL	2 1 BCC REL	2 1 BCC REL	2 1 BCS REL	2 1 BCS REL	4 2 JSRn EXT	4 2 JMPn EXT	4 1 INC S D	4 1 DEC S D	5 3 BRSET1 B T B	4 2 BSET1 BSC	#	4 2 STA DIR	9 1001
A 1010	2 1 BNE REL	2 1 BNE REL	2 1 BEQ REL	2 1 BEQ REL	2 1 BCC REL	2 1 BCC REL	2 1 BCS REL	2 1 BCS REL	4 2 JSRn EXT	4 2 JMPn EXT	4 1 INC S D	4 1 DEC S D	5 3 BRSET2 B T B	4 2 BSET2 BSC	4 2 ADD IMM	4 2 ADD DIR	A 1010
B 1011	2 1 BNE REL	2 1 BNE REL	2 1 BEQ REL	2 1 BEQ REL	2 1 BCC REL	2 1 BCC REL	2 1 BCS REL	2 1 BCS REL	4 2 JSRn EXT	4 2 JMPn EXT	4 1 INC S D	4 1 DEC S D	5 3 BRSET3 B T B	4 2 BSET3 BSC	SUB IMM	4 2 SUB DIR	B 1011
C 1100	2 1 BNE REL	2 1 BNE REL	2 1 BEQ REL	2 1 BEQ REL	2 1 BCC REL	2 1 BCC REL	2 1 BCS REL	2 1 BCS REL	4 2 JSRn EXT	4 2 JMPn EXT	4 1 LDA S D	4 1 STA S D	5 3 BRSET4 B T B	4 2 BSET4 BSC	4 2 CMP IMM	CMP DIR	C 1100
D 1101	2 1 BNE REL	2 1 BNE REL	2 1 BEQ REL	2 1 BEQ REL	2 1 BCC REL	2 1 BCC REL	2 1 BCS REL	2 1 BCS REL	4 2 JSRn EXT	4 2 JMPn EXT	4 1 LDA S D	4 1 STA S D	5 3 BRSET5 B T B	4 2 BSET5 BSC	4 2 AND IMM	4 2 AND DIR	D 1101
E 1110	2 1 BNE REL	2 1 BNE REL	2 1 BEQ REL	2 1 BEQ REL	2 1 BCC REL	2 1 BCC REL	2 1 BCS REL	2 1 BCS REL	4 2 JRSn EXT	4 2 JMPn EXT	4 1 LDA S D	4 1 STA S D	5 3 BRSET6 B T B	4 2 BSET6 BSC	#	4 2 INC DIR	E 1110
F 1111	2 1 BNE REL	2 1 BNE REL	2 1 BEQ REL	2 1 BEQ REL	2 1 BCC REL	2 1 BCC REL	2 1 BCS REL	2 1 BCS REL	4 2 JSRn EXT	4 2 JMPn EXT	4 1 LDA S D	4 1 STA S D	5 3 BRSET7 B T B	4 2 BSET7 BSC	#	4 2 DEC DIR	F 1111

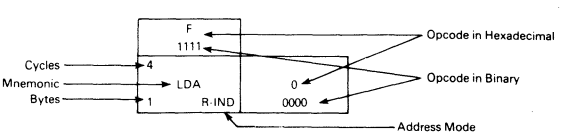
Abbreviations for Address Modes

INH	Inherent
S-D	Short Direct
B-T-B	Bit Test and Branch
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
R-IND	Register Indirect

* Indicates Instruction Reserved for Future Use

\# Indicates Illegal Instruction

LEGEND



A read operation on these registers is undefined. Since BSET and BCLR are read-modify-write functions, they cannot be used to set or clear a DDR bit; all "unaffected" bits would be set. Write all DDR bits in a port using a single-store instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit to be tested, and its condition (set or clear), is included in the opcode. The data space address of the byte to be tested is in the single byte immediately following the opcode byte. The third byte is sign extended to twelve bits and becomes the offset added to the PC if the condition is true. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the 256 locations of data space. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry flag.

REGISTER-INDIRECT

In the register-indirect addressing mode, the operand is at the address in data space pointed to by the contents of one of the indirect registers, X or Y. The particular indirect register is selected by bit 4 of the opcode. Bit 4 decodes into an address that represents the register, $80 or $81. A register-indirect instruction is one byte long.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. These instructions are one byte long.

3

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range (Comm.)	T_A	0 to 70	°C
Operating Temperature Range (Ind.)	T_A	−40 to +85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature Plastic PLCC Cerdip	T_J	 150 150 175	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{CC}$. Reliability of operation is enhanced if unused inputs except EXTAL are connected to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic PLCC Cerdip	θ_{JA}	 70 120 60	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

ELECTRICAL CHARACTERISTICS (V_{CC} = +5.0 Vdc ± 0.5 Vdc, V_{SS} = GND, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Internal Power Dissipation — No Port Loading	P_{INT}	—	120	165	mW
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Input Capacitance	C_{in}	—	10	—	pF
Input Current ($\overline{IRQ}$, $\overline{RESET}$)	I_{in}	—	2	20	μA

SWITCHING CHARACTERISTICS (V_{CC} = +5.0 Vdc ± 0.5 Vdc, V_{SS} = GND, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Oscillator Frequency	f_{osc}	4.0	—	11.0	MHz
Bit Time	t_{bit}	0.364	—	1.0	μs
Byte Cycle Time	t_{byte}	4.36	—	12.0	μs
$\overline{IRQ}$ and TIMER Pulse Width	t_{WL}, t_{WH}	2 × t_{byte}	—	—	—
$\overline{RESET}$ Pulse Width	t_{RWL}	2 × t_{byte}	—	—	—
$\overline{RESET}$ Delay Time (External Capacitance = 1.0 μF)	t_{RHL}	100	—	—	ms

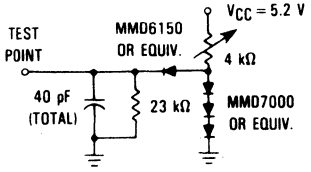


Figure 9. LSTTL Equivalent Test Load (Port B)

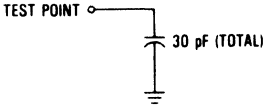


Figure 10. CMOS Equivalent Test Load (Ports A, B, C)

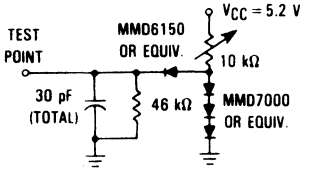


Figure 11. LSTTL Equivalent Test Load (Ports A, C, and TIMER)

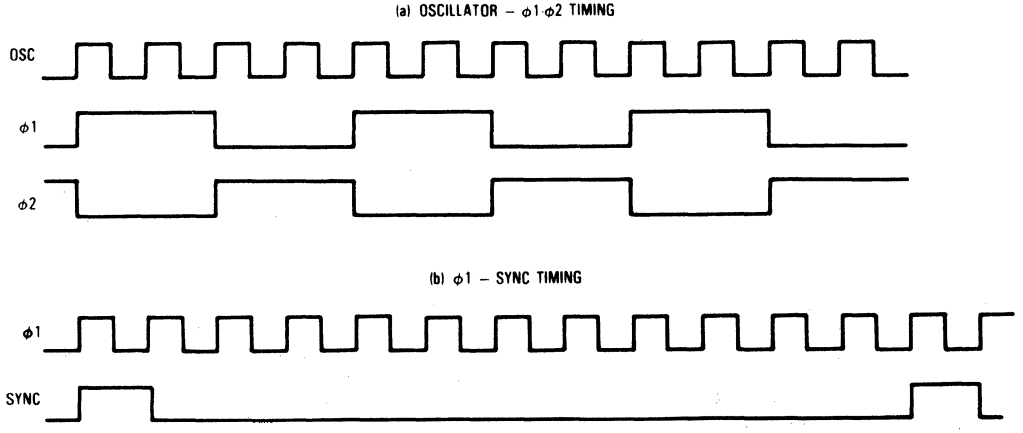


Figure 12. Clock Generator Timing Diagram

PORT DC ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.0 Vdc ± 0.5 Vdc, V_{SS} = GND, T_A = 0° to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Ports A, C, and Timer (Standard)					
Output Low Voltage, I_{Load} = 0.4 mA	V_{OL}	—	—	0.5	V
Output High Voltage, I_{Load} = −50 μA	V_{OH}	2.3	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	4	40	μA
Ports A and C (Open Drain)					
Output Low Voltage, I_{Load} = 0.4 mA	V_{OL}	—	—	0.5	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	4	40	μA
Open Drain Leakage (V_{out} = V_{CC})	I_{LOD}	—	4	40	μA
Ports A and C (CMOS Drive)					
Output Low Voltage, I_{Load} = 0.4 mA (Sink)	V_{OL}	—	—	0.5	V
Output High Voltage, I_{Load} = −10 μA	V_{OH}	V_{CC} − 1.0	—	—	V
Output High Voltage, I_{Load} = −50 μA	V_{OH}	2.3	—	—	V
Input High Voltage, I_{Load} = −300 μA Max	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage, I_{Load} = −300 μA Max	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current (V_{in} = 0.4 V to V_{CC})	I_{TSI}	—	—	−300	μA
Port B (Standard)					
Output Low Voltage, I_{Load} = 1.0 mA	V_{OL}	—	—	0.5	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.5	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.3	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	8	80	μA
Port B (Open Drain)					
Output Low Voltage, I_{Load} = 1.0 mA	V_{OL}	—	—	0.5	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.5	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	8	80	μA
Open Drain Leakage (V_{out} = V_{CC})	I_{LOD}	—	8	80	μA
Port B (CMOS Drive)					
Output Low Voltage, I_{Load} = 1.0 mA	V_{OL}	—	—	0.5	V
Output High Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.5	V
Output High Voltage, I_{Load} = −10 μA	V_{OH}	V_{CC} − 1.0	—	—	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.3	—	—	V
Input High Voltage, I_{Load} = −300 μA Max	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage, I_{Load} = −300 μA Max	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current (V_{in} = 0.4 V to V_{CC})	I_{TSI}	—	—	−300	μA
Ports A, B, and C (Low Current Clamping Diode*)					
Input High Current V_{IH} = V_{CC} + 1.0 V	I_{IH}	—	—	100	μA
Input Low Current V_{IL} = 0.8 V	I_{IL}	—	—	−4.0	μA

*Denotes not tested unless specified on ordering form.

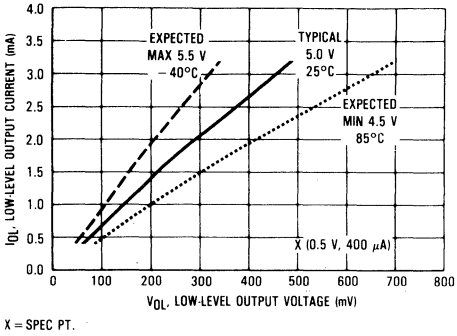


Figure 13. Typical V_{OL} vs I_{OL} for Port A and TIMER

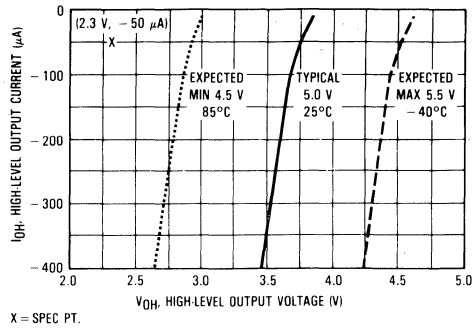


Figure 14. Typical V_{OH} vs I_{OH} for Port A and TIMER

3

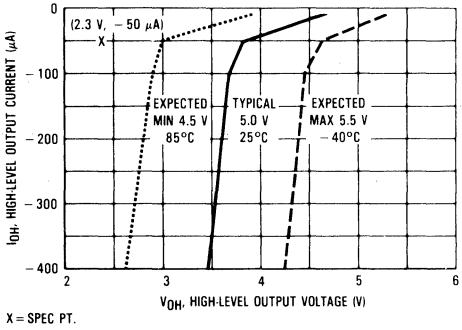


Figure 15. Typical V_{OH} vs I_{OH} for Port A with CMOS Pullups

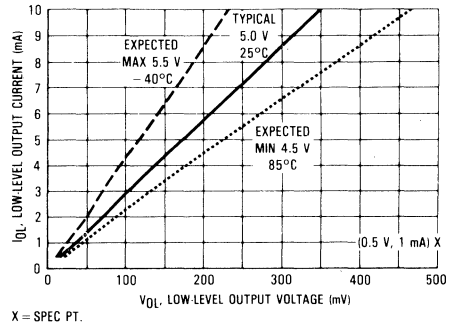


Figure 16. Typical V_{OL} vs I_{OL} for Port B

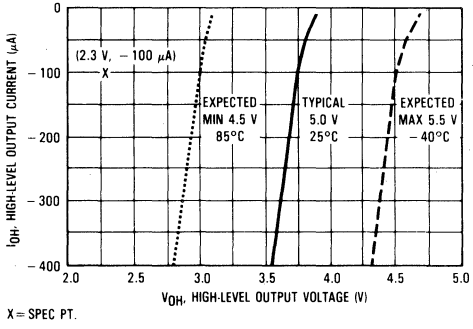


Figure 17. Typical V_{OH} vs I_{OH} for Port B

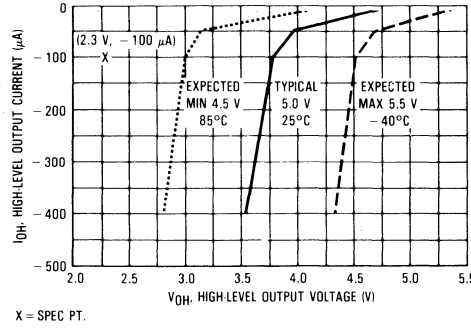


Figure 18. Typical V_{OH} vs I_{OH} for Port B with CMOS Pullups

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola using the following media:

MDOS, disk file
MS-DOS/PC-DOS disk file (360K)
EPROM(s) 2516, 2716, 2532, 2732

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, sales person, or a Motorola representative.

FLEXIBLE DISKS

Several types of flexible disks (MDOS™ or MS™-DOS/PC-DOS disk file) may be submitted for pattern generation. They should be programmed with the customer's program, using positive logic sense for address and data. The diskette should be clearly labeled with the customer's name, date, project or product name, and the filename containing the pattern.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to expedite the process in case of any difficulty with the pattern file.

MDOS Disk File

MDOS is Motorola's Disk Operating System available on the EXORciser® development system. The disk media submitted must be a single-sided, single-density, 8-inch MDOS compatible floppy diskette. The diskette must contain the minimum set of MDOS system files in addition to the pattern file.

The .LO otuput of the M6804 cross assembler should be furnished. In addition, the file must be produced using the ROLLOUT command, so that it contains the absolute image of the M6804 memory. It is necessary to include the entire memory image of both program and data space. All unused bytes, including those in the user space, must be set to logic zero.

MS-DOS/PC-DOS Disk File

MS-DOS is Microsoft's Disk Operating System. PC-DOS is IBM® Personal Computer Disk Operating System. Disk media submitted must be standard density (360K), double-sided 5 1/4 inch compatible floppy diskette. The diskette must contain the object file code in Motorola's S-record format. The S-record format is a character-based object file format generated by M6804 cross assemblers and linkers on IBM PC style machines.

EPROMS

Four K of EPROM are necessary to contain the entire MC6804P2 program. Two 2516 or 2716 type EPROMs or a single 2532 or 2732 type EPROM can be submitted for pattern generation. The EPROM is programmed with the customer's program using positive logic sense for address and data. Submissions on two EPROMs must be clearly marked. All unused bytes, including the user's space, must be set to zero.

If the MC6804P2 MCU ROM pattern is submitted on one 2532 or 2732 EPROM, or on two 2516 or 2716 type EPROMs, memory map addressing is one-for-one. The data space ROM runs from EPROM address $018 to $05F, and program space ROM runs from EPROM address $C00 to $FF7, with vectors from $FFC to $FFF.

For shipment to Motorola, EPROMs should be placed in a conductive IC carrier and packed securely. Styrofoam is not acceptable for shipment.

Verification Media

All original pattern media, EPROMs or floppy disks, are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked and the verification form should be completed, signed and returned to Motorola. The signed verification form constitutes the contractual agreement for the creation of the customer mask. To aid in the verification process, Motorola will program customer supplied blank EPROM(s) or DOS disks from the data file used to create the custom mask.

ROM Verification Units (RVUs)

Ten MCUs containing the customer's ROM pattern will be sent for program verification. These units will have been made using the custom mask, but are for the purpose of ROM verification only. For expediency, the MCUs are unmarked, packaged in ceramic, and tested with five volts at room temperature. These RVUs are free with the minimum order quantity, but are not production parts. These RVUs are not guaranteed by Motorola Quality Assurance.

Ordering Information

The following table provides generic information pertaining to the package type and temperature for the MC6804P2. This MCU device is available in both the 28-pin plastic dual-in-line (DIP) and the 28-lead PLCC package.

Generic Information

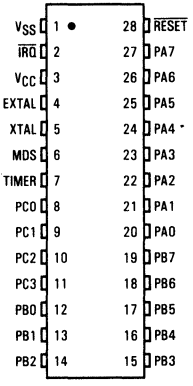
Package Type	Temperature	Order Number
Plastic (P Suffix)	0°C to 70°C −40°C to +85°C	MC6804P2P MC6804P2CP
Plastic Leaded Chip Carrier (FN Suffix)	0°C to 70°C −40°C to +85°C	MC6804P2FN MC6804P2CFN

MDOS is a trademark of Motorola Inc.
MS is a trademark of Microsoft, Inc.
EXORciser is a registered trademark of Motorola Inc.
IBM is a registered trademark of International Business Machines Corporation.

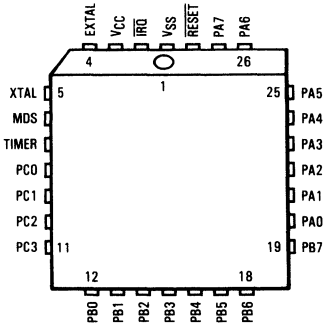
MECHANICAL DATA

PIN ASSIGNMENTS

28-PIN DUAL-IN-LINE PACKAGE



28-LEAD PLCC PACKAGE



MC68704P2

Technical Summary
HMOS Microcomputer Unit

MC68704P2 HMOS (high-density NMOS) microcomputer unit (MCU) is an EPROM member of the M6804 Family of microcomputers. User programmable EPROM allows program changes and lower volume applications. This feature further heightens the versatility of an MCU whose design-ability to process 8-bit variables, one bit at a time, already makes it tremendously cost effective.

This technical summary contains limited information on the MC68704P2. For detailed information, refer to the advanced information data sheet for the MC6804J1, MC6804J2, MC6804P2, and MC68704P2 8-bit microcomputers, (MC6804J1/D) or to the *M6804 MCU Manual* (DLE404/D).

Major hardware and software features of the MC68704P2 MCU are:

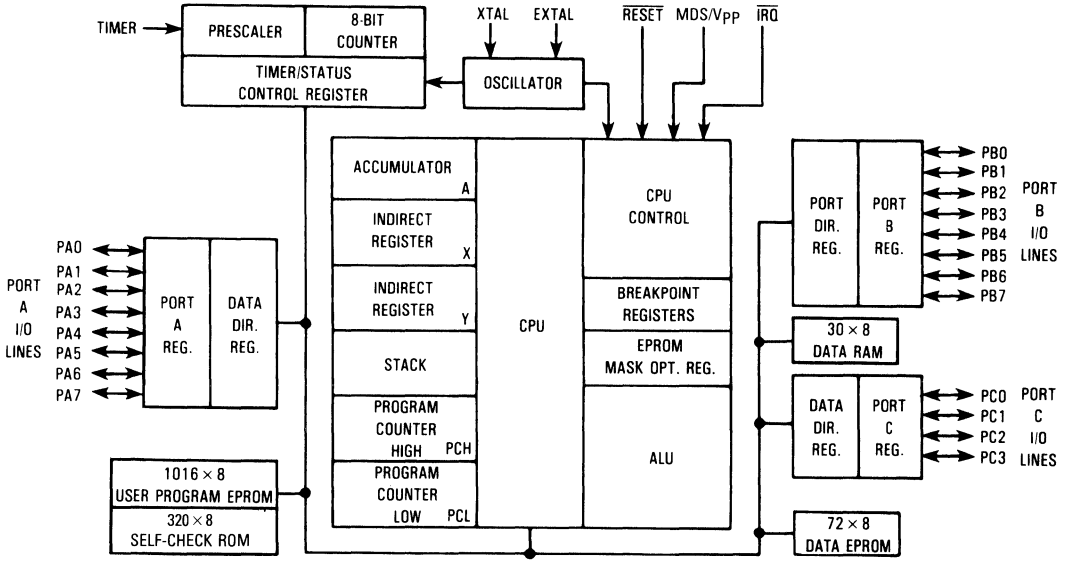
- On-Chip Clock Generator
- I/O and Registers Mapped in Data Space Memory
- Software Programmable 8-Bit Timer with 7-Bit Prescaler
- Single Instruction Memory Examine/ Change
- MC6804J1/J2/P2 Emulation
- 1088 Bytes of EPROM
- True Bit Manipulation
- Bit Test and Branch Instruction
- Breakpoint and Mask Option Registers
- Self-Check
- Conditional Branches
- Timer Pin is Software Programmable as Event Counter or Timer Output
- MC68HC04P2/P3 Pin Compatibility
- 32 Bytes of RAM

User selectable options are:

- Mask Selectable Edge- or Level-Sensitive Interrupt Pin
- Push-Pull or Open-Drain Interface Ports

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

V_{CC} AND V_{SS}

Power is supplied to the microcomputer using these two pins. V_{CC} is +5 volts (±0.5 V) power, and V_{SS} is ground.

$\overline{IRQ}$

This pin provides the capability for asynchronously applying an external interrupt to the microcomputer.

EXTAL AND XTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a resistor/capacitor combination, or an external signal is connected to these pins to provide a system clock. Selection is made through the mask option register (MOR). The different clock generator options are shown in Figure 1, along with crystal specifications.

Internal Clock Options

The crystal oscillator start-up time is a function of many variables. To ensure rapid oscillator start-up, neither the crystal characteristics nor load capacitances should exceed recommendations. When using the on-board oscillator, the MCU should remain in a reset condition, with the $\overline{RESET}$ pin voltage below V_{IRES+}, until the oscillator has stabilized at its operating frequency. See Figure 2 for resistor/capacitor oscillator options.

TIMER

The TIMER pin can be configured to operate in either the input or output mode. As input, this pin is connected to the prescaler input and serves as the timer clock. As output, the timer pin reflects the contents of the DOUT bit of the timer status/control register, the last time the TMZ bit was logic high.

$\overline{RESET}$

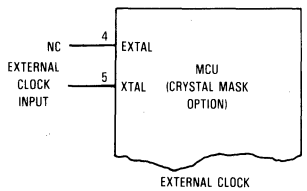
The $\overline{RESET}$ pin is used to restart the processor to the beginning of a program. The program counter is loaded with the address of the restart vector. This should be a jump instruction to the first instruction of the main program. Together with the MDS pin, the $\overline{RESET}$ pin selects the operating mode of the MCU.

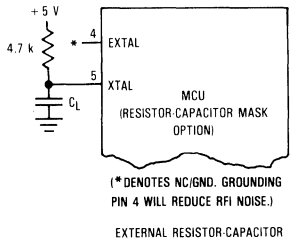
MDS/V_{PP}

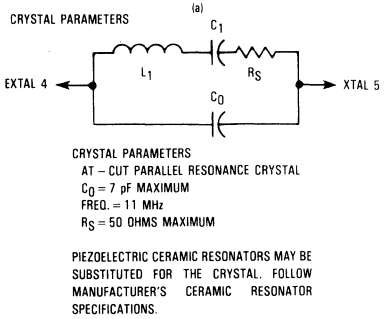
The mode select (MDS) pin places the MCU into special operating modes. When this pin is logic high at the exit of the reset state, the decoded state of PA6 and PA7 is latched to determine the operating mode. This choice can be either the single-chip, self-check, or EPROM programming. However, if MDS is logic low at the end of the reset state, the single-chip operating mode is automatically selected. No external diodes, switches, transistors, etc. are required for single-chip mode selection. This pin is raised to V_{PP} voltage to program the EPROM.

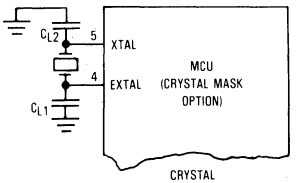
INPUT/OUTPUT LINES (PA0-PA7, PB0-PB7, PC0-PC3)

These 20 lines are arranged into two 8-bit ports (A and B) and one 4-bit port (C). All lines are programmable as



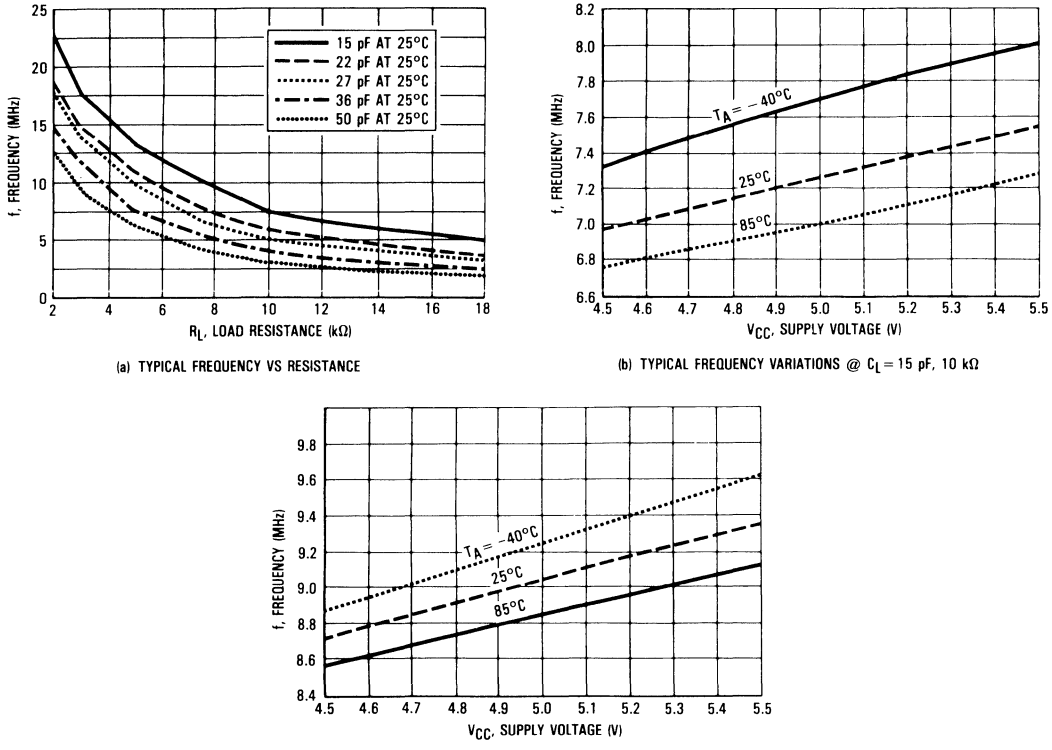






NOTE: Keep crystal leads and circuit connections as short as possible.

Figure 1. Clock Generator Options and Crystal Parameters



(a) TYPICAL FREQUENCY VS RESISTANCE

(b) TYPICAL FREQUENCY VARIATIONS @ $C_L = 15$ pF, 10 kΩ

(c) TYPICAL FREQUENCY VARIATIONS @ $C_L = 50$ pF, 3 kΩ

Figure 2. Typical Frequency Selection for Resistor/Capacitor Oscillator Options

either inputs or outputs under software control of the data direction registers.

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

There are 20 input/output pins. All pins of each port are programmable as inputs or outputs under the control of the data direction registers (DDR).

The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic one for output, or a logic zero for input, as shown in Figure 3. When the registers are programmed as outputs, the latched data is readable regardless of the logic levels at the output pin due to output loading.

All the I/O pins are LSTTL compatible as both inputs and outputs. In addition, all three ports may use either or both of two output options: open drain or push-pull.

Any write to a port writes to all of its data bits even though the port DDR may be set to input. This can be used as a tool to initialize the data registers and avoid undefined outputs. However, care must be exercised when using read-modify-write instructions. The data read corresponds to the pin level if the DDR is an input or to the latched output data when the DDR is an output.

The 20 bidirectional lines may be configured by port to be the standard configuration, push-pull, or open drain. Port B outputs are LED compatible.

Port Data Registers ($00, $01, $02)

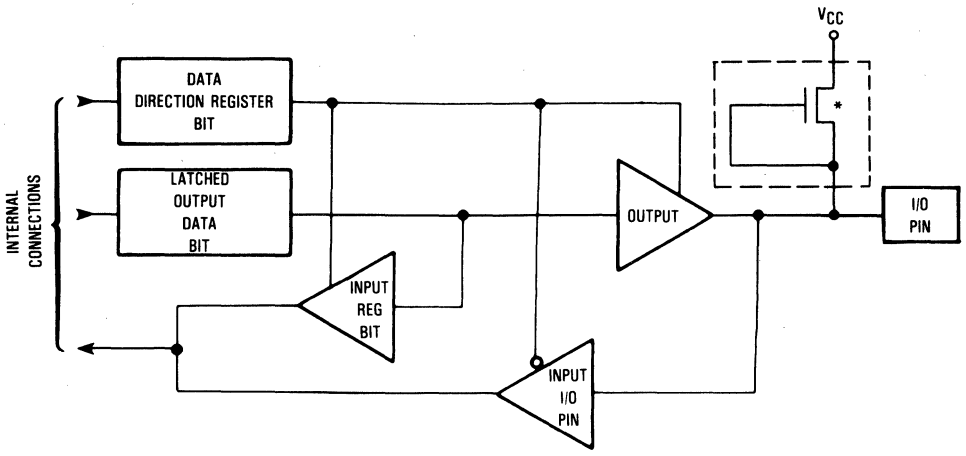
The port data registers are not initialized on reset. These registers should be initialized before changing the DDR bits to avoid undefined levels.

Port A ($00) and Port B ($01)

7	6	5	4	3	2	1	0

Port C ($02)

7	6	5	4	3	2	1	0
X	X	X	X				



DATA DIRECTION REGISTER BIT	OUTPUT DATA BIT	OUTPUT STATE	INPUT TO MCU
1	0	0	0
1	1	1	1
0	X	HI-Z	PIN

*For CMOS option transistor acts as resistor (approximately 40 kΩ) to V_{CC}.
For LSTTL/open-drain options transistor acts as low current clamping diode to V_{CC}.

Figure 3. Typical I/O Port Circuitry

With regard to Port C only, the four MSB bits are unused. These bits are "don't care" (X) bits when written to but are always logic high when read.

Port Data Direction Registers ($04, $05, $06)

Port DDRs configure the port pins as either outputs or inputs. Each port pin can be programmed individually to be an input or an output. A zero in the pin's corresponding DDR bit programs it as an input; a logic one programs it as an output. On reset, all the DDRs are initialized to a logic zero state to put the ports in the input mode.

Port A ($04) and Port B ($05)

7	6	5	4	3	2	1	0

Port C ($06)

7	6	5	4	3	2	1	0
X	X	X	X				

With regard to Port C only, the four MSB bits are unused. These bits are "don't care" (X) bits when written to but are always logic high when read.

MEMORY

The MCU memory map (Figure 4) consists of 4352 bytes of addressable memory and I/O register locations. This MCU has three separate memory spaces: program space, data space, and stack space.

The MCU is capable of addressing 4096 bytes of program space memory with its program counter and 256 bytes of data space memory with its instructions. Program space memory includes self-check ROM, program EPROM, self-check vectors (mask ROM), user program vectors (EPROM), and reserved memory locations.

A non-accessible subroutine stack space RAM is provided. This stack space consists of a last-in-first-out (LIFO) register. This register is used with inherent addressing to stack the return address for subroutines and interrupts.

Indirect X and Y register locations $80 and $81 are generally used as pointers for such tasks as indirect addressing to data space locations. Short direct addressing allows access to the four data space addresses $80-$83 with single byte opcodes. The operations allowed are increment, decrement, load, and store. Data space locations $82 and $83 can be used for 8-bit counter locations.

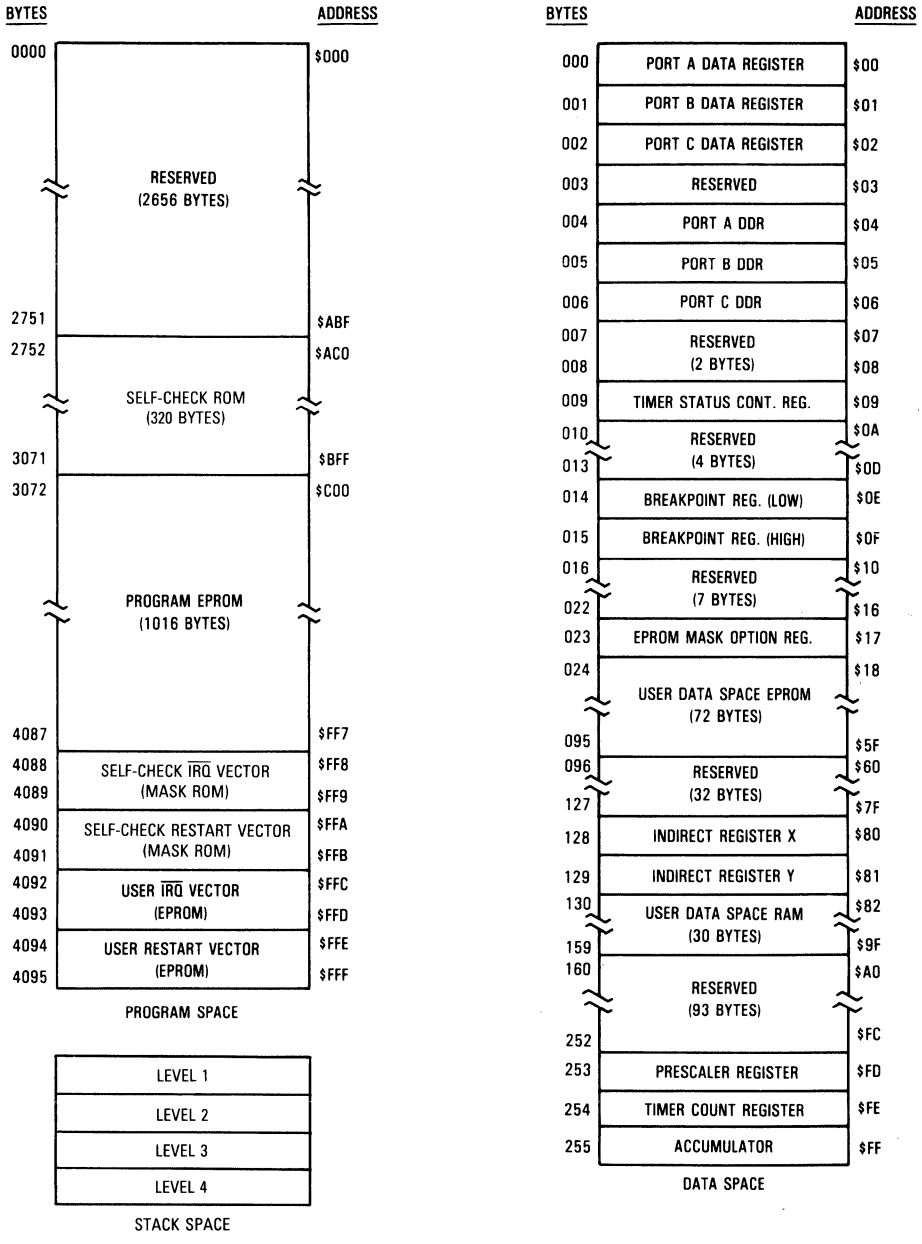
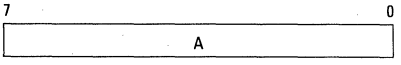


Figure 4. Memory Map

REGISTERS

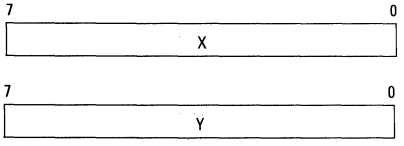
ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



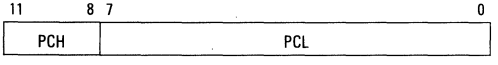
INDIRECT REGISTERS (X,Y)

These two registers are used to maintain pointers to other memory locations in data space. They are used in the register-indirect addressing mode and can be accessed with the direct, indirect, short direct, or bit set/clear modes.



PROGRAM COUNTER (PC)

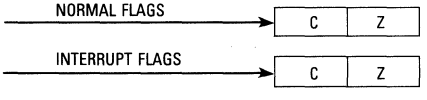
The program counter is a 12-bit register that contains the address of the next byte to be fetched from program space. The program counter is contained in low byte (PCL) and high nibble (PCH).



FLAGS (C,Z)

The first flag, the carry (C) bit, is set on a carry or borrow out of the arithmetic logic unit (ALU). It is cleared if the arithmetic operation does not result in a carry or borrow. The C bit is also set to the value of the bit tested in a bit test instruction. It participates in the rotate left (ROLA) instruction, as well.

The second flag, the zero (Z) bit, is set if the result of the last arithmetic or logic operation was equal to zero. Otherwise, it is cleared. Bit test instructions do not affect the Z bit.



There are two sets of these flags. One set is for interrupt processing (interrupt mode flags). The other set is for normal operations (program mode flags). When an interrupt occurs, a context switch is made from the program flags to the interrupt flags. An RTI forces the context switch back. While in either mode, only the flags for that mode are available. A context switch does not affect the value of the C or Z bits. Both sets of flags are cleared by $\overline{\text{RESET}}$.

STACK

A last-in-first-out (LIFO) stack is incorporated in the MCU that eliminates the need for a stack pointer. This non-accessible subroutine stack space is implemented in separate RAM, 12-bits wide. Whenever a subroutine call or interrupt occurs, the contents of the PC are shifted into the top register of the stack. At the same time, the top register is shifted one level deeper. This happens to all registers, with the bottom register falling out of the stack.

Whenever a return from subroutine or interrupt occurs, the top register is shifted into the PC and all lower registers are shifted one level higher. The stack RAM is four levels deep. If the stack is pulled more than four times with no pushes, then the address that was stored in the bottom level of the stack is shifted into the PC.

SELF CHECK

The MCU implements two forms of internal check: self check and the verify mode phase of EPROM programming. Self check performs an extensive functional check of the MCU using a signature analysis technique. For information on the verify mode in EPROM programming, see application note, *MC68704P2 8-Bit EPROM Microcomputer Programming Module* (AN-942).

Self-check mode is selected by holding the MDS and PA7 pins logic high and the PA6 pin logic low as $\overline{\text{RESET}}$ goes low to high. Monitoring the self-check mode's stages for successful completion requires external circuitry, see Motorola's *M6804 MCU Manual* (DLE404/D).

RESET

$\overline{\text{RESET}}$

All resets of the MC68704P2 are caused by the external reset input ($\overline{\text{RESET}}$). A reset can be achieved by pulling the $\overline{\text{RESET}}$ pin to logic low for a minimum of 96 oscillator cycles.

During reset, a delay of 96 oscillator cycles is needed before allowing the $\overline{\text{RESET}}$ input to go high. If power is being applied, $\overline{\text{RESET}}$ must be held low long enough for the oscillator to stabilize and then provide the 96 clocks. Connecting a capacitor and resistor to the $\overline{\text{RESET}}$ input, as shown in Figure 5 below typically provides sufficient delay.

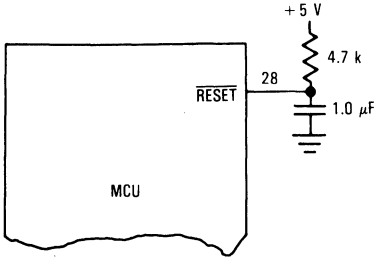


Figure 5. Powerup $\overline{\text{RESET}}$ Delay Circuit

INTERRUPT

The MCU can be interrupted by applying a logic low signal to the $\overline{IRQ}$ pin. However, a bit in the mask option register (MOR) determines whether the falling edge or the actual low level of the $\overline{IRQ}$ pin is sensed to indicate an interrupt.

EDGE-SENSITIVE OPTION

When the $\overline{IRQ}$ pin is pulled low, the internal interrupt request latch is set. Prior to each instruction fetch, this interrupt request latch is tested. If its output is low, an interrupt sequence is initiated at the end of the current instruction, provided the interrupt mask is cleared. Figure 6 contains a flowchart that illustrates both the reset and interrupt sequences.

The interrupt sequence consists of one cycle during which:

the interrupt request latch is cleared,

the interrupt mode flags are selected,

the program counter (PC) is saved on the stack,

the interrupt mask is set, and

The $\overline{IRQ}$ vector jump address is loaded into the PC.

The $\overline{IRQ}$ vector jump address is $FFC-$FFD in the single-chip mode and $FF8-$FF9 in the self-check mode. The contents of these locations are not decoded as an address to which the PC should jump. Instead, they are decoded like any other EPROM program word. So, it is essential that the vector contents specify a JMP instruction in addition to the starting address of the interrupt service routine. If required, this routine should save the values of the accumulator and the X and Y registers, since these values are not stored on the stack.

Internal processing of the interrupt continues until a return from interrupt (RTI) instruction is processed. During RTI the interrupt mask is cleared and the program mode flags are selected. The next instruction of the program is then fetched and executed.

When the interrupt was initially detected and the interrupt sequence started, the interrupt request latch was cleared so that the next interrupt could be detected. These steps occurred even as the first interrupt was being serviced. However, even though the second interrupt edge set the interrupt request latch during the first interrupt's processing, the second interrupt's sequence can not begin until completion of the interrupt service routine for the first interrupt. Completion of an interrupt service rou-

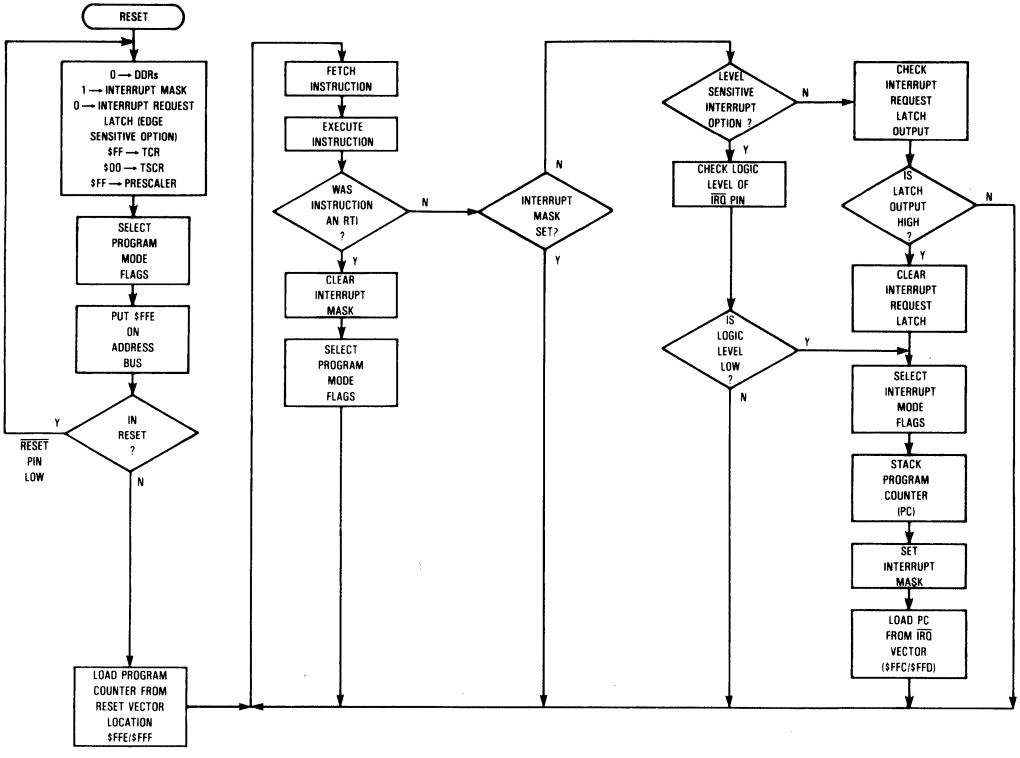


Figure 6. Reset and Interrupt Flowchart

tine is always accomplished using an RTI instruction to return to the main program. The interrupt mask, which is not directly available to the programmer, is cleared during the last cycle of the RTI instruction.

LEVEL-SENSITIVE OPTION

Actual operation of the level-sensitive and edge-sensitive options are similar. However, the level-sensitive option does not have an interrupt request latch. Since there is no interrupt request latch, the logic level of the $\overline{IRQ}$ pin is checked to detect the interrupt. Also, in the interrupt sequence there is no need to clear the interrupt request latch. These differences are shown in Figure 6.

POWERUP AND TIMING

During the powerup sequence, the interrupt mask is closed. This precludes any false interrupts. The PC is also loaded with the appropriate restart vector (jump instruction).

To open the interrupt mask, the user should do a JSR to an initialization subroutine that ends with an RTI instead of an RTS. The RTI opens the interrupt mask. Typical $\overline{RESET}$ and $\overline{IRQ}$ processes and their relationship to the interrupt mask are shown in Figure 7.

Maximum interrupt response time is six machine cycles. This includes five cycles for the longest instruction plus one for stacking the PC and switching flags.

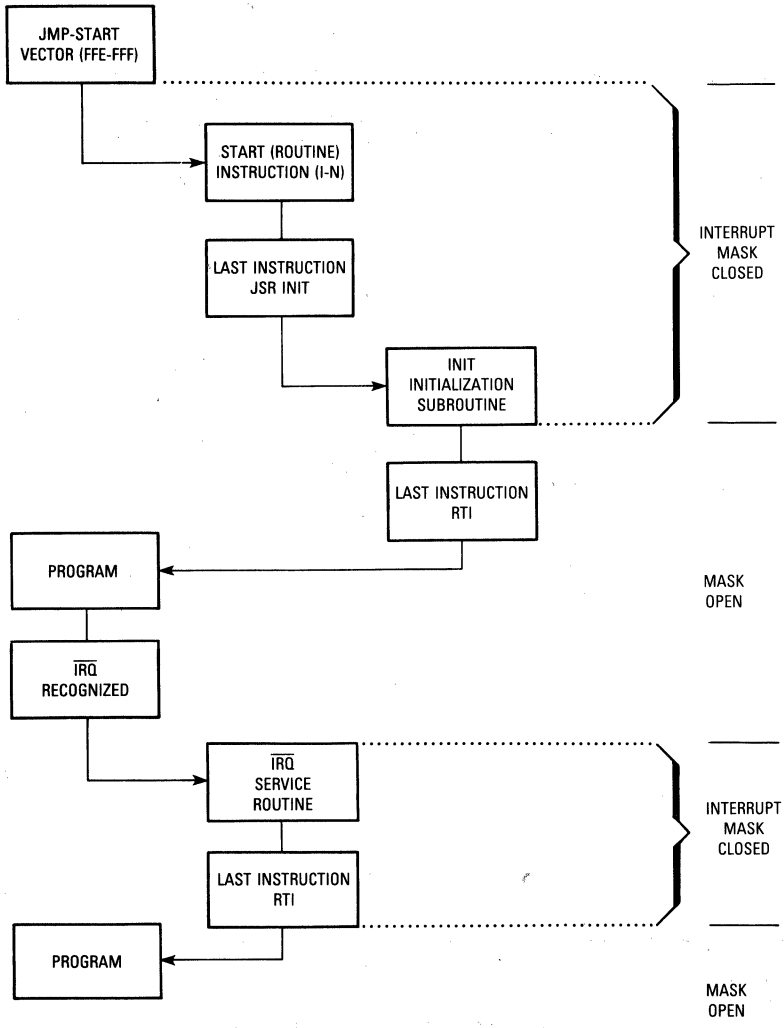
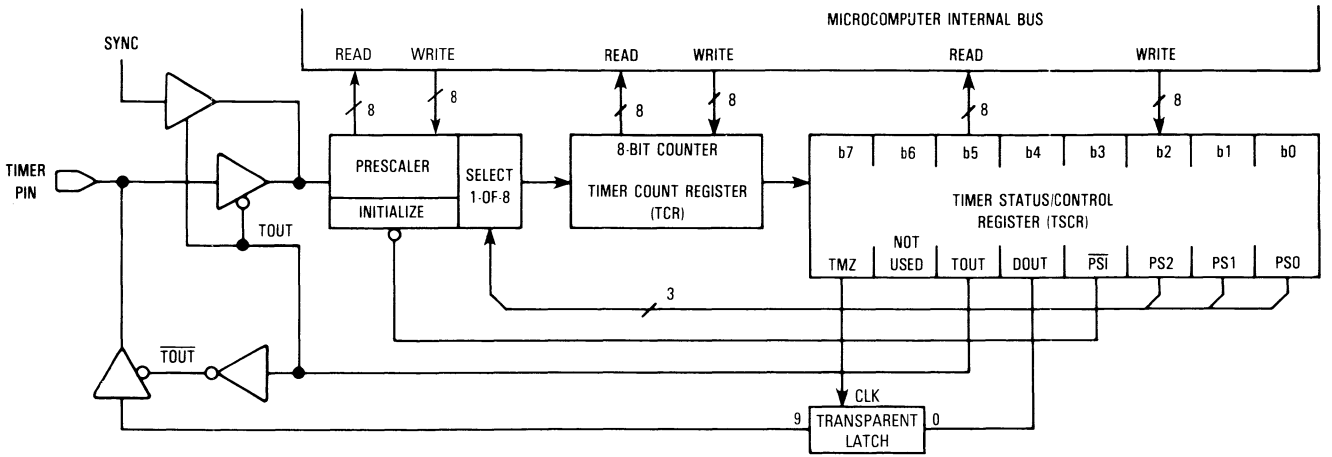


Figure 7. Interrupt Mask



TIMER PIN STATUS

TOUT	PRESCALER CLOCK	TIMER PIN
0	TIMER PIN	INPUT MODE
1	SYNC	OUTPUT MODE

Figure 8. Timer Block Diagram

TIMER

A block diagram of the MC68704P2 timer circuitry is shown in Figure 8. The timer logic in the MCU is comprised of a simple 8-bit counter called the timer counter. This counter is decremented by a 7-bit prescaler at a rate determined by the timer status/control register (TSCR).

PRESCALER

The prescaler is a 7-bit counter used to extend the maximum interval of the overall timer. This counter is clocked by a signal from the TIMER pin or by the internal sync pulse. It divides the frequency received by some factor to create the prescaler output. The factor by which the TIMER pin signal is divided is called the prescaler tap. The value of this tap is selected by three bits of the TSCR (PS0-PS2). These bits control the division of the prescaler input within the range of divide-by-2^0, to divide-by-2^7.

TIMER COUNTER

The timer counter, which may be read or loaded under program control, is decremented from a maximum value of 256 toward zero by the prescaler output. Both are decremented on rising clock edges.

The prescaler register and timer count register are readable and writeable. A write to either one will take precedence over the normal counter function. For example, if a value is written to the timer count register, and this write and a decrement-to-zero occur at the same time, the write takes precedence. TSCR bit one (TMZ) is not set until the next timer time out.

TIMER PIN

The TIMER pin may be programmed as either an input or an output. Its status depends on the value of TSCR bit 5 (TOUT). This relationship is shown in the TIMER pin status section of Figure 8. The frequency of the internal clock applied to the TIMER pin must be less than t_{byte}, which is (f_{osc}/48).

TIMER INPUT MODE

In the timer input mode, TOUT is logic zero and the TIMER pin is connected directly to prescaler input. So, the prescaler is clocked by the signal from the TIMER pin. The prescaler divides the TIMER pin clock input by the prescaler tap. The prescaler output then clocks the 8-bit timer count register. When this register is decremented to zero, it sets TSCR bit one (TMZ). This TMZ bit can be tested under program control to tell when the counter register has reached zero.

TIMER OUTPUT MODE

In the output mode, the TIMER pin is output. TOUT is a logic one. The prescaler is clocked by the internal sync pulse. This pulse is a divide-by-48 of the internal oscillator (f_{osc}/48). From this point on, operation is similar to that described for the input mode. However, in the output mode, once the prescaler decrements the timer counter to zero, the high TMZ bit state allows TSCR bit 4 (DOUT) to become direct input to the TIMER pin.

NOTE

TMZ is normally set to logic one when TCR decrements to zero and the timer times out. However, it may be set by a write of $00 to the timer counter or by a write to bit 7 of TSCR.

TIMER COUNT REGISTER ($FE)

The timer count register reflects the current count in the internal 8-bit counter. The register is the counter and can be read or written.

7							0
MSB							LSB
RESET:							
1	1	1	1	1	1	1	1

TIMER STATUS/CONTROL REGISTER (TSCR) ($09)

7	6	5	4	3	2	1	0
TMZ	—	TOUT	DOUT	$\overline{PSI}$	PS2	PS1	PS0
RESET:							
0	0	0	0	0	0	0	0

TMZ — Timer zero
- 1 = Timer count register has reached the all zero's state since the last time the TMZ bit was read
- 0 = This bit is cleared by a read of the TSCR if TMZ is read as logic one

Bit 6
- Not used by this register

TOUT — Timer output
- 1 = Output mode is selected for the timer
- 0 = Input mode is selected for the timer

DOUT — Data output
- Latched data at this bit is sent to the TIMER pin when both the TMZ and TOUT bits are logic high.

$\overline{PSI}$ — Prescaler initialize
- 1 = Prescaler begins to decrement
- 0 = Prescaler is initialized and counting is inhibited

PS0 — PS2
- These bits are used to select the prescaler tap. The coding of the bits is shown below:

PS2	PS1	PS0	Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

It is recommended that MVI or loading and storing instructions be used when changing bit values in the TSCR. Read-modify-write instructions can cause the TMZ to assume an unexpected state.

During reset, the TSCR is set to all zeroes; the TIMER pin is in the high impedance input mode; and DOUT LATCH is forced to a logic high. At the same time, PS0-PS2 coding sets the prescaler tap at divide-by-one, and bit 3 initializes the prescaler.

TIMER PRESCALER REGISTER ($FD)

The timer prescaler register reflects the current count of the 7-bit prescaler. This register is the prescaler counter and can be read or written.

6						0
MSB						LSB
RESET:						
1	1	1	1	1	1	1

EPROM

BREAKPOINT REGISTERS

The breakpoint registers are used as a program debugging aid. To enable the breakpoint registers:

— The MDS pin must be pulled high using a 300 ohm resistor to +5 volts.

— In the Port A I/O register, both PA6 and PA7 pins must be pulled low using a 10 kilohm resistor to ground.

A breakpoint address is written into address registers ARL and ARH by the user. The lower eight bits of the breakpoint address (A0-A7) are written into the ARL. The upper four bits (A8-A11) are written into the ARH.

Breakpoint Address Register Low (ARL) ($0E)

7	6	5	4	3	2	1	0
A7	A6	A5	A4	A3	A2	A1	A0
RESET:							
0	0	0	0	0	0	0	0

A7-A0
Breakpoint address bits A7 through A0.

Breakpoint Address Register High (ARH) ($0F)

7	6	5	4	3	2	1	0
X	X	X	X	A11	A10	A9	A8
RESET:							
0	0	0	0	0	0	0	0

A11-A8
Breakpoint address bits A11 through A8.

NOTE

ARL must be written after writing to ARH.

ARL and ARH are concatenated to form the breakpoint address. When the processor fetches an instruction having the same address as the breakpoint address, the MDS pin goes logic low for one machine cycle. This operation does not alter program flow.

MASK OPTION REGISTER (MOR) ($17)

The MC68704P2 uses the EPROM MOR during emulation to select the clock/oscillator, port, and interrupt request edge- and level-sensitive triggering options available on the MC6804J1/J2/P2 devices. The mask option register is not affected by $\overline{\text{RESET}}$.

7	6	5	4	3	2	1	0
OSC	X	PORT A	X	PORT B	PORT C	$\overline{\text{IRQ}}$	X

OSC — The oscillator option bit
1 = Resistor/capacitor mode of operation
0 = Crystal mode of operation
The crystal mode is selected in the EPROM programming mode, regardless of the state of OSC.

PORT A — Port A output selection bit
1 = Open drain output mode
0 = Three-state output mode

PORT B — Port B output selection bit
1 = Open drain output mode
0 = Three-state output mode

PORT C — Port C output selection bit
1 = Open drain output mode
0 = Three-state output mode

$\overline{\text{IRQ}}$ — Interrupt request bit
1 = Level-sensitive triggering input mode
0 = Edge-sensitive triggering input mode

Bits 6, 4, and 0
Not used in this register

Emulation

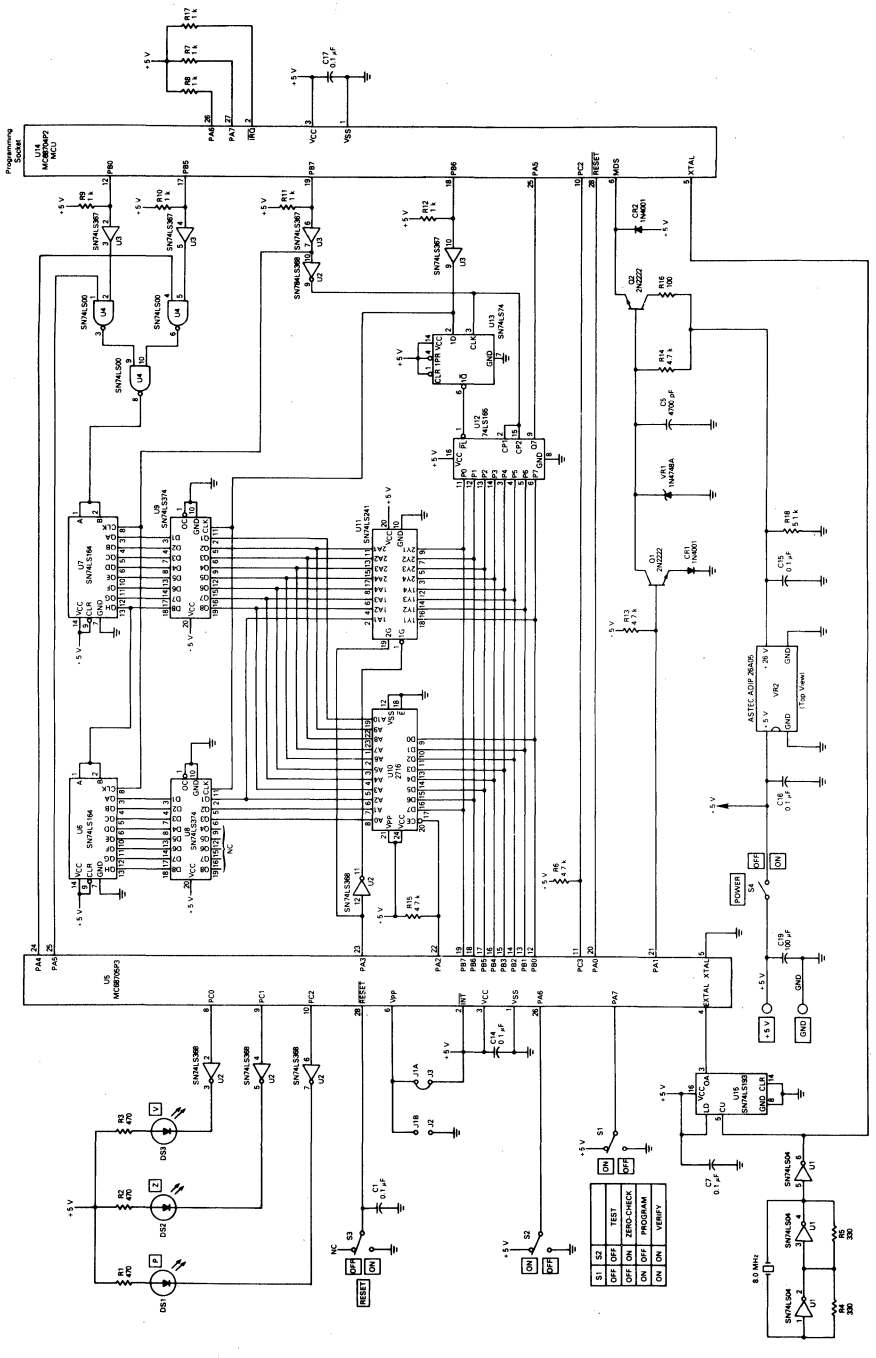
The MC68704P2 MCU internal EPROM can be programmed to emulate either the MC6804J1, MC6804J2, or the MC6804P2 MCU device. While the M6805 Family of EPROM MCUs have an on-chip bootstrap-loader program stored in mask ROM, the MC68704P2 does not. Additional programming hardware and software are required to program this MCU EPROM. For more specific information regarding the programming and erasing of the MCU EPROM; see application note, *MC68704P2 8-Bit EPROM Microcomputer Programming Module* (AN-942).

Emulation Limitations

This EPROM MCU is designed to emulate the functions of the MC6804J1/J2/P2 devices as closely as possible. Limitations to this capability pertain to the CMOS pullup option, execution out of data space, and packaging pin assignments of the MCU being emulated. The limitations do not apply to the timing, execution speed, or functionality of the MCU being emulated.

This MCU cannot emulate the CMOS pullup option. To implement the CMOS option, external 40 kilohm pullup resistors are connected to the specific I/O port signal lines. All other options are available through correct use of the MOR bytes.

It was necessary that the PC of this MCU have access to both the program and data space EPROM because of the implementation of the MCU programming hardware. Therefore, the MC68704P2 will execute code out of the



Notes:
1. Pins 1, 8, and 15 connected to GND for devices U2 and U3.
2. Pin 16 connected to +5 V for devices U2 and U3.
3. Pin 7 connected to GND, and pin 14 connected to +5 V for devices U1 and U4.

Figure 9. Programming Module Schematic

data space EPROM ($18-$5F). This anomaly is not permitted on the MC6804J1/J2/P2 ROM devices. When planning on operating ROM patterns from this EPROM MCU, the programmer should not use data space as extra program space.

The MC6804J1/J2 devices are packaged in 20-pin dual-in-line (DIL). The MC6804P2 and the MC68704P2 devices are packaged in 28-pin DIL packages. Device pin assignments must be adhered to. When emulating a 20-pin MCU with this EPROM MCU, all unused pins (PA0-PA3, PC0-PC3) should be grounded externally through a 10 kilohm resistor. This allows the MC68704P2 to emulate the software execution exactly as it would occur on the 20-pin device.

EPROM ERASING

This MCU EPROM is erased by exposure to a high intensity ultraviolet light (UV) with a wavelength of 2537 Angstrom. The recommended dosage is 15Ws/cm^2, (UV intensity at EPROM surface/area to be erased). UV lamps should be used without filters. The MC68704P2 should be positioned about one inch from the UV source. The duration of the exposure is a function of the radiant strength of the individual UV source.

EPROM PROGRAMMING HARDWARE

The MC68704P2 programming module, shown in Figure 9, is used to program the MC68704P2 MCU EPROM. To do this, the module requires a 2K EPROM of the 2716 type, a +5 Vdc power supply, and either a MC68705P3 or MC6805P2 MCU as the module MCU. For more specific information regarding the hardware and procedures necessary to program the MC68704P2; see either the advanced information data sheet for MC6804J1, MC6804J2, MC6804P2, and MC68704P2 8-bit microcomputers (MC6804J1/D) or application note, *MC68704P2 8-Bit EPROM Microcomputer Programming Module* (AN-942).

INSTRUCTION SET

The MCU has a set of 42 basic instructions. They can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is the accumulator; the other is obtained from memory using one of the addressing modes. Refer to the following list of instructions.

Function	Mnemonic
Load A from Memory	LDA
Load XP from Memory	LDX
Load YP from Memory	LDY
Store A in Memory	STA
Add to A	ADD
Subtract from A	SUB
AND Memory to A	AND
Transfer A to XP	TAX
Transfer A to YP	TAY
Transfer YP to A	TYA
Clear A	CLRA
Clear XP	CLRX
Clear YP	CLRY
Arithmetic Compare with Memory	CMP
Move Immediate Value to Memory	MVI
Arithmetic Left Shift of A	ASLA
Complement A	COMA
Rotate A Left and Carry	ROLA
Transfer XP to A	TPA

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. All INC and DEC forms along with all bit manipulation instructions use this method. Refer to the following list of instructions.

Function	Mnemonic
Increment Memory Location	INC
Increment A	INCA
Increment XP	INCX
Increment YP	INCY
Decrement Memory Location	DEC
Decrement A	DECA
Decrement XP	DECX
Decrement YP	DECY

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two byte instructions. Refer to the following list of instructions.

Function	Mnemonic
Branch if Carry Clear	BCC
Branch if Higher or Same	(BHS)
Branch if Carry Set	BCS
Branch if Lower	(BLO)
Branch if Not Equal	BNE
Branch if Equal	BEQ

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the 256 bytes of data space, where all port registers, port DDRs, timer, timer control, and on-chip

RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear, and bit test and branch functions are all implemented with a single instruction. For the test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list of instructions.

Function	Mnemonic
Branch If Bit n is Set	BRSET n(n=0 . . . 7)
Branch If Bit n is Clear	BRCLR n(n=0 . . . 7)
Set Bit n	BSET n(n=0 . . . 7)
Clear Bit n	BCLR n(n=0 . . . 7)

CONTROL INSTRUCTIONS

These instructions are used to control processor operation during program execution. The jump conditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following list of instructions.

Function	Mnemonic
Return from Subroutine	RTS
Return from Interrupt	RTI
No Operation	NOP
Jump to Subroutine	JSR
Jump Unconditional	JMP

IMPLIED INSTRUCTIONS

Since the accumulator and all other registers are located in RAM, many implied instructions exist. Some of the instructions recognized and translated by the assembler are shown below:

Mnemonic	Becomes	Mnemonic	Becomes
ASLA	ADD $FF	INCX	INC $80
BHS	BCC	INCY	INC $81
BLO	BCS	LDXI	MVI $80 DATA
CLRA	SUB $FF	LDYI	MVI $81 DATA
CLRX	MVI $80 #0	NOP	BEQ (PC) +1
CLRY	MVI $81 #0	TAX	STA $80
DECA	DEC $FF	TAY	STA $81
DECX	DEC $80	TXA	LDA $80
DECY	DEC $81	TYA	LDA $81
INCA	INC $FF		

Some examples of valuable instructions not specifically recognized by the assembler are shown below:

Mnemonic	Meaning
BCLR 7,$FF	Ensures A is plus
BSET 7, $FF	Ensures A is minus
BRCLR 7, $FF	Branch if A is plus
BRSET 7, $FF	Branch if A is minus
BRCLR 7, $80	Branch if X is plus (BXPL)
BRSET 7, $80	Branch if X is minus (BXMI)
BRCLR 7, $81	Branch if Y is plus (BYPL)
BRSET 7, $81	Branch if Y is minus (BYMI)

OPCODE MAP

Table 1 is a listing of all the instruction set opcodes applicable to the MC6804P2 MCU.

ADDRESSING MODES

The MCU has nine different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. It deals with objects in three different address spaces: program space, data space, and stack space. The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is located in program ROM. It is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution, such as a constant used to initialize a loop counter.

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the 256 bytes of data space with a single two-byte instruction.

SHORT DIRECT

In the short direct addressing mode, the MCU has four locations in data space RAM it can use, ($80, $81, $82, and $83). The opcode determines the data space RAM location, and the instruction is only one byte. Short direct addressing is a subset of the direct addressing mode. The X and Y registers are at locations $80 and $81, respectively.

EXTENDED

In the extended addressing mode, the effective address of the argument is obtained by concatenating the four least-significant bits of the opcode with the byte following the opcode to form a 12-bit address. Instructions using the extended addressing mode, such as JMP or JSR, are capable of branching anywhere in program space. An extended addressing mode instruction is two bytes long.

Table 1. Opcode Map

Hi / Low	Branch Instructions								Register/Memory, Control, and Read/Modify/Write Instructions				Bit Manipulation Instructions		Register/Memory and Read/Modify/Write		Hi / Low
	0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111	8 1000	9 1001	A 1010	B 1011	C 1100	D 1101	E 1110	F 1111	
0 0000	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	4 MVI 3 IMM	5 BRCLR0 3 B-T-B	4 BCLR0 2 BSC	4 LDA 1 R-IND	4 LDA 1 R-IND	0 0000
1 0001	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	*	5 BRCLR1 3 B-T-B	4 BCLR1 2 BSC	4 STA 1 R-IND	4 STA 1 R-IND	1 0001
2 0010	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	2 RTI 1 INH	5 BRCLR2 3 BT-B	4 BCLR2 2 BSC	4 ADD 1 R-IND	4 ADD 1 R-IND	2 0010
3 0011	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	2 RTS 1 INH	5 BRCLR3 3 B-T-B	4 BCLR3 2 BSC	4 SUB 1 R-IND	4 SUB 1 R-IND	3 0011
4 0100	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	4 COMA 1 INH	5 BRCLR4 3 B-T-B	4 BCLR4 2 BSC	4 CMP 1 R-IND	4 CMP 1 R-IND	4 0100
5 0101	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	4 ROLA 1 INH	5 BRCLR5 3 B-T-B	4 BCLR5 2 BSC	4 AND 1 R-IND	4 AND 1 R-IND	5 0101
6 0110	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	*	5 BRCLR6 3 B-T-B	4 BCLR6 2 BSC	4 INC 1 R-IND	4 INC 1 R-IND	6 0110
7 0111	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	*	5 BRCLR7 3 B-T-B	4 BCLR7 2 BSC	4 DEC 1 R-IND	4 DEC 1 R-IND	7 0111
8 1000	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	4 INC 1 S-D	4 DEC 1 S-D	5 BRSET0 3 B-T-B	4 BSET0 2 BSC	4 LDA 2 IMM	4 LDA 2 DIR	8 1000
9 1001	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	4 INC 1 S-D	4 DEC 1 S-D	5 BRSET1 3 B-T-B	4 BSET1 2 BSC	#	4 STA 2 DIR	9 1001
A 1010	2 BNE 1 REL	2 BNE 2 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	4 INC 1 S-D	4 DEC 1 S-D	5 BRSET2 3 B-T-B	4 BSET2 2 BSC	4 ADD 2 IMM	4 ADD 2 DIR	A 1010
B 1011	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	4 INC 1 S-D	4 DEC 1 S-D	5 BRSET3 3 B-T-B	4 BSET3 2 BSC	SUB IMM	4 SUB 2 DIR	B 1011
C 1100	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	4 LDA 1 S-D	4 STA 1 S-D	5 BRSET4 3 B-T-B	4 BSET4 2 BSC	4 CMP 2 IMM	CMP DIR	C 1100
D 1101	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	4 LDA 1 S-D	4 STA 1 S-D	5 BRSET5 3 B-T-B	4 BSET5 2 BSC	4 AND 2 IMM	4 AND 2 DIR	D 1101
E 1110	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JRSn 2 EXT	4 JMPn 2 EXT	4 LDA 1 S-D	4 STA 1 S-D	5 BRSET6 3 B-T-B	4 BSET6 2 BSC	#	4 INC 2 DIR	E 1110
F 1111	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	4 LDA 1 S-D	4 STA 1 S-D	5 BRSET7 3 B-T-B	4 BSET7 2 BSC	#	4 DEC 2 DIR	F 1111

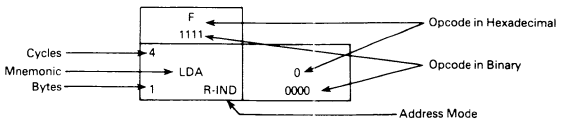
Abbreviations for Address Modes

INH	Inherent
S-D	Short Direct
B-T-B	Bit Test and Branch
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
R-IND	Register Indirect

* Indicates Instruction Reserved for Future Use

\# Indicates Illegal Instruction

LEGEND



RELATIVE

The relative addressing mode is only used in conditional branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −15 to +16 from the opcode address. The programmer need not calculate the offset when using the Motorola assembler, since it calculates the proper offset and checks to see that it is within the span of the branch.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte in which the specified bit is to be set or cleared. Thus, any bit in the 256 locations of data space memory that can be written to can be set or cleared with a single 2-byte instruction.

CAUTION

The corresponding DDRs for ports A, B, and C are write only registers (registers at $04, $05, and $06). A read operation on these registers is undefined. Since BSET and BCLR are read-modify-write functions, they cannot be used to set or clear a DDR bit; all "unaffected" bits would be set. Write all DDR bits in a port using a single-store instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit that is to be tested and its condition (set or clear) is included in the opcode. The data space address of the byte to be tested is in the single byte immediately following the opcode byte. The third byte is sign extended to twelve bits and becomes the offset added to the PC if the condition is true. This single 3-byte instruction allows the program to branch based on the condition of any readable bit in the 256 locations of data space. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry flag.

REGISTER-INDIRECT

In the register-indirect addressing mode, the operand is at the address in data space pointed to by the contents of one of the indirect registers, X or Y. The particular indirect register is selected by bit 4 of the opcode. Bit 4 decodes into an address that represents the register, $80 or $81. A register-indirect instruction is one byte long.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. These instructions are one byte long.

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range (Comm.)	T_A	0 to 70	°C
Operating Temperature Range (Ind.)	T_A	−40 to +85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature (Cerdip)	T_J	175	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{CC}$. Reliability of operation is enhanced if unused inputs except EXTAL are connected to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Cerdip	θ_{JA}	60	°C/W

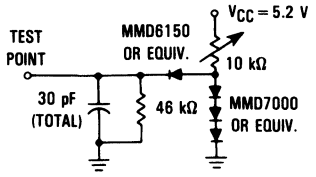


Figure 10. LSTTL Equivalent Test Load (Ports A, C, and TIMER)

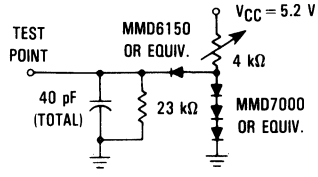


Figure 11. LSTTL Equivalent Test Load (Port B)

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

PROGRAMMING OPERATION ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.0 Vdc ± 0.5 Vdc, V_{SS} = GND, T_A = 20°C to 30°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Programming Voltage	V_{PP}	20	21	22	V
V_{PP} Supply Current (V_{PP} = 22.0 V)	I_{PP}	—	10	20	mA
Programming Oscillator Frequency	f_{oscp}	—	10	11	MHz
Programming Time (Per Byte)	t_{PRG}	—	5	50	ms

ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.0 Vdc ± 0.5 Vdc, V_{SS} = GND, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Internal Power Dissipation — No Port Loading	P_{INT}	—	165	275	mW
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Input Capacitance	C_{in}	—	10	—	pF
Input Current ($\overline{IRQ}$, $\overline{RESET}$)	I_{in}	—	2	20	μA

SWITCHING CHARACTERISTICS

(V_{CC} = +5.0 Vdc ± 0.5 Vdc, V_{SS} = GND, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Oscillator Frequency	f_{osc}	4.0	—	11.0	MHz
Bit Time	t_{bit}	0.364	—	1.0	μs
Byte Cycle Time	t_{byte}	4.36	—	12.0	μs
$\overline{IRQ}$ and TIMER Pulse Width	t_{WL}, t_{WH}	$2 \times t_{byte}$	—	—	—
$\overline{RESET}$ Pulse Width	t_{RWL}	$2 \times t_{byte}$	—	—	—
$\overline{RESET}$ Delay Time (External Capacitance = 1.0 μF)	t_{RHL}	100	—	—	ms

PORT DC ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.0 Vdc ± 0.5 Vdc, V_{SS} = GND, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Ports A and C (Open Drain)					
Output Low Voltage, I_{Load} = 0.4 mA	V_{OL}	—	—	0.5	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	4	40	μA
Open Drain Leakage (V_{out} = V_{CC})	I_{LOD}	—	4	40	μA
Port B (Open Drain)					
Output Low Voltage, I_{Load} = 1.0 mA	V_{OL}	—	—	0.5	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.5	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	8	80	μA
Open Drain Leakage (V_{out} = V_{CC})	I_{LOD}	—	8	80	μA
Ports A, C, and Timer (Push-Pull)					
Output Low Voltage, I_{Load} = 0.4 mA	V_{OL}	—	—	0.5	V
Output High Voltage, I_{Load} = −50 μA	V_{OH}	2.3	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	4	40	μA
Port B (Push-Pull)					
Output Low Voltage, I_{Load} = 1.0 mA	V_{OL}	—	—	0.5	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.5	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.3	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	8	80	μA

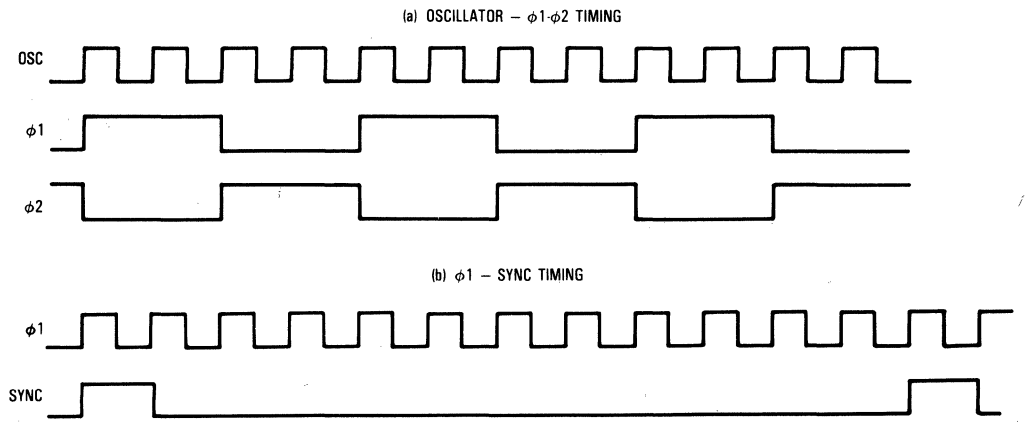


Figure 12. Clock Generator Timing Diagrams

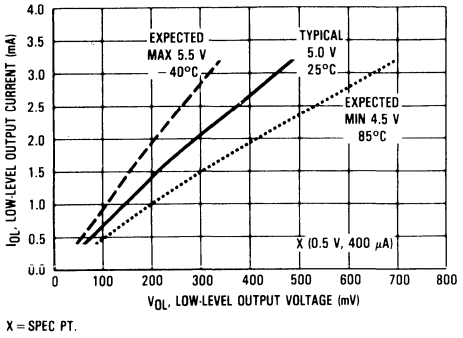


Figure 13. Typical V_{OL} vs I_{OL} for Ports A, C, and Timer

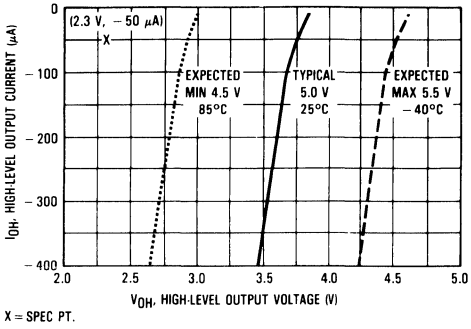


Figure 14. Typical V_{OH} vs I_{OH} for Ports A, C, and TIMER

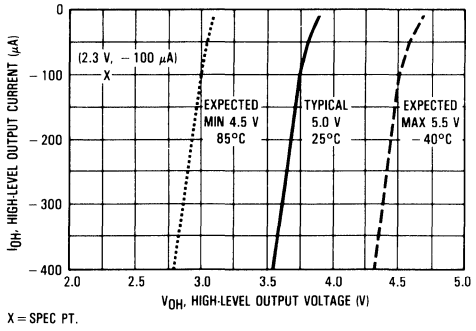


Figure 15. Typical V_{OH} vs I_{OH} for Port B

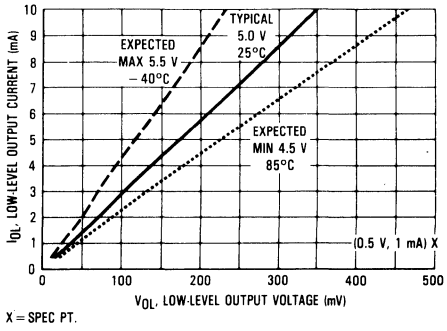


Figure 16. Typical V_{OL} vs I_{OL} for Port B

3

ORDERING INFORMATION

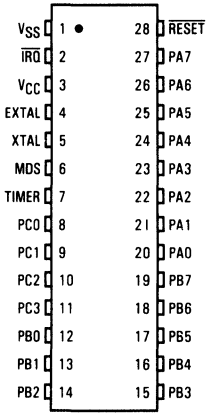
The MC68704P2 EPROM MCU device is only available in the 28-pin ceramic dual-in-line (CERDIP) package. The following table provides information pertaining to the temperature and MC order numbers of the MC68704P2.

Table 2. Generic Information

Package Type	Temperature	Order Number
Cerdip (S Suffix)	0°C to 70°C −40°C to +85°C	MC68704P2S MC68704P2CS

MECHANICAL DATA

PIN ASSIGNMENTS



3

MC68HC04J2

Technical Summary

8-Bit Microcontroller Unit

MC68HC04J2 HCMOS microcontroller unit (MCU) device is a member of the M6804 Family of single-chip microcontrollers. This device displays all the versatility of an MCU whose design-ability to process 8-bit variables one bit at a time already makes it tremendously cost effective.

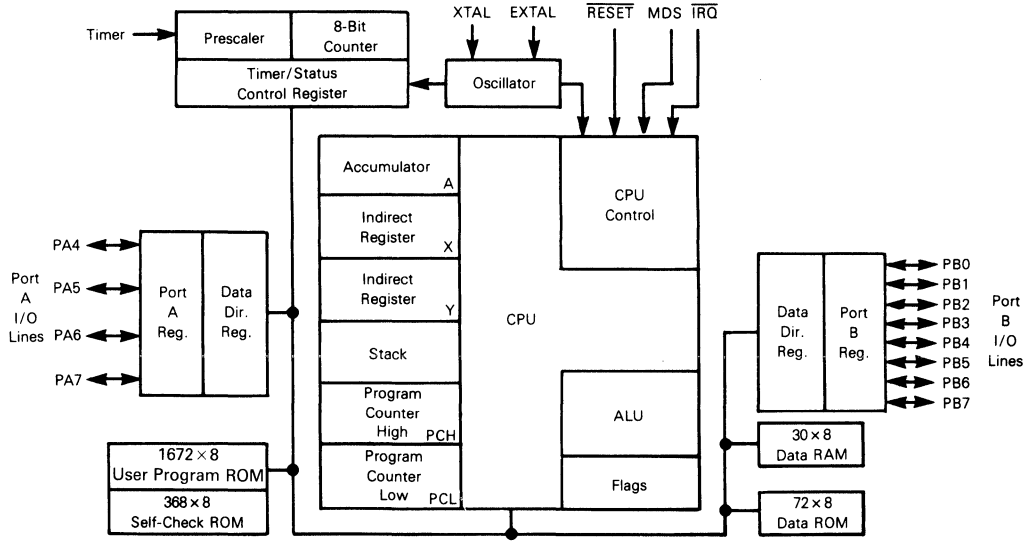
This technical summary contains limited information on the MC68HC04J2. For detailed information, refer to the advanced information data sheet for the MC68HC04J2, MC68HC04J3, and MC68HC04P3 8-bit microcontrollers (MC68HC04J2/D), or to the *M6804 MCU Manual*, DLE404/D.

Major hardware and software features of the MC68HC04J2 MCU are:

- On-Chip Clock Generator
- Memory Mapped I/O
- Software Programmable 8-Bit Timer with 7-Bit Prescaler
- Single Instruction Memory Examine/Change
- 72 Bytes of User Data ROM
- User Selectable Input Drive Options
- Optional Pull Down Devices on I/O Ports
- Mask Selectable Edge- or Level-Sensitive Interrupt Pin
- True Bit Manipulation
- Bit Test and Branch Instruction
- 368 Bytes Self-Check ROM
- Conditional Branches
- Timer Pin is Software Programmable as Clock Input or Timer Output
- 1672 Bytes of User Program ROM
- 30 Bytes of User RAM

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

V_{DD} AND V_{SS}

Power is supplied to the MCU using these two pins. V_{DD} is power, and V_{SS} is ground.

$\overline{IRQ}$

This pin provides the capability for asynchronously applying an external interrupt to the MCU. A pull-up resistor on this pin is a manufacturing mask option.

EXTAL AND XTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a resistor/capacitor combination, or an external signal is connected to these pins to provide a system clock. Selection is made by a manufacturing mask option. The different clock generator options are shown in Figure 1, along with crystal specifications.

Internal Clock Options

The crystal oscillator start-up time is a function of many variables. To ensure rapid oscillator start up, neither the crystal characteristics nor load capacitances should exceed recommendations. When using the on-board oscillator, the MCU should remain in a reset condition, with the $\overline{RESET}$ pin voltage below V_{IRES+}, until the oscillator has stabilized at its operating frequency.

TIMER

Two TIMER input modes as well as an output mode are available. In the input modes, the TIMER pin is configured as either a TIMER enable, or as the TIMER clock. In the output mode, the TIMER pin may generate transitions upon each occurence of timer underflow.

$\overline{RESET}$

The $\overline{RESET}$ pin is used to restart the processor to the beginning of a program. The program counter is loaded with the address of the restart vector. This should be a jump instruction to the first instruction of the main program. Together with the MDS pin, the $\overline{RESET}$ pin selects the operating mode of the MCU. A pullup resistor on this pin is a manufacturing mask option.

MDS

The mode select (MDS) pin places the MCU into special operating modes. When this pin is logic high at the exit of the reset state, the decoded state of PA6 and PA7 is latched to determine the operating mode. This choice can be either the single-chip, self-check, or ROM verify mode. However, if MDS is logic low at the end of the reset state, the single-chip operating mode is automatically selected. No external diodes, switches, transistors, etc. are required for single-chip mode selection.

INPUT/OUTPUT LINES (PA4-PA7, PB0-PB7)

These 12 lines are arranged into one 4-bit port (A) and one 8-bit port (B). All lines are programmable as either inputs or outputs under software control of the data direction registers.

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

There are 12 input/output pins. The 12 bidirectional lines can be selected to have internal pulldowns at the time of manufacture. All pins of each port are programmable as inputs or outputs under the control of the data direction registers (DDR).

The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic one for output, or a logic zero for input, as shown in Figure 2. When the registers are programmed as outputs, the latched data is readable regardless of the logic levels at the output pin due to output loading.

All the I/O pins are CMOS compatible as both inputs and outputs. Their standard configuration as outputs is three-state drive. Port B outputs are LED compatible. In addition, certain pins of both ports may be ordered equipped with pull down resistors.

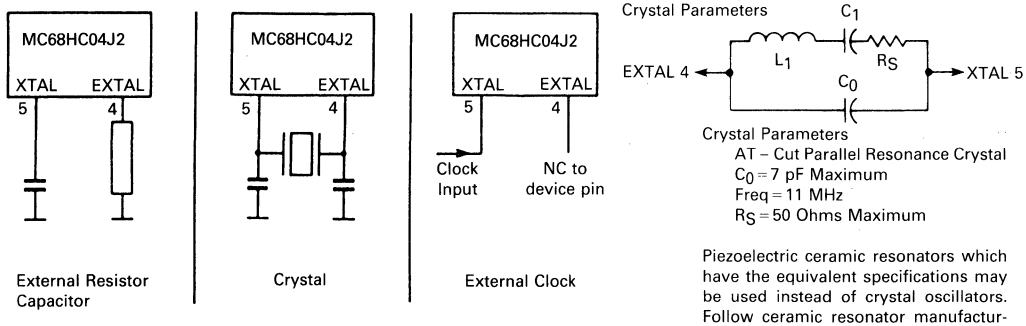
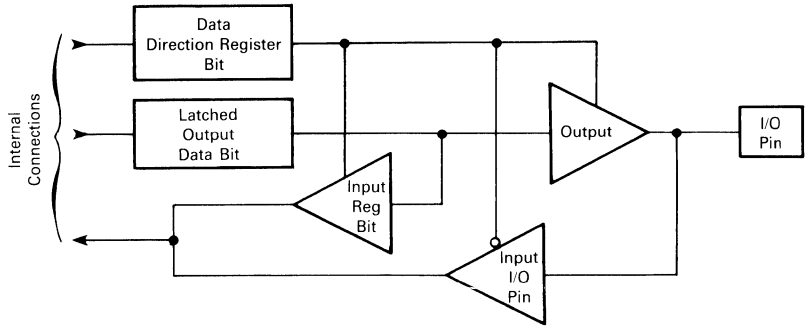


Figure 1. Clock Generator Options and Crystal Parameters



Data Direction Register Bit	Output Data Bit	Output State	Input To MCU
1	0	0	0
1	1	1	1
0	X	Hi-Z	Pin

Figure 2. Typical I/O Port Circuitry

Any write to a port writes to all of its data bits even though the port DDR may be set to input. This can be used as a tool to initialize the data registers and avoid undefined outputs. However, care must be exercised when using read-modify-write instructions. The data read corresponds to the pin level if the DDR is an input or to the latched output data when the DDR is an output.

Pull Down Device Option

The use of pull down devices on particular groupings of I/O ports is a manufacturing mask option available to the user. It is of use in applications where keyboards are interfaced directly to the MCU and similar situations. This option is available in the following configurations:

I/O Port	Resistor-Option Pin Groupings
Port A	PA4-PA7
Port B	PB3-PB7, PB4-PB7, PB1-PB2, PB0

Port Data Registers ($00, $01)

The port data registers are not initialized on reset. These registers should be initialized before changing the DDR bits to avoid undefined levels.

The source of data read from the port register is either the port I/O pin or previously latched output data. The source depends upon the contents of the corresponding DDR. The destination of data written to the port data register is an output data latch. If the corresponding DDR for the port I/O pin is programmed as an output, the data appears on the port pin.

Port A ($00)

7	6	5	4	3	2	1	0
				X	X	X	X

Port B ($01)

7	6	5	4	3	2	1	0

With regard to Port A only, the four LSB bits are unused. They are "don't care" (X) bits when written to, but are always logic high when read.

Port Data Direction Registers ($04, $05)

Port DDRs configure the port pins as either outputs or inputs. Each port pin can be programmed individually to act as an input or an output. A zero in the pin's corresponding DDR bit programs it as an input. A logic one programs it as an output. On reset, all the DDRs are initialized to a logic zero state to put the ports in the input mode.

Port A ($04)

7	6	5	4	3	2	1	0
				0	0	0	0

Port B ($05)

7	6	5	4	3	2	1	0

With regard to Port A DDR only, the four LSB bits are cleared after reset. These bits must not be set (logic one).

MEMORY

The MCU memory map (Figure 3), consists of 4352 bytes of addressable memory, I/O register locations, and stack space. This MCU has three separate memory spaces; program space, data space, and stack space.

The MCU is capable of addressing 4096 bytes of program space memory with its program counter and 256 bytes of data space memory with its instructions. Program space memory includes self-check ROM, program ROM, self-check and user program vectors, and reserved memory locations.

A non-accessible subroutine stack space RAM is provided. This stack space consists of a last-in-first-out (LIFO)

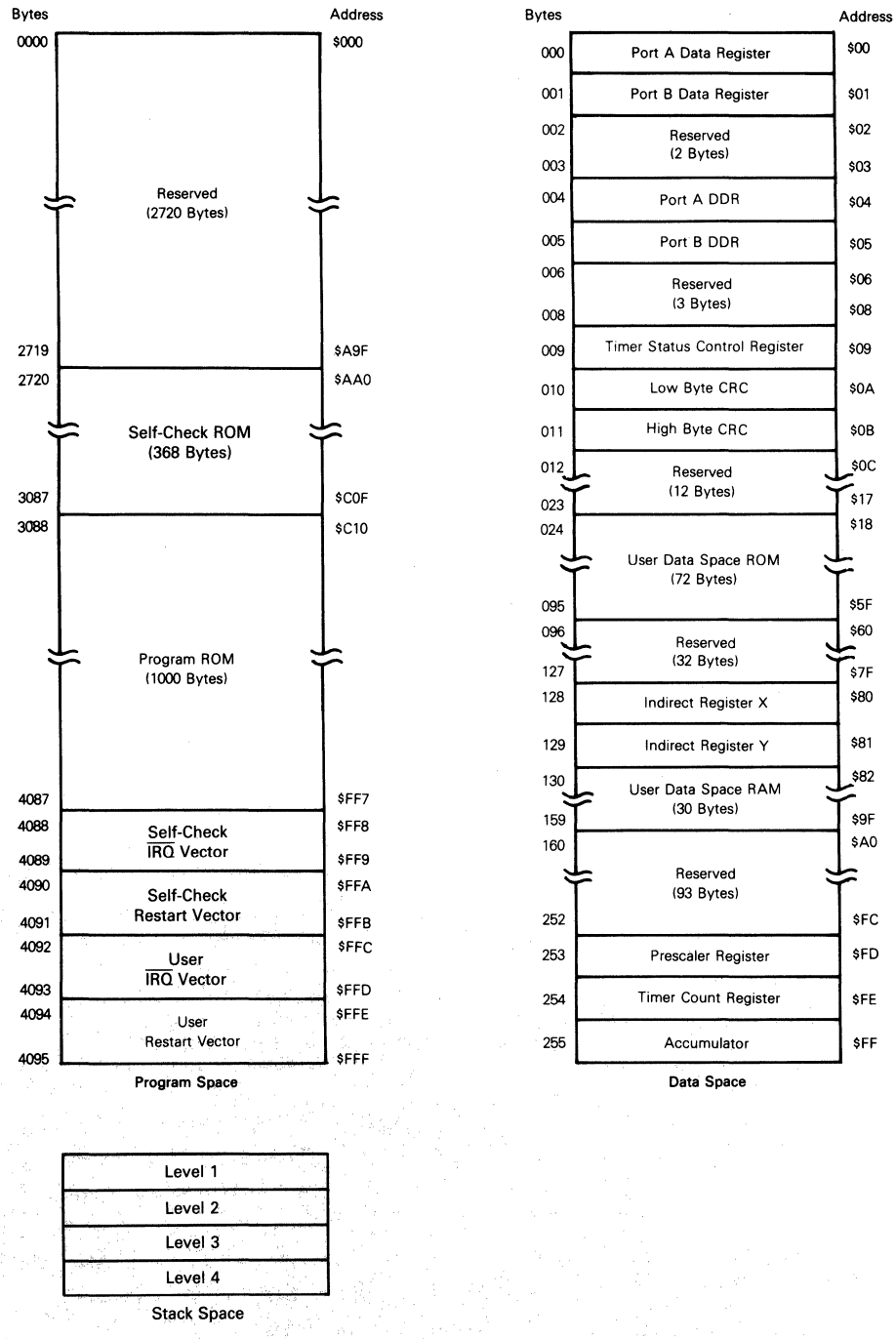


Figure 3. Memory Map

register. This register is used with inherent addressing to stack the return address for subroutines.

Indirect X and Y register locations $80 and $81 are generally used as pointers for such tasks as indirect addressing to data space locations. Short direct addressing allows access to the four data space addresses $80-$83 with single byte opcodes. The operations allowed are increment, decrement, load, and store. Data space locations $82 and $83 can be used for 8-bit counter locations.

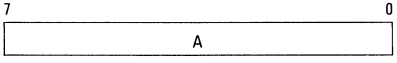
Program ROM Protect

A manufacturing mask option available to the user enables program ROM protection. Enabled, this option prevents the ROM contents from being output during self-check/ROM verify. This option does not prevent a go, no-go test of the ROM contents using the ROM verify mode.

REGISTERS

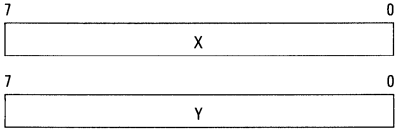
ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



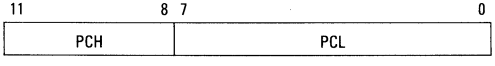
INDIRECT REGISTERS (X,Y)

These two registers are used to maintain pointers to other memory locations in data space. They are used in the register-indirect addressing mode and can be accessed with the direct, indirect, short direct, or bit set/clear modes.



PROGRAM COUNTER (PC)

The program counter is a 12-bit register that contains the address of the next byte to be fetched from program space. The program counter is contained in low byte (PCL) and high nibble (PCH).



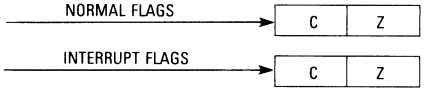
FLAGS (C,Z)

The first flag, the carry (C) bit, is set on a carry or borrow out of the arithmetic logic unit (ALU). It is cleared if the arithmetic operation does not result in a carry or borrow. The C bit is also set to the value of the bit tested in a bit test instruction. It participates in the rotate left (ROLA) instruction, as well.

The second flag, the zero (Z) bit, is set if the result of the last arithmetic or logic operation was equal to zero. Otherwise, it is cleared. Bit test instructions do not affect the Z bit.

NORMAL AND INTERRUPT FLAGS

There are two sets of these flags. One set is for interrupt processing (the interrupt mode flags). The other set is for normal operations (the program mode flags). When an interrupt occurs, a context switch is made from the program flags to the interrupt flags. An RTI forces the context switch back. While in either mode, only the flags for that mode are available. A context switch does not affect the value of the C or Z bits. Both sets of flags are cleared by $\overline{\text{RESET}}$.



STACK

A last-in-first-out (LIFO) stack is incorporated in the MCU that eliminates the need for a stack pointer. This non-accessible subroutine stack space is implemented in separate RAM, 12-bits wide. Whenever a subroutine call or interrupt occurs, the contents of the PC are shifted into the top register of the stack. At the same time, the top register is shifted one level deeper. This happens to all registers, with the bottom register falling out of the stack.

Whenever a return from subroutine or interrupt occurs, the top register is shifted into the PC and all lower registers are shifted one level higher. The stack RAM is four levels deep. If the stack is pulled more than four times with no pushes, then the address that was stored in the bottom level of the stack is shifted into the PC.

CRC REGISTERS

Two 8-bit registers are implemented in RAM primarily as self-check and ROM verify modes. The two registers are memory mapped in data space at addresses $0A (CRC low), and $0B (CRC high).

Provided no write or read/modify/write operation is used to change the contents of these two locations, the registers are configured to perform CRC calculations. By simply reading a register, a pseudo-random number can be generated.

If a write or a read/modify/write is performed on addresses $0A or $0B, then the CRC circuitry is disabled. Both registers can be used as RAM locations until the next $\overline{\text{RESET}}$. $\overline{\text{RESET}}$ enables the CRC circuitry again.

SELF CHECK

The MCU implements two forms of internal check, self check and ROM verify. Self check performs an extensive functional check of the MCU using a signature analysis technique. ROM verify uses a similar method to check the contents of program ROM.

Self-check mode is selected by holding the MDS and PA7 pins logic high, and PA6 logic low as $\overline{\text{RESET}}$ goes low to high. ROM verify mode is entered by holding MDS, PA7, and PA6 logic high as $\overline{\text{RESET}}$ goes low to high. Unimplemented program space ROM locations are also tested. Monitoring the self-check mode's stages for successful completion requires external circuitry.

RESET

The MCU can be reset by initial power up or by external reset input ($\overline{\text{RESET}}$).

POWER-ON-RESET (POR)

During a power-on-reset, the timer is used to count 1920 external clock cycles. This allows the oscillator to stabilize before releasing the internal reset, irrespective of the state of the $\overline{\text{RESET}}$ pin. If the $\overline{\text{RESET}}$ pin is low at the end of the delay, the processor remains in the reset condition.

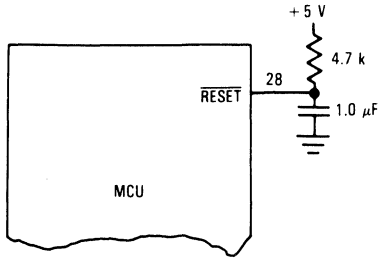


Figure 4. Power Up $\overline{\text{RESET}}$ Delay Circuit

$\overline{\text{RESET}}$

A reset can also be achieved by pulling the $\overline{\text{RESET}}$ pin to logic low for a minimum of two clock cycles. The delay is not implemented in this case.

INTERRUPT

There are two ways this MCU can be interrupted; by applying a logic low signal to the $\overline{\text{IRQ}}$ pin, or by a positive transition of the TMZ bit of TSCR with the ETI bit set. However, a manufacturing mask option determines whether the falling edge or the actual low level of the $\overline{\text{IRQ}}$ pin is sensed to indicate an interrupt.

EXTERNAL INTERRUPT EDGE-SENSITIVE OPTION

When the $\overline{\text{IRQ}}$ pin is pulled low, the internal interrupt request latch is set. Prior to each instruction fetch, this interrupt request latch is tested. If its output is low, an interrupt sequence is initiated at the end of the current instruction, provided the interrupt mask is cleared. Figure 5 contains a flowchart that illustrates the interrupt and instruction processing sequences.

The interrupt sequence consists of one cycle during which:

The interrupt request latch is cleared;

The interrupt mode flags are selected;

The program counter (PC) is saved on the stack;

The interrupt mask is set; and

The $\overline{\text{IRQ}}$ vector jump address is loaded into the PC.

The $\overline{\text{IRQ}}$ vector jump address is $FFC-$FFD in the single-chip mode and $FF8-$FF9 in the self-check mode. The contents of these locations are not decoded as an address to which the PC should jump. Instead, they are decoded like any other ROM word. So, it is essential that the vector contents specify a JMP instruction in addition to the starting address of the interrupt service routine. If required, this routine should save the values of the accumulator and the X and Y registers, since these values are not stored on the stack.

Internal processing of the interrupt continues until a return from interrupt (RTI) instruction is processed. During RTI the interrupt mask is cleared and the program mode flags are selected. The next instruction of the program is then fetched and executed.

When STOP is processed, the interrupt mask is cleared and the oscillator stopped. Checks are made for either $\overline{\text{REST}}$ or $\overline{\text{IRQ}}$. If $\overline{\text{RESET}}$ is detected, the $\overline{\text{RESET}}$ sequence is initiated. If $\overline{\text{IRQ}}$ is detected, the system oscillator is enabled along with the clock. In both cases, a delay is executed by the timer to allow oscillator stabilization before the CPU is enabled and the interrupt serviced.

When WAIT is processed, the interrupt mask is cleared and the CPU clock disabled. The interrupt latch is tested. Detection of $\overline{\text{RESET}}$ initiates the $\overline{\text{RESET}}$ sequence. Detection of $\overline{\text{IRQ}}$ or timer interrupt enables the CPU clock and initiates servicing of the interrupts.

When RTI is processed, the program counter is pulled from the stack. The program flags are selected and the interrupt mask cleared. The interrupt latch is then tested before the next instruction.

When the interrupt was initially detected and the interrupt sequence started, the interrupt request latch was cleared so that the next interrupt could be detected. This was done even as the first interrupt was being serviced. However, even though the second interrupt set the interrupt request latch during the first interrupt's processing, the second interrupt's sequence cannot begin until completion of the interrupt service routine for the first interrupt. Completion of an interrupt service routine is always accomplished using an RTI instruction to return to the main program. The interrupt mask, which is not directly available to the programmer, is cleared during the last cycle of the RTI instruction.

EXTERNAL INTERRUPT EDGE/LEVEL-SENSITIVE OPTION

The edge/level-sensitive option performs as described in the preceding section but adds the potential for level-sensitive operation. Level-sensitive operation tests the state of the $\overline{\text{IRQ}}$ and initiates and interrupt service routine if the $\overline{\text{IRQ}}$ pin is found to be logic low.

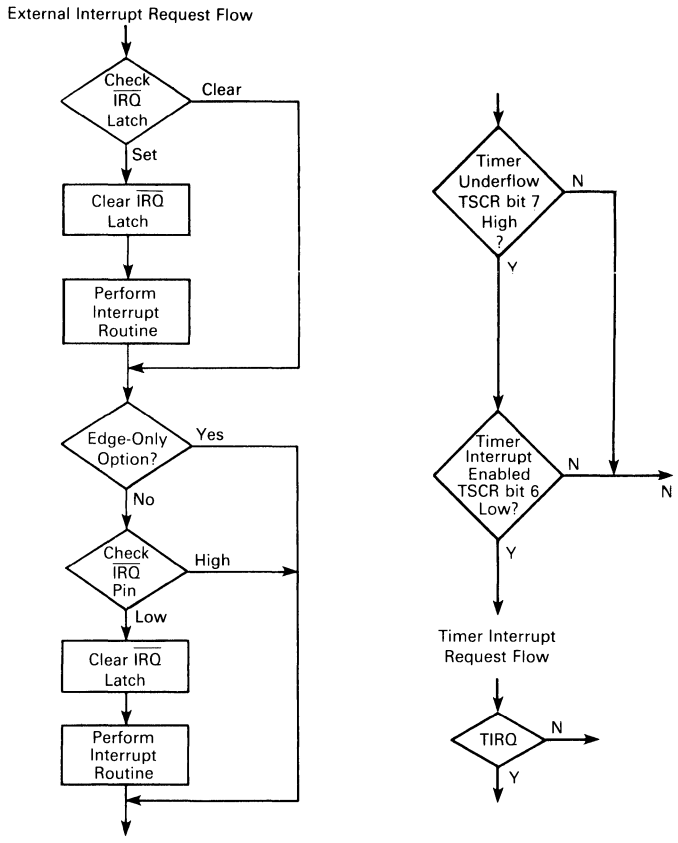


Figure 5. Interrupt Sequences

POWER UP AND TIMING

During the power up sequence, the interrupt mask is closed. This precludes any false interrupts. The PC is also loaded with the appropriate restart vector (jump instruction).

To open the interrupt mask, the user should do a JSR to an initialization subroutine that ends with an RTI instead of an RTS. The RTI opens the interrupt mask. Typical $\overline{\text{RESET}}$ and $\overline{\text{IRQ}}$ processes and their relationship to the interrupt mask are shown in Figure 7.

Maximum interrupt response time is eight machine cycles. This includes five cycles for the longest instruction plus one for stacking the PC and switching flags. Two additional cycles are used to synchronize $\overline{\text{IRQ}}$ input with the internal machine cycle frequency.

TIMER INTERRUPT

A timer interrupt is requested by a transition of the TMZ bit of the timer status/control register (TSCR) from logic low to high. Such a positive transition is caused either by the timer count register reaching the all zero state, or by any program instruction that writes a one to the TMZ bit.

The timer interrupt request is maskable by clearing bit 6 of the TSCR (ETI bit). ETI is cleared by $\overline{\text{RESET}}$.

During the interrupt routine, to determine whether an interrupt was caused externally or by the timer, it is necessary to test the state of the TMZ bit in the TSCR.

It is important to service a timer interrupt and clear the TMZ bit before the timer counter underflows again. Otherwise, because only a single interrupt can be latched, there is no way of telling how many timer interrupts occur while the original interrupt is being serviced.

LOW-POWER MODES

STOP

The STOP instruction places the MCU in its lowest power consumption mode. In the STOP mode, the internal oscillator is turned off, causing all internal processing and the timer to be halted. Current consumption is thus dropped to leakage levels.

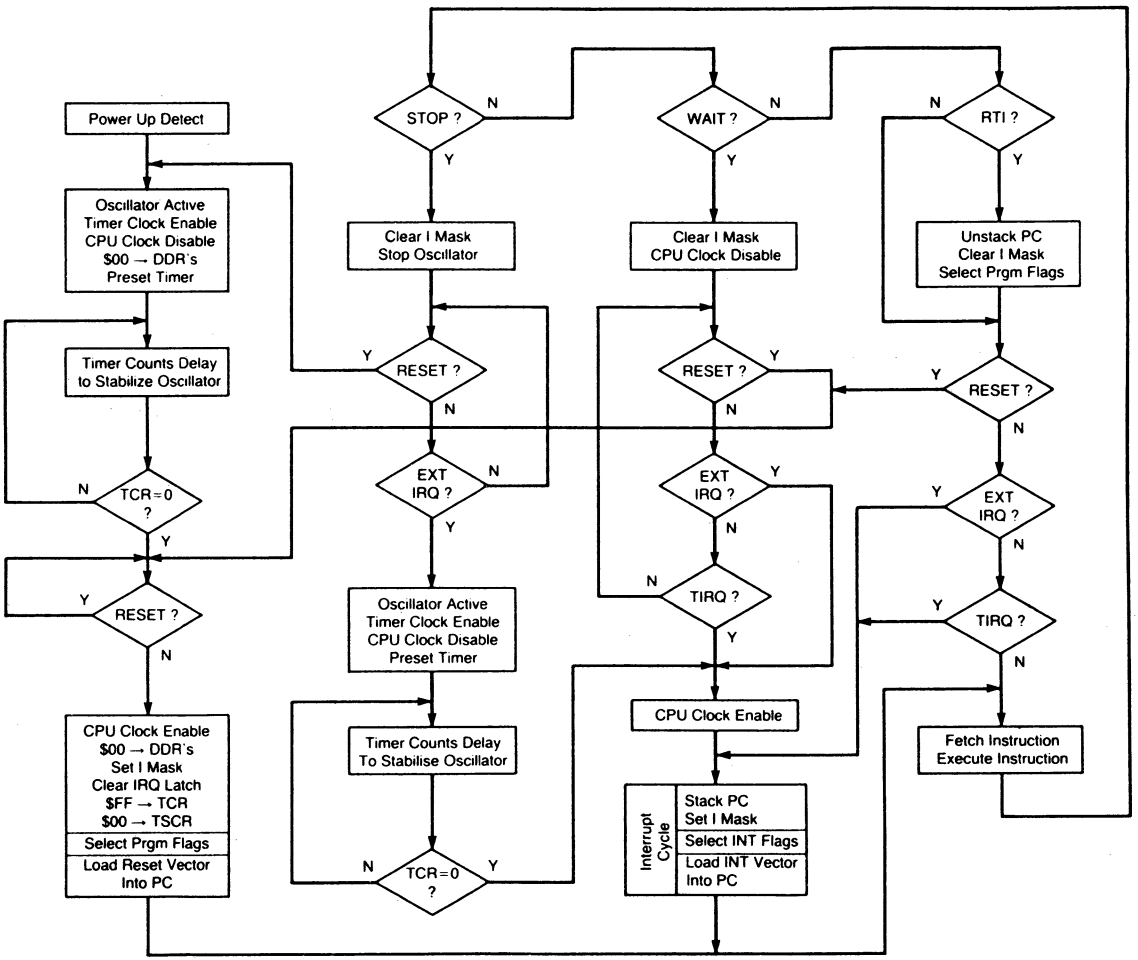


Figure 6. Instruction Processing Sequence

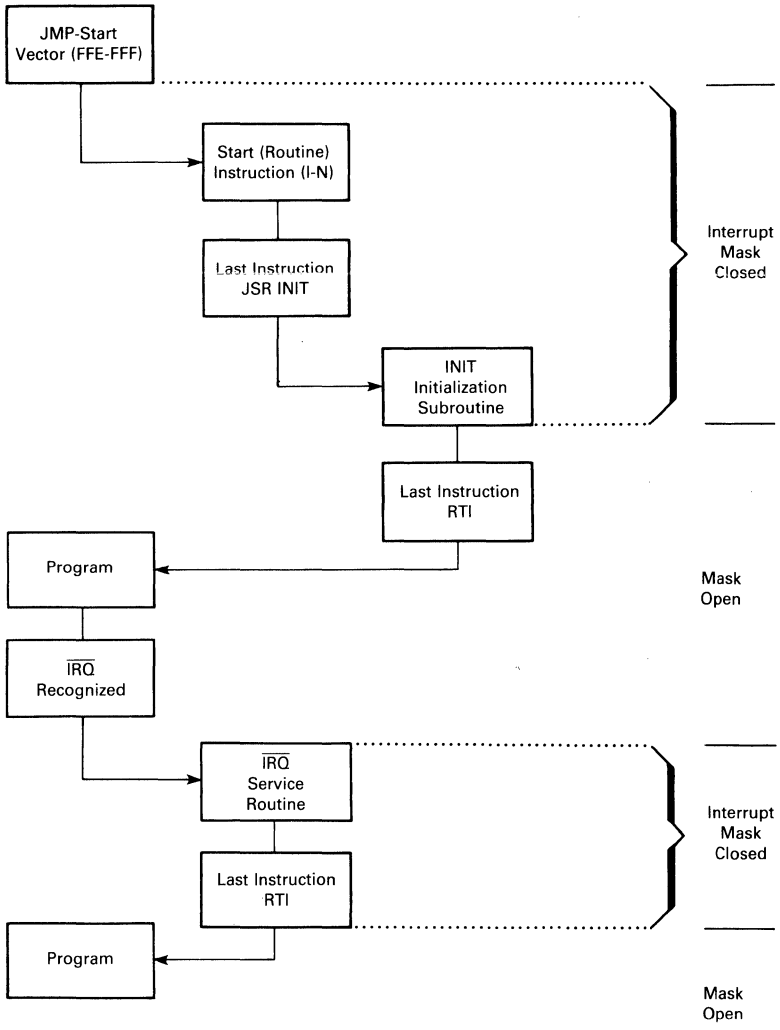


Figure 7. Interrupt Mask

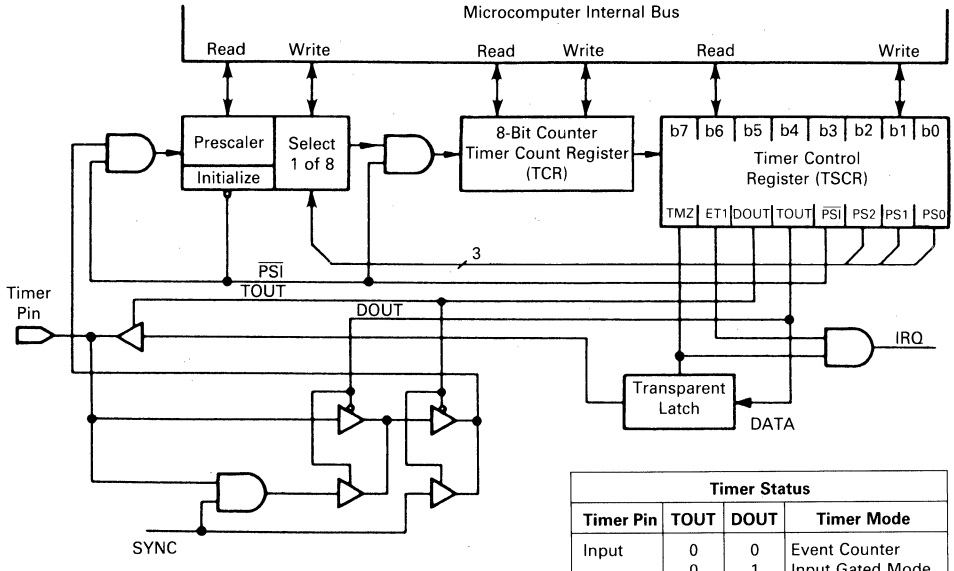
Providing the supply voltage remains within data sheet limits, the contents of the TSCR, accumulator, and all data space RAM remain unchanged in STOP mode.

Causing an interrupt or reset by pulling the $\overline{\text{RESET}}$ or $\overline{\text{IRQ}}$ pins low is the only way to bring the processor out of STOP mode. During this exit from STOP, the timer is used to provide the delay time necessary for the oscillator to stabilize. So, the prescaler and timer count register contents must be considered corrupted.

WAIT

The WAIT instruction places the MCU in a low power consumption mode, but the WAIT mode consumes somewhat more power than the STOP mode. In the WAIT mode, the internal clock is disabled from all internal circuitry except for the timer. So, all internal processing is halted. However, the timer continues to decrement normally if the PSI bit of TSCR is set.

During the WAIT mode, external interrupts are enabled. All other registers, memory, and I/O lines remain in their last state. Pulling the $\overline{\text{IRQ}}$ or $\overline{\text{RESET}}$ pin to logic low causes an exit from the WAIT mode. In addition, ETI bit of TSCR can be enabled by software prior to entering the WAIT state. This allows an exit from WAIT via a timer interrupt as well as via external interrupts.



Timer Status			
Timer Pin	TOUT	DOUT	Timer Mode
Input	0	0	Event Counter
	0	1	Input Gated Mode
Output	1	0	Output
	1	1	

Figure 8. Timer Block Diagram

TIMER

A block diagram of the MC68HC04J2 timer circuitry is shown in Figure 8. The timer logic in the MCU is comprised of a simple 8-bit counter called the timer counter. This counter is decremented by a 7-bit prescaler at a rate determined by the timer status/control register (TSCR).

PRESCALER

The prescaler is a 7-bit counter used to extend the maximum interval of the overall timer. This counter is clocked by a signal from the TIMER pin or by the internal sync pulse. It divides the frequency received by some factor to create the prescaler output. The factor by which the TIMER pin signal is divided is called the prescaler tap. The value of this tap is selected by three bits of the TSCR (PS0-PS2). These bits control the division of the prescaler input within the range of divide-by-2^0, to divide-by-2^7.

TIMER COUNTER

The timer counter, which may be read or loaded under program control, is decremented from a maximum value of 256 toward zero by the prescaler output. Both are decremented on rising clock edges.

The prescaler register and timer count register are readable and writeable. A write to either one will take precedence over the normal counter function. For example, if a value is written to the timer count register, and this write and a decrement-to-zero occur at the same time, the write takes precedence and TSCR bit one (TMZ) is not set until the next timer time out.

TIMER PIN

The TIMER pin may be programmed as either an input or an output. Its status depends on the value of TSCR bits 4 (DOUT) and 5 (TOUT). Two distinct input modes exist; input gated mode and input event counter mode. This relationship is shown in the TIMER pin status section of Figure 8. The frequency of the internal clock applied to the TIMER pin must be less than t_{byte}, which is the frequency of the oscillator divided by either 12, 24, or 48. Whether f_{osc} is divided by 12, 24, or 48 is determined by the clock divide ratio, which is selected by the manufacturing mask.

TIMER INPUT EVENT COUNTER MODE

In the timer input event counter mode, both TOUT and DOUT are logic zero. The TIMER pin is effectively connected directly to prescaler input. So, the timer/prescaler is clocked by the signal applied from the TIMER pin.

TIMER INPUT GATED MODE

In the input gated mode, TOUT is logic zero and DOUT is logic one. The timer pin is an input which decrements the prescaler each machine cycle as long as timer pin is logic high. When the pin is logic low, counting is inhibited. This mode permits the counting of the period of time during which the timer pin is logic high, based on the system clock and prescaler values. Gate times are $f_{osc}/12$, $f_{osc}/24$, and $f_{osc}/48$.

TIMER OUTPUT MODE

In the output mode, TOUT is logic one and the TIMER pin is connected to the DOUT latch. So, the timer prescaler is clocked by the internal sync pulse. This pulse is a divide-by-12, 24, or 48 of the internal oscillator depending on the mask option. However, in the output mode, once the prescaler decrements the timer count register to zero, the low TSCR bit 1 (TMZ) bit state is used to drive the data latched at TSCR bit 4 (DOUT), onto the TIMER pin.

NOTE

TMZ is normally set to logic one when TCR decrements to zero and the timer times out. However, it may be set by a write of $00 to TCR or by a write to bit 7 of TSCR.

TIMER COUNT REGISTER ($FE)

The timer count register reflects the current count in the internal 8-bit counter. The register is the counter and can be written.

7							0
MSB							LSB
RESET:							
1	1	1	1	1	1	1	1

TIMER STATUS/CONTROL REGISTER (TSCR) ($09)

7	6	5	4	3	2	1	0
TMZ	ETI	TOUT	DOUT	$\overline{PSI}$	PS2	PS1	PS0
RESET:							
0	0	0	0	0	0	0	0

TMZ — Timer Zero
- 1 = Timer count register has reached the all-zero state since the last time the TMZ bit was read.
- 0 = This bit is cleared by a read of the TSCR if TMZ is read as logic one.

ETI — Enable Timer Interrupt
- 1 = Timer interrupt enabled
- 0 = Timer interrupt disabled

TOUT — Timer Output
- 1 = Output mode is selected for the timer.
- 0 = Input modes are selected for the timer.

DOUT — Data Output

In the output mode, latched data at this bit is sent to the TIMER pin when both the TMZ and TOUT bits are logic high.

In the input mode:
- 1 = Timer input gated mode is selected.
- 0 = Timer input event counter mode is selected.

$\overline{PSI}$ — Prescaler Initialization
- 1 = Prescaler begins to decrement.
- 0 = Prescaler is initialized and counting is inhibited.

PS0-PS2

These bits are used to select the prescaler tap. The coding of the bits is shown below:

PS2	PS1	PS0	Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

It is recommended that MVI or loading and storing instructions be used when changing bit values in the TSCR. Read-modify-write instructions can cause the TMZ to assume an unexpected state.

During reset, the TSCR is set to all zeroes. The TIMER pin is in the high impedance input mode; and DOUT LATCH is forced to a logic high. At the same time, PS0-PS2 coding sets the prescaler tap at divide-by-one, and bit 3 initializes the prescaler.

TIMER PRESCALER REGISTER ($FD)

The timer prescaler register reflects the current count of the 7-bit prescaler. This register is the prescaler counter and can be written.

6						0
MSB						LSB
RESET:						
1	1	1	1	1	1	1

INSTRUCTION SET

The MCU has a set of 42 basic instructions. They can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. All INC and DEC forms along with all bit manipulation instructions use this method. Refer to the following list of instructions.

Function	Mnemonic
Increment Memory Location	INC
Increment A	INCA
Increment XP	INCX
Increment YP	INCY
Decrement Memory Location	DEC
Decrement A	DECA
Decrement XP	DECX
Decrement YP	DECY

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is the accumulator; the other is obtained from memory using one of the addressing modes. Refer to the following list of instructions.

Function	Mnemonic
Load A from Memory	LDA
Load XP from Memory	LDX
Load YP from Memory	LDY
Store A in Memory	STA
Add to A	ADD
Subtract from A	SUB
AND Memory to A	AND
Transfer A to XP	TAX
Transfer A to YP	TAY
Transfer YP to A	TYA
Transfer XP to A	TPA
Clear A	CLRA
Clear XP	CLRX
Clear YP	CLRY
Arithmetic Compare with Memory	CMP
Move Immediate Value to Memory	MVI
Arithmetic Left Shift of A	ASLA
Complement A	COMA
Rotate A Left and Carry	ROLA

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two byte instructions. Refer to the following list of instructions.

Function	Mnemonic
Branch if Carry Clear	BCC
Branch if Higher or Same	(BHS)
Branch if Carry Set	BCS
Branch if Lower	(BLO)
Branch if Not Equal	BNE
Branch if Equal	BEQ

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the 256 bytes of the memory space, where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear, and bit test and branch functions are all implemented with a single instruction. For the test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list of instructions.

Function	Mnemonic
Branch If Bit n is Set	BRSET n(n=0 . . . 7)
Branch If Bit n is Clear	BRCLR n(n=0 . . . 7)
Set Bit n	BSET n(n=0 . . . 7)
Clear Bit n	BCLR n(n=0 . . . 7)

CONTROL INSTRUCTIONS

These instructions are used to control processor operation during program execution. The jump conditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following list of instructions.

Function	Mnemonic
Return from Subroutine	RTS
Return from Interrupt	RTI
No Operation	NOP
Jump to Subroutine	JSR
Jump Unconditional	JMP

IMPLIED INSTRUCTIONS

Since the accumulator and all other registers are located in RAM, many implied instructions exist. Some of the instructions recognized and translated by the assembler are shown below:

Mnemonic	Becomes	Mnemonic	Becomes
ASLA	ADD $FF	INCX	INC $80
BHS	BCC	INCY	INC $81
BLO	BCS	LDXI	MVI $80 DATA
CLRA	SUB $FF	LDYI	MVI $81 DATA
CLRX	MVI $80 #0	NOP	BEQ (PC) +1
CLRY	MVI $81 #0	TAX	STA $80
DECA	DEC $FF	TAY	STA $81
DECX	DEC $80	TXA	LDA $80
DECY	DEC $81	TYA	LDA $81
INCA	INC $FF		

Some examples of valuable instructions not specifically recognized by the assembler are shown below:

Mnemonic	Meaning
BCLR 7,$FF	Ensures A is plus
BSET 7, $FF	Ensures A is minus
BRCLR 7, $FF	Branch if A is plus
BRSET 7, $FF	Branch if A is minus
BRCLR 7, $80	Branch if X is plus (BXPL)
BRSET 7, $80	Branch if X is minus (BXMI)
BRCLR 7, $81	Branch if Y is plus (BYPL)
BRSET 7, $81	Branch if Y is minus (BYMI)

OPCODE MAP

Table 1 is a listing of all the instruction set opcodes applicable to the MC68HC04J2 MCU.

ADDRESSING MODES

The MCU has nine different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. It deals with objects in three different address spaces: program space, data space, and stack space. The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is located in program ROM. It is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution, such as a constant used to initialize a loop counter.

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the 256 bytes in memory with a single two-byte instruction.

SHORT DIRECT

In the short direct addressing mode, the MCU has four locations in data space RAM it can use, ($80, $81, $82, and $83). The opcode determines the data space RAM location, and the instruction is only one byte. Short direct addressing is a subset of the direct addressing mode. The X and Y registers are at locations $80 and $81, respectively.

EXTENDED

In the extended addressing mode, the effective address of the argument is obtained by concatenating the four least-significant bits of the opcode with the byte following the opcode to form a 12-bit address. Instructions using the extended addressing mode, such as JMP or JSR, are capable of branching anywhere in program space. An extended addressing mode instruction is two bytes long.

RELATIVE

The relative addressing mode is only used in conditional branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −15 to +16 from the opcode address. The programmer need not calculate the offset when using the Motorola assembler, since it calculates the proper offset and checks to see that it is within the span of the branch.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte in which the specified bit is to be set or cleared. Thus, any bit in the 256 locations of data space memory that can be written to can be set or cleared with a single two-byte instruction.

CAUTION

The corresponding DDRs for ports A and B are write-only registers (registers at $04 and $05). A read operation on these registers is undefined. Since BSET and BCLR are read-modify-write functions, they cannot be used to set or clear a DDR bit; all "unaffected" bits would be set. Write all DDR bits in a port using a single-store instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit that is to be tested and its condition (set or clear) is included in the opcode. The data space address of the byte to be tested is in the single byte immediately following the opcode byte. The third byte is sign extended to twelve bits and becomes the offset added to the PC if the condition is true. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry flag.

REGISTER-INDIRECT

In the register-indirect addressing mode, the operand is at the address in data space pointed to by the contents of one of the indirect registers, X or Y. The particular indirect register is selected by bit 4 of the opcode. Bit 4 decodes into an address that represents the register, $80 or $81. A register-indirect instruction is one byte long.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. These instructions are one byte long.

Table 1. Opcode Map

Hi / Low	Branch Instructions 0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111	Register/Memory, Control, and Read/Modify/Write Instructions 8 1000	9 1001	A 1010	B 1011	Bit Manipulation Instructions C 1100	D 1101	Register/Memory and Read/Modify/Write E 1110	F 1111	Hi / Low
0 0000	BNE REL	BNE REL	BEQ REL	BEQ REL	BCC REL	BCC REL	BCS REL	BCS REL	JSRn EXT	JMPn EXT	•	MVI IMM	BRCLR0 BTB	BCLR0 BSC	LDA R IND	LDA R IND	0 0000
1 0001	BNE REL	BNE REL	BEQ REL	BEQ REL	BCC REL	BCC REL	BCS REL	BCS REL	JSRn EXT	JMPn EXT	•	•	BRCLR1 BTB	BCLR1 BSC	STA R IND	STA R IND	1 0001
2 0010	BNE REL	BNE REL	BEQ REL	BEQ REL	BCC REL	BCC REL	BCS REL	BCS REL	JSRn EXT	JMPn EXT	•	RTI INH	BRCLR2 BTB	BCLR2 BSC	ADD R IND	ADD R IND	2 0010
3 0011	BNE REL	BNE REL	BEQ REL	BEQ REL	BCC REL	BCC REL	BCS REL	BCS REL	JSRn EXT	JMPn EXT	•	RTS INH	BRCLR3 BTB	BCLR3 BSC	SUB R IND	SUB R IND	3 0011
4 0100	BNE REL	BNE REL	BEQ REL	BEQ REL	BCC REL	BCC REL	BCS REL	BCS REL	JSRn EXT	JMPn EXT	•	COMA INH	BRCLR4 BTB	BCLR4 BSC	CMP R IND	CMP R IND	4 0100
5 0101	BNE REL	BNE REL	BEQ REL	BEQ REL	BCC REL	BCC REL	BCS REL	BCS REL	JSRn EXT	JMPn EXT	•	ROLA INH	BRCLR5 BTB	BCLR5 BSC	AND R IND	AND R IND	5 0101
6 0110	BNE REL	BNE REL	BEQ REL	BEQ REL	BCC REL	BCC REL	BCS REL	BCS REL	JSRn EXT	JMPn EXT	•	STOP INH	BRCLR6 BTB	BCLR6 BSC	INC R IND	INC R IND	6 0110
7 0111	BNE REL	BNE REL	BEQ REL	BEQ REL	BCC REL	BCC REL	BCS REL	BCS REL	JSRn EXT	JMPn EXT	•	WAIT INH	BRCLR7 BTB	BCLR7 BSC	DEC R IND	DEC R IND	7 0111
8 1000	BNE REL	BNE REL	BEQ REL	BEQ REL	BCC REL	BCC REL	BCS REL	BCS REL	JSRn EXT	JMPn EXT	INC SD	DEC SD	BRSET0 BTB	BSET0 BSC	LDA IMM	LDA DIR	8 1000
9 1001	BNE REL	BNE REL	BEQ REL	BEQ REL	BCC REL	BCC REL	BCS REL	BCS REL	JSRn EXT	JMPn EXT	INC SD	DEC SD	BRSET1 BTB	BSET1 BSC	#	STA DIR	9 1001
A 1010	BNE REL	BNE REL	BEQ REL	BEQ REL	BCC REL	BCC REL	BCS REL	BCS REL	JSRn EXT	JMPn EXT	INC SD	DEC SD	BRSET2 BTB	BSET2 BSC	ADD IMM	ADD DIR	A 1010
B 1011	BNE REL	BNE REL	BEQ REL	BEQ REL	BCC REL	BCC REL	BCS REL	BCS REL	JSRn EXT	JMPn EXT	INC SD	DEC SD	BRSET3 BTB	BSET3 BSC	SUB IMM	SUB DIR	B 1011
C 1100	BNE REL	BNE REL	BEQ REL	BEQ REL	BCC REL	BCC REL	BCS REL	BCS REL	JSRn EXT	JMPn EXT	LDA SD	STA SD	BRSET4 BTB	BSET4 BSC	CMP IMM	CMP DIR	C 1100
D 1101	BNE REL	BNE REL	BEQ REL	BEQ REL	BCC REL	BCC REL	BCS REL	BCS REL	JSRn EXT	JMPn EXT	LDA SD	STA SD	BRSET5 BTB	BSET5 BSC	AND IMM	AND DIR	D 1101
E 1110	BNE REL	BNE REL	BEQ REL	BEQ REL	BCC REL	BCC REL	BCS REL	BCS REL	JRSn EXT	JMPn EXT	LDA SD	STA SD	BRSET6 BTB	BSET6 BSC	#	INC DIR	E 1110
F 1111	BNE REL	BNE REL	BEQ REL	BEQ REL	BCC REL	BCC REL	BCS REL	BCS REL	JSRn EXT	JMPn EXT	LDA SD	STA SD	BRSET7 BTB	BSET7 BSC	#	DEC DIR	F 1111

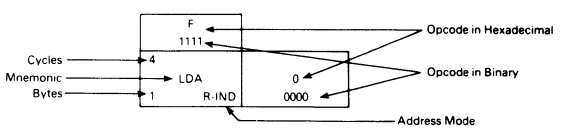
Abbreviations for Address Modes

INH	Inherent
S-D	Short Direct
B-T-B	Bit Test and Branch
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
R-IND	Register Indirect

* Indicates Instruction Reserved for Future Use

\# Indicates Illegal Instruction

LEGEND



ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{DD}	−0.3 to +7.0	V
Input Voltage	V_{in}	V_{SS}−0.3 to V_{DD}+0.3	V
Current drain per pin Excluding V_{DD} and V_{SS}	I	10	mA
Total current for sink Ports A, B, C EXTAL, TiM source	I i	30 15	mA
Operating Temperature Range (Comm.)	T_A	0 to 70	°C
Operating Temperature Range (Ind.)	T_A	−40 to +85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature Plastic	T_J	150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields. However, it is advised that normal precautions be taken to avoid applications of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}) \leq V_{DD}$. Reliability of operation is enhanced if unused inputs except EXTAL are connected to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic	θ_{JA}	70	°C/W

3

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

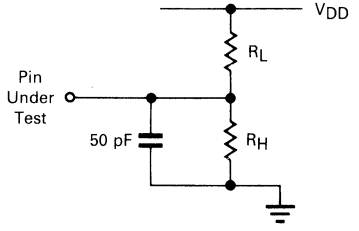
An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.



V_{DD} = +4.5V
I_{OL}/I_{OH} = 800 µA
$R_L = R_H$ = 4.6 kΩ

V_{DD} = +2.7V
I_{OL}/I_{OH} = 200 µA
$R_L = R_H$ = 10.5 kΩ

V_{DD} = +2.0V
I_{OL}/I_{OH} = 100 µA
$R_L = R_H$ = 16 kΩ

Figure 9. Equivalent Test Load

CONTROL TIMING CHARACTERISTICS

Characteristic	Symbol	Min	Typ	Max	Unit
(V_{DD} = +5 Vdc ±10%, V_{SS} = 0 Vdc; T_A = 0°C to 70°C					
Oscillator Frequency	f_{osc}	0	—	11.0	MHz
PHI1 Clock Frequency	f_{CL}	0	—	5.5	MHz
Cycle Time (Min)	t_{cyc}	2.2	—	—	µs
$\overline{IRQ}$ Pulse Width	t_{IWL}	2×t_{cyc}	—	—	µs
$\overline{RESET}$ Pulse Width	RWL	2×t_{cyc}	—	—	µs
Oscillator Clock Pulse Width	t_{OL}, t_{OH}	45	—	—	ns
V_{DD} = +3 Vdc ±10%, V_{SS} = 0 Vdc, T_A = 0°C to 70°C					
Oscillator Frequency	f_{osc}	—	—	11.0	MHz
PHI1 Clock Frequency	f_{CL}	—	—	4.2	MHz
Cycle Time (Min)	t_{cyc}	2.9	—	—	µs
$\overline{IRQ}$ Pulse Width	t_{IWL}	2×t_{cyc}	—	—	µs
$\overline{RESET}$ Pulse Width	t_{RWL}	2×t_{cyc}	—	—	µs
Oscillator Clock Pulse Width	t_{OL},t_{OH}	45	—	—	ns
V_{DD} = +2.2 Vdc ±10%, V_{SS} = 0 Vdc, T_A = 0°C to 70°C					
Oscillator Frequency	f_{osc}	0	—	8.4	MHz
PHI1 Clock Frequency	f_{CL}	0	—	2.1	MHz
Cycle Time (Min)	t_{cyc}	5.7	—	—	µs
$\overline{IRQ}$ Pulse Width	t_{IWL}	2×t_{cyc}	—	—	µs
$\overline{RESET}$ Pulse Width	t_{RWL}	2×t_{cyc}	—	—	µs
Oscillator Clock Pulse Width	t_{OL},t_{OH}	45	—	—	ns

NOTE: 2 V operation is a user-selectable option only. Prior consultation with the factory is required.

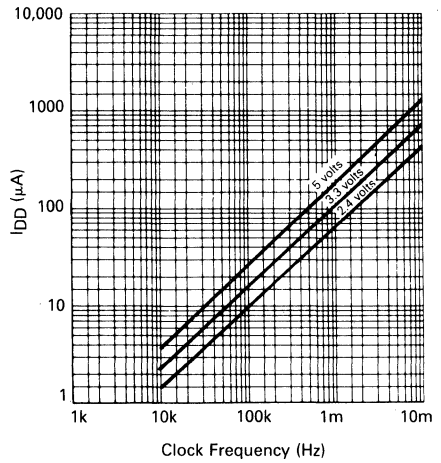


Figure 10. Typical RUN Current vs Clock Frequency (f_{CL})

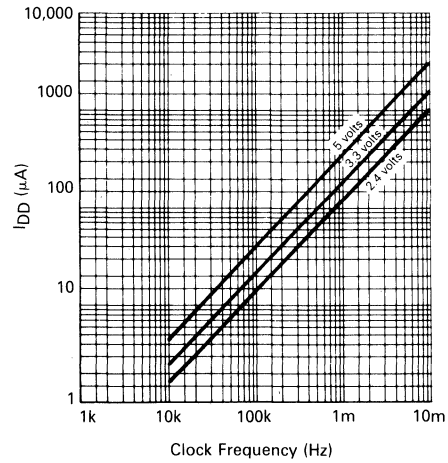


Figure 11. Typical WAIT Current vs Clock Frequency (f_{CL})

DC ELECTRICAL CHARACTERISTICS (Typical pull-down sink current for $V_{out} = V_{DD}$ is 50 μA.)

Characteristic		Symbol	Min	Typ	Max	Unit
V_{DD} = +5 Vdc ±10%, V_{SS} = 0 Vdc, T_A = 0°C to 70° C						
Output Voltage, I_{Load} ⟨10.0 μA		V_{OL} V_{OH}	— $V_{DD}-0.1$	— —	0.1 —	V
Output High Voltage, I_{Load} = +800 μA) Output Low Voltage, I_{Load} = +800 μA)	Ports, TIM Ports, TIM	V_{OH} V_{OL}	$V_{DD}-0.4$ —	— —	— 0.4	V
Input High Voltage	Ports, TIM, XTAL, MDS IRQ, RESET	V_{IH} V_{IH}	$0.7 \times V_{DD}$ $0.8 \times V_{DD}$	— —	V_{DD} V_{DD}	V
Input Low Voltage	Ports, TIM, XTAL, MDS IRQ, RESET	V_{IL} V_{IL}	V_{SS} V_{SS}	— —	$0.3 \times V_{DD}$ $0.2 \times V_{DD}$	V
Total Supply Current C_L = 50 pF, Ports, TIM, No dc load, $t_{cyc} = 1/f_{CL}$ (max), V_{IL} = 0.2 V, $V_{IH} = V_{DD} - 0.2$ V	RUN WAIT* STOP*	I_{DD} I_{DD} I_{DD}	— — —	2 0.5 3	3 1 5	mA mA μA
I/O Ports Input Leakage $V_{SS} \langle V_I \langle V_{DD}$		I_{IL}	—	—	±1	μA
Input Current	RESET, IRQ, TIM	I_{in}	—	—	±1	μA
Capacitance per Pin	PORTS (as Input or Output) RESET, IRQ, TIM, XTAL, MDS	C_{out} C_{in}	— —	— —	12 8	pF
V_{DD} = +3 Vdc ±10%, V_{SS} = 0 Vdc, T_A = 0°C to 70°C						
Output Voltage, I_{Load} ⟨10.0 μA		V_{OL} V_{OH}	— $V_{DD}-0.1$	— —	0.1 —	V
Output High Voltage, I_{Load} = −200 μA) Output Low Voltage, I_{Load} = +200 μA)	Ports, TIM Ports, TIM	V_{OH} V_{OL}	$V_{DD}-0.3$ —	— —	— 0.3	V
Input High Voltage	Ports, TIM, XTAL, MDS IRQ, RESET	V_{IH} V_{IH}	$0.7 \times V_{DD}$ $0.8 \times V_{DD}$	— —	V_{DD} V_{DD}	V
Input Low Voltage	Ports, TIM, MDS, XTAL IRQ, RESET	V_{IL} V_{IL}	V_{SS} V_{SS}	— —	$0.3 \times V_{DD}$ $0.2 \times V_{DD}$	V
Total Supply Current C_L = 50 pF, Ports, TIM, No dc load, $t_{cyc} = 1/f_{CL}$ (Max), V_{IL} = 0.2 V, $V_{IH} = V_{DD} - 0.2$ V	RUN WAIT* STOP*	I_{DD} I_{DD} I_{DD}	— — —	0.8 0.3 1.5	1.5 0.5 4	mA mA μA
I/O Ports Input Leakage $V_{SS} \langle V_I \langle V_{DD}$		I_{IL}	—	—	±1	μA
Input Current	RESET, IRQ, TIM	I_{in}	—	—	±1	μA
Capacitance per Pin	PORTS (as Input or Output) RESET, IRQ, TIM, XTAL, MDS	C_{out} C_{in}	— —	— —	12 8	pF
V_{DD} = +2.2 Vdc ±10%, V_{SS} = 0 Vdc, T_A = 0°C to 70°C						
Output Voltage, I_{Load} ⟨10.0 μA		V_{OL} V_{OH}	— $V_{DD}-0.1$	— —	0.1 —	V
Output High Voltage, I_{Load} = −100μA) Output Low Voltage, I_{Load} = +100 μA)	Ports, TIM Ports, TIM	V_{OH} V_{OL}	$V_{DD}-0.3$ —	— —	— 0.3	V
Input High Voltage	Ports, TIM, XTAL, MDS IRQ, RESET	V_{IH} V_{IH}	$0.7 \times V_{DD}$ $0.8 \times V_{DD}$	— —	V_{DD} V_{DD}	V
Input Low Voltage	Ports, TIM, MDS, XTAL IRQ, RESET	V_{IL} V_{IL}	V_{SS} V_{SS}	— —	$0.3 \times V_{DD}$ $0.2 \times V_{DD}$	V
Total Supply Current C_L = 50 pF, Ports, TIM, No dc load, $t_{cyc} = 1/f_{CL}$ (Max), V_{IL} = 0.2v, $V_{IH} = V_{DD} - 0.2$ V	RUN WAIT* STOP*	I_{DD} I_{DD} I_{DD}	— — —	0.6 0.2 1	1 0.3 3	mA mA μA
I/O Ports Input Leakage $V_{SS} \langle V_I \langle V_{DD}$		I_{IL}	—	—	±1	μA
Input Current	REST, IRQ, TIM	I_{in}	—	—	±1	μA
Capacitance per Pin	PORTS (as Input or Output) RESET, IRQ, TIM, XTAL, MDS	C_{out} C_{in}	— —	— —	12 8	pF

*Measured under the following conditions:
- All ports and timer pin are configured as input
- XTAL is driven by a square wave input
- EXTAL is open circuit
- port pull downs not enabled

NOTE: Typical pull-down sink current for $V_{out} = V_{DD}$ is 50 μA.

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola using the following media:

MDOS™, disk file
MS™-DOS disk file (360K)
EPROM(s) 2516, 2716, 2532, 2732

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, a sales person, or a Motorola representative.

FLEXIBLE DISKS

Several types of flexible disks (MDOS or MS-DOS disk file) may be submitted for pattern generation. They should be programmed with the customer program, using positive logic sense for address and data. The diskette should be clearly labeled with the customer's name, date, project or product name, and the filename containing the pattern.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to expedite the process in case of any difficulty with the pattern file.

MDOS Disk File

MDOS is Motorola's Disk Operating System available on the EXORciser® development system. The disk media submitted must be a single-sided, single-density, 8-inch MDOS compatible floppy diskette. The diskette must contain the minimum set of MDOS system files in addition to the pattern file.

The .LO output of the M6804 cross assembler should be furnished. In addition, the file must be produced using the ROLLOUT command, so that it contains the absolute image of the M6804 memory. It is necessary to include the entire memory image of both program and data space. All unused bytes, including those in the user space, must be set to logic zero.

MS-DOS Disk File

MS-DOS is Microsoft's Disk Operating System. Disk media submitted must be standard density (360K), double-sided 5 1/4 inch compatible floppy diskette. The diskette must contain the object file code in Motorola's S-record format. The S-record format is a character-based object file format generated by M6804 cross assemblers and linkers on IBM PC style machines.

EPROMS

Four K of EPROM are necessary to contain the entire MC68HC04J2 program. Two 2516 or 2716 type EPROMs or a single 2532 or 2732 type EPROM can be submitted for pattern generation. The EPROM is programmed with the customer's program using positive logic sense for address and data. Submissions on two EPROMs must be clearly marked. All unused bytes, including the user's space, must be set to zero.

If the MC68HC04J2 MCU ROM pattern is submitted on one 2532 or 2732 EPROM, or on two 2516 or 2716 type EPROMs, memory map addressing is one-for-one. The data space ROM runs from EPROM address $018 to $05F, and program space ROM runs from EPROM address $C10 to $FF7 with vectors from $FFC to $FFF.

For shipment to Motorola, EPROMs should be placed in a conductive IC carrier and packed securely. Styrofoam is not acceptable for shipment.

Verification Media

All original pattern media, EPROMs or floppy disks, are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked and the verification form completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for the creation of the customer mask. To aid in the verification process, Motorola will program *customer supplied* blank EPROM(s) or DOS disks from the data file used to create the custom mask.

ROM Verification Units (RVUs)

Ten MCUs containing the customers ROM pattern will be sent for program verification. These units will have been made using the custom mask, but are for the purpose of ROM verification only. For expediency, the MCUs are unmarked, packaged in ceramic, and tested with five volts at room temperature. These RVUs are free with the minimum order quantity, but are not production parts. These RVUs are not guaranteed by Motorola Quality Assurance.

Ordering Information

The following table provides generic information pertaining to the package type, temperature, and order numbers for the MC68HC04P3.

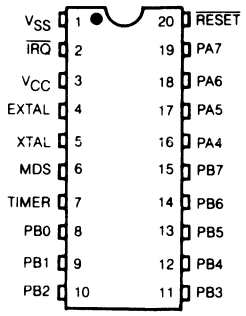
Ordering Information

Package Type	Temperature	Order Number
Plastic (P Suffix)	0°C to 70°C	MC68HC04J2P
	−40°C to +85°C	MC68HC04J2CP

MDOS is a trademark of Motorola Inc.
MS™-DOS is a trademark of Microsoft, Inc.
EXORciser is a registered trademark of Motorola Inc.
IBM is a registered trademark of International Business Machines Corporation.

MECHANICAL DATA

PIN ASSIGNMENTS



3

Technical Summary
8-Bit Microcomputer Unit

MC68HC04J3 HCMOS microcomputer unit (MCU) device is a member of the M6804 Family of single-chip microcomputers. This device is tremendously versatile and cost effective. These qualities are based on the MCU's simple design and ability to process 8-bit variables, one bit at a time.

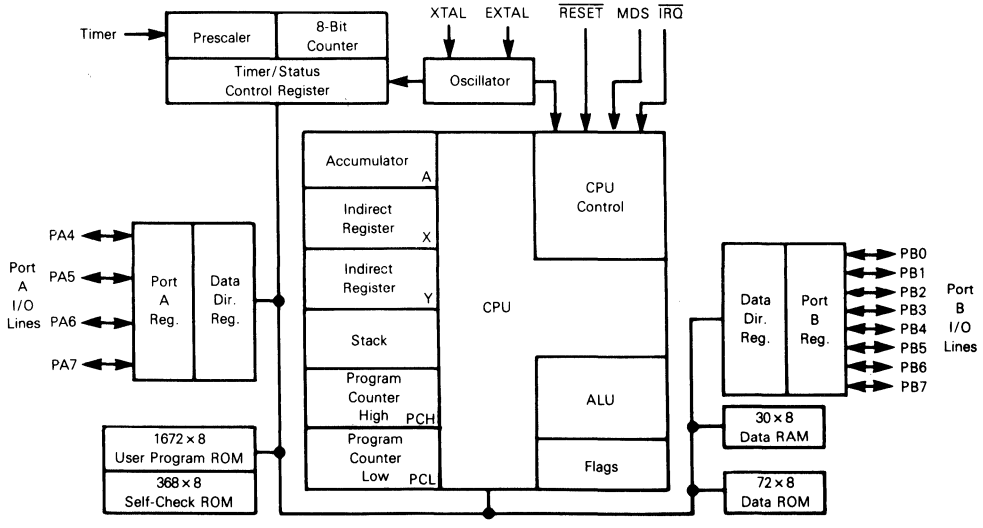
This technical summary contains limited information on the MC68HC04J3. For detailed information, refer to the advanced information data sheet for the MC68HC04J2, MC68HC04J3, and MC68HC04P3 8-bit microcomputers (MC68HC04J2/D) or to the *M6804 MCU Manual* (DLE404/D).

Major hardware and software features of the MC68HC04J3 MCU are:

- On-Chip Clock Generator
- Memory Mapped I/O
- Software Programmable 8-Bit Timer with 7-Bit Prescaler
- Single Instruction Memory Examine/ Change
- 72 Bytes of Data ROM
- 30 Bytes of User RAM
- User Selectable Input Drive Options
- Optional Pull Down Devices on I/O Ports
- Mask Selectable Edge- or Level-Sensitive Interrupt Pin
- True Bit Manipulation
- Bit Test and Branch Instruction
- 368 Bytes Self-Check ROM
- Conditional Branches
- Timer Pin is Software Programmable as Clock Input or Timer Output
- 1672 Bytes of User Program ROM

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

V_{DD} AND V_{SS}

Power is supplied to the microcomputer using these two pins. V_{DD} is power, and V_{SS} is ground.

$\overline{IRQ}$

This pin provides the capability for asynchronously applying an external interrupt to the microcomputer. A pullup resistor on this pin is a manufacturing mask option.

EXTAL AND XTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a resistor/capacitor combination, or an external signal is connected to these pins to provide a system clock. Selection is made by a manufacturing mask option. The different clock generator options are shown in Figure 1, along with crystal specifications.

Internal Clock Options

The crystal oscillator start-up time is a function of many variables. To ensure rapid oscillator start-up, neither the crystal characteristics nor load capacitances should exceed recommendations. When using the on-board oscillator, the MCU should remain in a reset condition, with the $\overline{RESET}$ pin voltage below V_{IRES+}, until the oscillator has stabilized at its operating frequency.

TIMER

Two TIMER input modes as well as an output mode are available. In the input modes, the TIMER pin is configured as either a TIMER enable, or as the TIMER clock. In the output mode, the TIMER pin may generate transitions upon each occurrence of timer underflow.

$\overline{RESET}$

The $\overline{RESET}$ pin is used to restart the processor to the beginning of a program. The program counter is loaded with the address of the restart vector. This should be a jump instruction to the first instruction of the main program. Together with the MDS pin, the $\overline{RESET}$ pin selects the operating mode of the MCU. A pullup resistor on this pin is a manufacturing mask option.

MDS

The mode select (MDS) pin places the MCU into special operating modes. When this pin is logic high at the exit of the reset state, the decoded state of PA6 and PA7 is latched to determine the operating mode. This choice can be either the single-chip, self-check, or ROM verify mode. However, if MDS is logic low at the end of the reset state, the single-chip operating mode is automatically selected. No external diodes, switches, transistors, etc. are required for single-chip mode selection.

INPUT/OUTPUT LINES (PA4-PA7, PB0-PB7)

These 12 lines are arranged into one 4-bit port (A) and one 8-bit port (B). All lines are programmable as either inputs or outputs under software control of the data direction registers.

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

There are 12 input/output pins. The 12 bidirectional lines can be selected to have internal pulldowns at the time of manufacture. All pins of each port are programmable as inputs or outputs under the control of the data direction registers (DDR).

The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic one for output or a logic zero for input, as shown in Figure 2. When the registers are programmed as outputs, the latched data is readable regardless of the logic levels at the output pin due to output loading.

All the I/O pins are CMOS compatible as both inputs and outputs. Their standard configuration as outputs is three-state drive. Port B outputs are LED compatible. In addition, certain pins of both ports may be ordered equipped with pull down resistors.

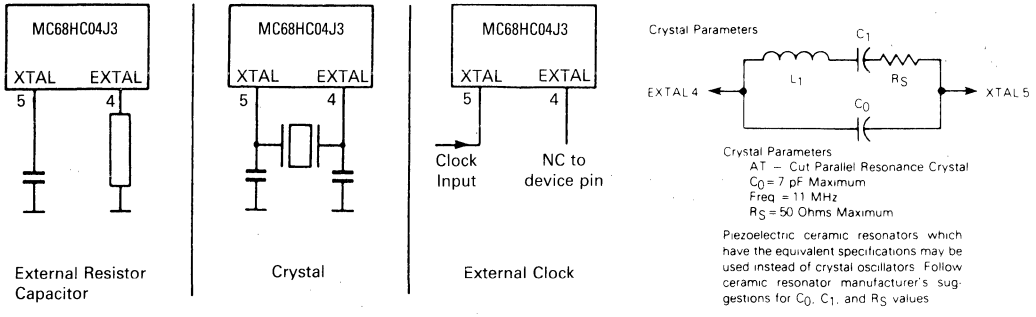
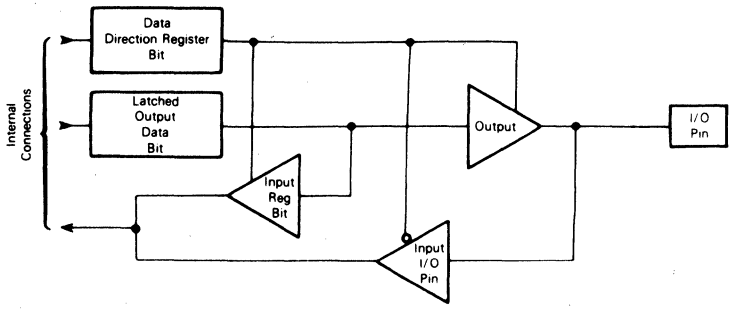


Figure 1. Clock Generator Options and Crystal Parameters



Data Direction Register Bit	Output Data Bit	Output State	Input To MCU
1	0	0	0
1	1	1	1
0	X	Hi-Z	Pin

Figure 2. Typical I/O Port Circuitry

Any write to a port writes to all of its data bits even though the port DDR may be set to input. This can be used as a tool to initialize the data registers and avoid undefined outputs. However, care must be exercised when using read-modify-write instructions. The data read corresponds to the pin level if the DDR is an input or to the latched output data when the DDR is an output.

Pull Down Device Option

The use of pull down devices on particular groupings of I/O ports is a manufacturing mask option available to the user. It is of use in applications where keyboards are interfaced directly to the MCU and similar situations. This option is available in the following configurations:

I/O Port	Resistor-Option Pin Groupings
Port A	PA4-PA7
Port B	PB3-PB7, PB4-PB7, PB1-PB2, PB0

Port Data Registers ($00, $01)

The port data registers are not initialized on reset. These registers should be initialized before changing the DDR bits to avoid undefined levels.

The source of data read from the port register is either the port I/O pin or previously latched output data. The source depends upon the contents of the corresponding DDR. The destination of data written to the port data register is an output data latch. If the corresponding DDR for the port I/O pin is programmed as an output, the data appears on the port pin.

Port A ($00)

7	6	5	4	3	2	1	0
				X	X	X	X

Port B ($01)

7	6	5	4	3	2	1	0

With regard to Port A only, the four MSB bits are unused. These are "don't care" (X) bits when written to but are always logic high when read.

Port Data Direction Registers ($04, $05)

Port DDRs configure the port pins as either outputs or inputs. Each port pin can be programmed individually to function as input or output. A zero in the pin's corresponding DDR bit programs it as an input; a logic one programs it as an output. On reset, all the DDRs are initialized to a logic zero state to put the ports in the input mode.

Port A ($04)

7	6	5	4	3	2	1	0
				0	0	0	0

Port B ($05)

7	6	5	4	3	2	1	0

With regard to Port A DDR only, the four MSB bits are cleared after reset. These bits must not be set (logic one).

MEMORY

The MCU memory map (Figure 3) consists of 4352 bytes of addressable memory, I/O register locations, and stack space. This MCU has three separate memory spaces: program space, data space, and stack space.

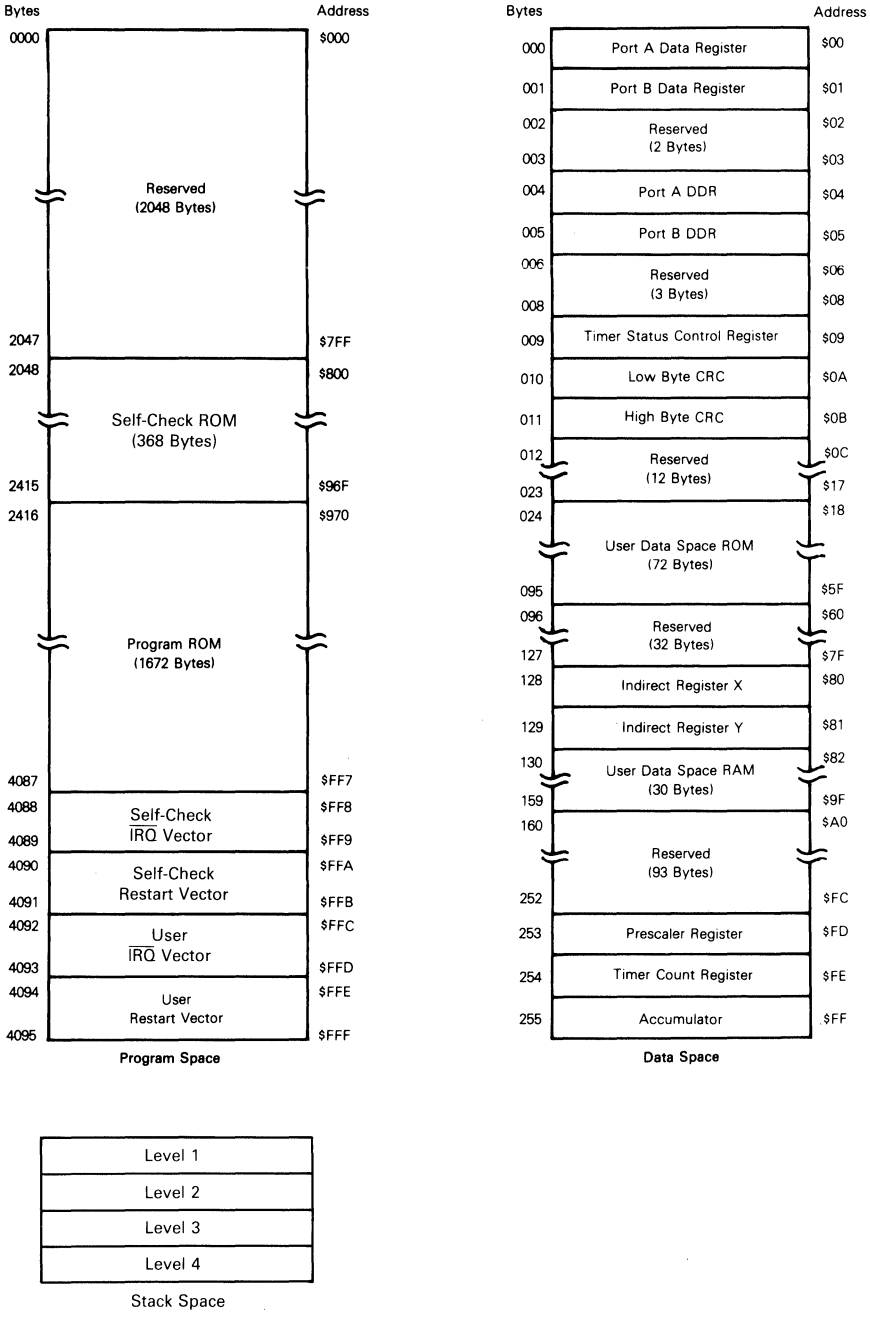


Figure 3. Memory Map

The MCU is capable of addressing 4096 bytes of program space memory with its program counter and 256 bytes of data space memory with its instructions. Program space memory includes self-check ROM, program ROM, self-check and user program vectors, and reserved memory locations.

A non-accessible subroutine stack space RAM is provided. This stack space consists of a last-in-first-out (LIFO) register. This register is used with inherent addressing to stack the return address for subroutines.

Indirect X and Y register locations $80 and $81 are generally used as pointers for such tasks as indirect addressing to data space locations. Short direct addressing allows access to the four data space addresses $80-$83 with single byte opcodes. The operations allowed are increment, decrement, load, and store. Data space locations $82 and $83 can be used for 8-bit counter locations.

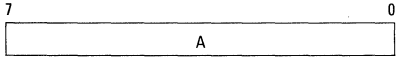
Program ROM Protect

A manufacturing mask option available to the user enables program ROM protection. Enabled, this option prevents the ROM contents from being output during self-check/ROM verify. This option does not prevent a go, no-go test of the ROM contents using the ROM verify mode.

REGISTERS

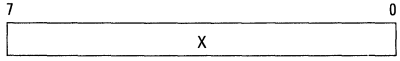
ACCUMULATOR (A)

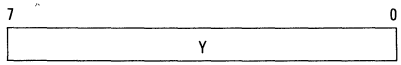
The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



INDIRECT REGISTERS (X,Y)

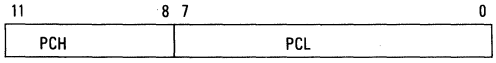
These two registers are used to maintain pointers to other memory locations in data space. They are used in the register-indirect addressing mode and can be accessed with the direct, indirect, short direct, or bit set/clear modes.





PROGRAM COUNTER (PC)

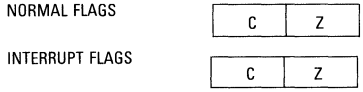
The program counter is a 12-bit register that contains the address of the next byte to be fetched from program space. The program counter is contained in low byte (PCL) and high nibble (PCH).



FLAGS (C,Z)

The first flag, the carry (C) bit, is set on a carry or borrow out of the arithmetic logic unit (ALU). It is cleared if the arithmetic operation does not result in a carry or borrow. The C bit is also set to the value of the bit tested in a bit test instruction. It participates in the rotate left (ROLA) instruction, as well.

The second flag, the zero (Z) bit, is set if the result of the last arithmetic or logic operation was equal to zero. Otherwise, it is cleared. Bit test instructions do not affect the Z bit.



There are two sets of these flags. One set is for interrupt processing (the interrupt mode flags). The other set is for normal operations (the program mode flags). When an interrupt occurs, a context switch is made from the program flags to the interrupt flags. An RTI forces the context switch back. While in either mode, only the flags for that mode are available. A context switch does not affect the value of the C or Z bits. Both sets of flags are cleared by $\overline{\text{RESET}}$.

STACK

A last-in-first-out (LIFO) stack is incorporated in the MCU that eliminates the need for a stack pointer. This non-accessible subroutine stack space is implemented in separate RAM, 12-bits wide. Whenever a subroutine call or interrupt occurs, the contents of the PC are shifted into the top register of the stack. At the same time, the top register is shifted one level deeper. This happens to all registers, with the bottom register falling out of the stack.

Whenever a return from subroutine or interrupt occurs, the top register is shifted into the PC and all lower registers are shifted one level higher. The stack RAM is four levels deep. If the stack is pulled more than four times with no pushes, then the address that was stored in the bottom level of the stack is shifted into the PC.

CRC Registers

Two eight bit registers are implemented in RAM primarily as self-check and ROM verify modes. The two registers are memory mapped in data space at addresses $0A (CRC low), and $0B (CRC high).

Provided no write or read/modify/write operation is used to change the contents of these two locations, the registers are configured to perform CRC calculations. By simply reading a register, a pseudo-random number can be generated.

If a write or a read/modify/write is performed on addresses $0A or $0B, then the CRC circuitry is disabled. Both registers can be used as RAM locations until the next $\overline{\text{RESET}}$. $\overline{\text{RESET}}$ enables the CRC circuitry again.

SELF CHECK

The MCU implements two forms of internal check, self check and ROM verify. Self check performs an extensive

functional check of the MCU using a signature analysis technique. ROM verify uses a similar method to check the contents of program ROM.

Self-check mode is selected by holding the MDS and PA7 pins logic high, and PA6 logic low as $\overline{\text{RESET}}$ goes low to high. ROM verify mode is entered by holding MDS, PA7, and PA6 logic high as $\overline{\text{RESET}}$ goes low to high. Unimplemented program space ROM locations are also tested. Monitoring the self-check mode's stages for successful completion requires external circuitry.

RESET

The MCU can be reset by initial power up or by external reset input ($\overline{\text{RESET}}$).

POWER-ON-RESET (POR)

During a power-on-reset, the timer is used to count 1920 external clock cycles. This allows the oscillator to stabilize before releasing the internal reset, irrespective of the state of the $\overline{\text{RESET}}$ pin. If the $\overline{\text{RESET}}$ pin is low at the end of the delay, the processor remains in the reset condition.

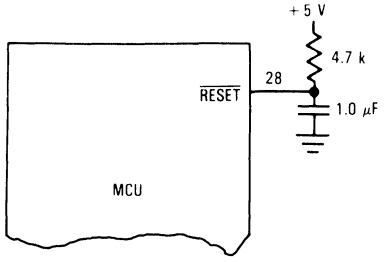


Figure 4. Power Up $\overline{\text{RESET}}$ Delay Circuit

$\overline{\text{RESET}}$

A reset can also be achieved by pulling the $\overline{\text{RESET}}$ pin to logic low for a minimum of two clock cycles. The delay is not implemented in this case.

INTERRUPT

There are two ways this MCU can be interrupted: by applying a logic low signal to the $\overline{\text{IRQ}}$ pin, or by a positive transition of the TMZ bit of TSCR with the ETI bit set. However, a manufacturing mask option determines whether the falling edge or the actual low level of the $\overline{\text{IRQ}}$ pin is sensed to indicate an interrupt.

EXTERNAL INTERRUPT EDGE-SENSITIVE OPTION

When the $\overline{\text{IRQ}}$ pin is pulled low, the internal interrupt request latch is set. Prior to each instruction fetch, this interrupt request latch is tested. If its output is low, an interrupt sequence is initated at the end of the current instruction, provided the interrupt mask is cleared. Figure contains a flowchart that illustrates the interrupt and instruction processing sequences.

The interrupt sequence consists of one cycle during which:

The interrupt request latch is cleared;

The interrupt mode flags are selected;

The program counter (PC) is saved on the stack;

The interrupt mask is set; and

The $\overline{\text{IRQ}}$ vector jump address is loaded into the PC.

The $\overline{\text{IRQ}}$ vector jump address is $FFC-$FFD in the single-chip mode and $FF8-$FF9 in the self-check mode. The contents of these locations are not decoded as an address to which the PC should jump. Instead, they are decoded like any other ROM word. So, it is essential that the vector contents specify a JMP instruction in addition to the starting address of the interrupt service routine. If required, this routine should save the values of the accumulator and the X and Y registers, since these values are not stored on the stack.

Internal processing of the interrupt continues until a return from interrupt (RTI) instruction is processed. During RTI, the interrupt mask is cleared and the program mode flags are selected. The next instruction of the program is then fetched and executed.

When STOP is processed, the interrupt mask is cleared and the oscillator stopped. Checks are made for either $\overline{\text{RESET}}$ or $\overline{\text{IRQ}}$. If $\overline{\text{RESET}}$ is detected, the $\overline{\text{RESET}}$ sequence is initiated. If $\overline{\text{IRQ}}$ is detected, the system oscillator is enabled along with the clock. In both cases, a delay is executed by the timer to allow oscillator stabilization before the CPU is enabled and the interrupt serviced.

When WAIT is processed, the interrupt mask is cleared and the CPU clock disabled. The interrupt latch is tested. Detection of $\overline{\text{RESET}}$ initiates the $\overline{\text{RESET}}$ sequence. Detection of $\overline{\text{IRQ}}$ or timer interrupt enables the CPU clock and initiates servicing of the interrupts.

When RTI is processed, the program counter is pulled from the stack. The program flags are selected and the interrupt mask cleared. The interrupt latch is then tested before the next instruction.

When the interrupt was initially detected and the interrupt sequence started, the interrupt request latch was cleared so that the next interrupt could be detected. This was done even as the first interrupt was being serviced. However, even though the second interrupt set the interrupt request latch during the first interrupt's processing, the second interrupt's sequence cannot begin until completion of the interrupt service routine for the first interrupt. Completion of an interrupt service routine is always accomplished using an RTI instruction to return to the main program. The interrupt mask, which is not directly available to the programmer, is cleared during the last cycle of the RTI instruction.

EXTERNAL INTERRUPT EDGE/LEVEL-SENSITIVE OPTION

The edge/level-sensitive option performs as described in the preceding section but adds the potential for level-sensitive operation. Level-sensitive operation tests the state of the $\overline{\text{IRQ}}$ and initiates an interrupt service routine if the $\overline{\text{IRQ}}$ pin is found to be logic low.

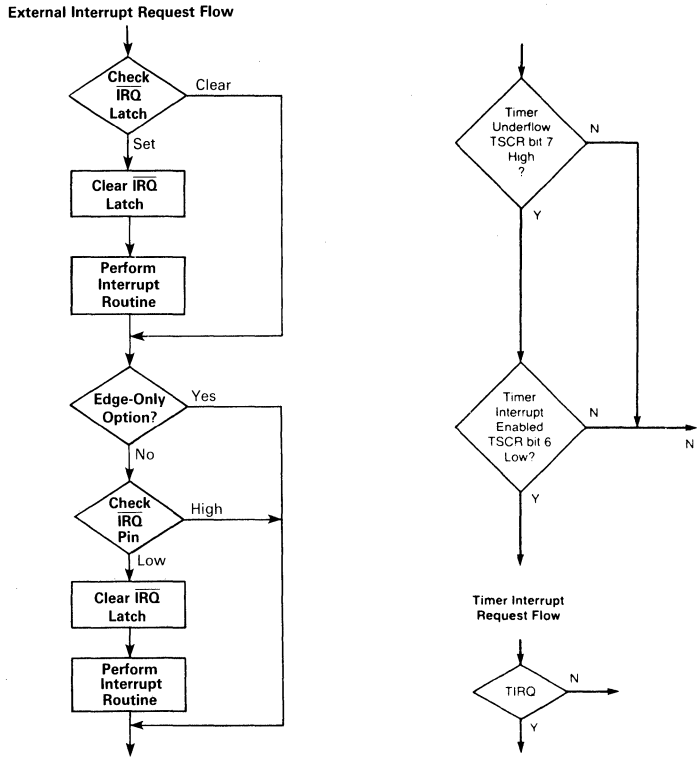


Figure 5. Interrupt Sequences

POWER UP AND TIMING

During the power up sequence, the interrupt mask is closed. This precludes any false interrupts. The PC is also loaded with the appropriate restart vector (jump instruction).

To open the interrupt mask, the user should do a JSR to an initialization subroutine that ends with an RTI instead of an RTS. The RTI opens the interrupt mask. Typical $\overline{\text{RESET}}$ and $\overline{\text{IRQ}}$ processes and their relationship to the interrupt mask are shown in Figure 7.

Maximum interrupt response time is eight machine cycles. This includes five cycles for the longest instruction plus one for stacking the PC and switching flags. Two additional cycles are used to synchronize $\overline{\text{IRQ}}$ input with the internal machine cycle frequency.

TIMER INTERRUPT

A timer interrupt is requested by a transition of the TMZ bit of the timer status/control register (TSCR) from logic low to high. Such a positive transition is caused either by the timer count register reaching the all zero state, or by any program instruction that writes a one to the TMZ bit.

The timer interrupt request is maskable by clearing bit 6 of the TSCR (ETI bit). ETI is cleared by $\overline{\text{RESET}}$.

During the interrupt routine, to determine whether an interrupt was caused externally or by the timer, it is necessary to test the state of the TMZ bit in the TSCR.

It is important to service a timer interrupt and clear the TMZ bit before the timer counter underflows again. Otherwise, because only a single interrupt can be latched, there is no way of telling how many timer interrupts occur while the original interrupt is being serviced.

LOW-POWER MODES

STOP

The STOP instruction places the MCU in its lowest power consumption mode. In the STOP mode, the internal oscillator is turned off, causing all internal processing and the timer to be halted. Current consumption is thus dropped to leakage levels.

Providing the supply voltage remains within data sheet limits, the contents of the TSCR, accumulator, and all data space RAM remain unchanged in STOP mode.

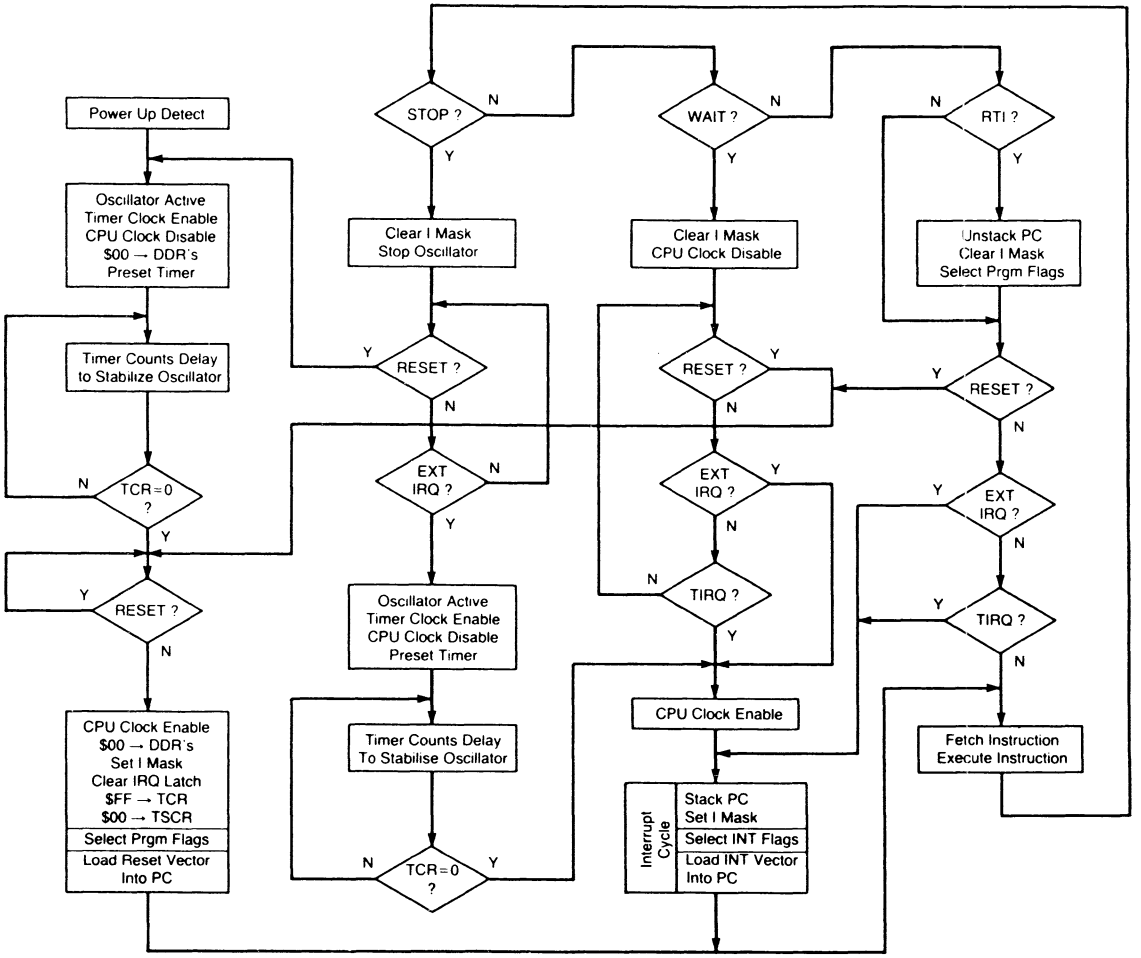


Figure 6. Instruction Processing Sequence

3

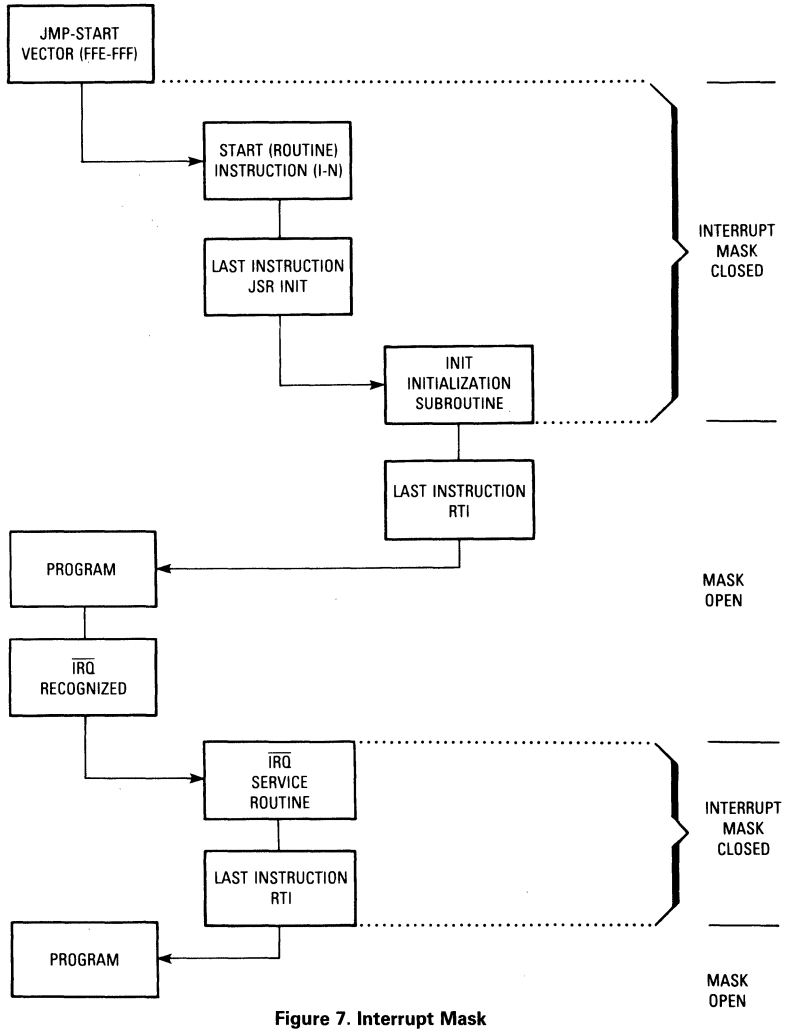


Figure 7. Interrupt Mask

Causing an interrupt or reset by pulling the $\overline{\text{RESET}}$ or $\overline{\text{IRQ}}$ pins low is the only way to bring the processor out of STOP mode. During this exit from STOP, the timer is used to provide the delay time necessary for the oscillator to stabilize. So, the prescaler and timer count register contents must be considered corrupted.

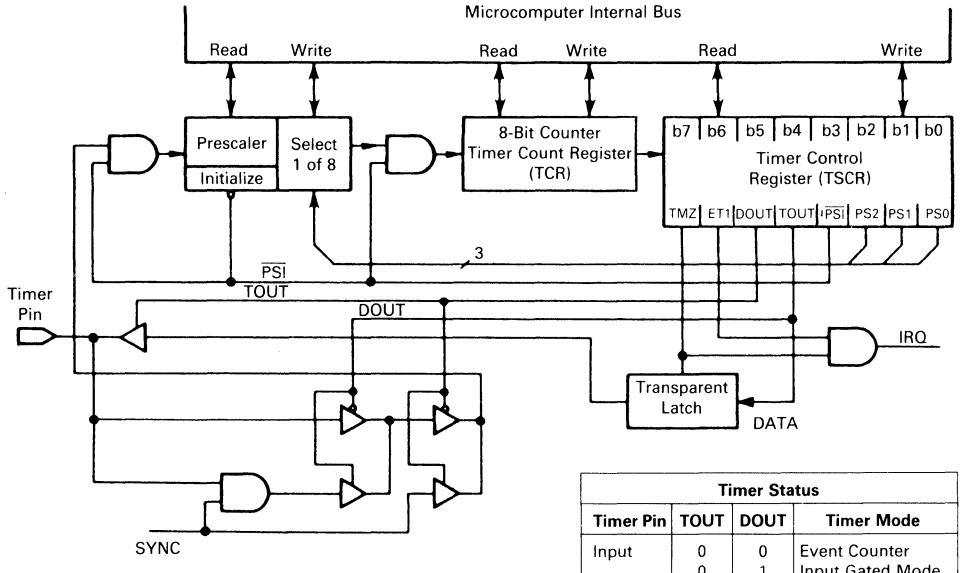
WAIT

The WAIT instruction places the MCU in a low power consumption mode, but the WAIT mode consumes somewhat more power than the STOP mode. In the WAIT mode, the internal clock is disabled from all internal circuitry except for the timer. So, all internal processing is halted. However, the timer continues to decrement normally if the PSI bit of TSCR is set.

During the WAIT mode, external interrupts are enabled. All other registers, memory, and I/O lines remain in their last state. Pulling the $\overline{\text{IRQ}}$ or $\overline{\text{RESET}}$ pin to logic low causes an exit from the WAIT mode. In addition, ETI bit of TSCR can be enabled by software prior to entering the WAIT state. This allows an exit from WAIT via a timer interrupt as well as via external interrupts.

TIMER

A block diagram of the MC68HC04J3 timer circuitry is shown in Figure 8. The timer logic in the MCU is comprised of a simple 8-bit counter called the timer counter. This counter is decremented by a 7-bit prescaler at a rate determined by the timer status/control register (TSCR).



Timer Status			
Timer Pin	**TOUT**	**DOUT**	**Timer Mode**
Input	0	0	Event Counter
	0	1	Input Gated Mode
Output	1	0	Output
	1	1	

Figure 8. Timer Block Diagram

3

PRESCALER

The prescaler is a 7-bit counter used to extend the maximum interval of the overall timer. This counter is clocked by a signal from the TIMER pin or by the internal sync pulse. It divides the frequency received by some factor to create the prescaler output. The factor by which the TIMER pin signal is divided is called the prescaler tap. The value of this tap is selected by three bits of the TSCR (PS0-PS2). These bits control the division of the prescaler input within the range of divide-by-2^0, to divide-by-2^7.

TIMER COUNTER

The timer counter, which may be read or loaded under program control, is decremented from a maximum value of 256 toward zero by the prescaler output. Both are decremented on rising clock edges.

The prescaler register and timer count register are readable and writeable. A write to either one will take precedence over the normal counter function. For example, if a value is written to the timer count register and this write and a decrement-to-zero occur at the same time, the write takes precedence and TSCR bit one (TMZ) is not set until the next timer time out.

TIMER PIN

The TIMER pin may be programmed as either an input or an output. Its status depends on the value of TSCR bits 4 (DOUT) and 5 (TOUT). Two distinct input modes exist; input gated mode and input event counter mode. This relationship is shown in the TIMER pin status section of Figure 8. The frequency of the internal clock applied to the TIMER pin must be less than t_{byte}, which is the frequency of the oscillator divided by either 12, 24, or 48, then multiplied by the clock divide ratio. Whether f_{osc} is divided by 12, 24, or 48 is a manufacturing mask option.

TIMER INPUT EVENT COUNTER MODE

In the timer input event counter mode, both TOUT and DOUT are logic zero. The TIMER pin is effectively connected directly to prescaler input. So, the timer/prescaler is clocked by the signal applied from the TIMER pin.

TIMER INPUT GATED MODE

In the input gated mode, TOUT is logic zero and DOUT is logic one. The timer pin is an input which decrements the prescaler each machine cycle as long as the timer pin is logic high. When the pin is logic low, counting is inhibited. This mode permits the counting of the period of time during which the timer pin is logic high, based on the system clock and prescaler values. Gate times are $f_{osc}/12$, $f_{osc}/24$, and $f_{ocs}/48$.

TIMER OUTPUT MODE

In the output mode, TOUT is logic one and the TIMER pin is connected to the DOUT latch. So, the timer prescaler is clocked by the internal sync pulse. This pulse is a divide-by-12, 24 or 48 of the internal oscillator depending on the mask option. However, in the output mode, once the prescaler decrements the timer count register

to zero, the low TSCR bit 1 (TMZ) bit state is used to drive the data latched at TSCR bit 4 (DOUT) onto the TIMER pin.

NOTE

TMZ is normally set to logic one when TCR decrements to zero and the timer times out. However, it may be set by a write of $00 to TCR or by a write to bit 7 of TSCR.

TIMER COUNT REGISTER ($FE)

The timer count register reflects the current count in the internal 8-bit counter. The register is the counter and can be written.

7							0
MSB							LSB
RESET:							
1	1	1	1	1	1	1	1

TIMER STATUS/CONTROL REGISTER (TSCR) ($09)

7	6	5	4	3	2	1	0
TMZ	ETI	TOUT	DOUT	$\overline{PSI}$	PS2	PS1	PS0
RESET:							
0	0	0	0	0	0	0	0

TMZ — Timer Zero
- 1 = Timer count register has reached the all-zero state since the last time the TMZ bit was read.
- 0 = This bit is cleared by a read of the TSCR if TMZ is read as logic one.

ETI — Enable Timer Interrupt
- 1 = Timer interrupt enabled.
- 0 = Timer interrupt disabled.

TOUT — Timer Output
- 1 = Output mode is selected for the timer.
- 0 = Input modes are selected for the timer.

DOUT — Data Output

In the input mode, latched data at this bit is sent to the TIMER pin when both the TMZ and TOUT bits are logic high.

In the input mode:
- 1 = Timer input gated mode is selected
- 0 = Timer input event counter mode is selected

$\overline{PSI}$ — Prescaler Initialization
- 1 = Prescaler begins to decrement.
- 0 = Prescaler is initialized and counting is inhibited.

PS0-PS2

These bits are used to select the prescaler tap. The coding of the bits is shown below:

PS2	PS1	PS0	Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

It is recommended that MVI or loading and storing instructions be used when changing bit values in the TSCR. Read-modify-write instructions can cause the TMZ to assume an unexpected state.

During reset, the TSCR is set to all zeroes. The TIMER pin is in the high impedance input mode; and DOUT LATCH is forced to a logic high. At the same time, PS0-PS2 coding sets the prescaler tap at divide-by-one, and bit 3 initializes the prescaler.

TIMER PRESCALER REGISTER ($FD)

The timer prescaler register reflects the current count of the 7-bit prescaler. This register is the prescaler counter and can be written.

6						0
MSB						LSB
RESET:						
1	1	1	1	1	1	1

INSTRUCTION SET

The MCU has a set of 42 basic instructions. They can be divided into five different types: register/memory, read/modify/write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is the accumulator; the other is obtained from memory using one of the addressing modes. Refer to the following list of instructions.

Function	Mnemonic
Load A from Memory	LDA
Load XP from Memory	LDX
Load YP from Memory	LDY
Store A in Memory	STA
Add to A	ADD
Subtract from A	SUB
AND Memory to A	AND
Transfer A to XP	TAX
Transfer A to YP	TAY
Transfer YP to A	TYA
Transfer XP to A	TPA
Clear A	CLRA
Clear XP	CLRX
Clear YP	CLRY
Complement A	COMA
Rotate A Left and Carry	ROLA
Arithmetic Compare with Memory	CMP
Move Immediate Value to Memory	MVI
Arithmetic Left Shift of A	ASLA

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. All INC and DEC forms along with all bit manipulation instructions use this method. Refer to the following list of instructions.

Function	Mnemonic
Increment Memory Location	INC
Increment A	INCA
Increment XP	INCX
Increment YP	INCY
Decrement Memory Location	DEC
Decrement A	DECA
Decrement XP	DECX
Decrement YP	DECY

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two byte instructions. Refer to the following list of instructions.

Function	Mnemonic
Branch if Carry Clear	BCC
Branch if Higher or Same	(BHS)
Branch if Carry Set	BCS
Branch if Lower	(BLO)
Branch if Not Equal	BNE
Branch if Equal	BEQ

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the 256 bytes of the memory space, where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear, and bit test and branch functions are all implemented with a single instruction. For the test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list of instructions.

Function	Mnemonic
Branch If Bit n is Set	BRSET n(n=0 . . . 7)
Branch If Bit n is Clear	BRCLR n(n=0 . . . 7)
Set Bit n	BSET n(n=0 . . . 7)
Clear Bit n	BCLR n(n=0 . . . 7)

CONTROL INSTRUCTIONS

These instructions are used to control processor operation during program execution. The jump conditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following list of instructions.

Function	Mnemonic
Return from Subroutine	RTS
Return from Interrupt	RTI
No Operation	NOP
Jump to Subroutine	JSR
Jump Unconditional	JMP
Stop	STOP
Wait	WAIT

IMPLIED INSTRUCTIONS

Since the accumulator and all other registers are located in RAM, many implied instructions exist. Some of the instructions recognized and translated by the assembler are shown below:

Mnemonic	Becomes	Mnemonic	Becomes
ASLA	ADD $FF	INCX	INC $80
BHS	BCC	INCY	INC $81
BLO	BCS	LDXI	MVI $80 DATA
CLRA	SUB $FF	LDYI	MVI $81 DATA
CLRX	MVI $80 #0	NOP	BEQ (PC) +1
CLRY	MVI $81 #0	TAX	STA $80
DECA	DEC $FF	TAY	STA $81
DECX	DEC $80	TXA	LDA $80
DECY	DEC $81	TYA	LDA $81
INCA	INC $FF		

Some examples of valuable instructions not specifically recognized by the assembler are shown below:

Mnemonic	Meaning
BCLR 7,$FF	Ensures A is plus
BSET 7, $FF	Ensures A is minus
BRCLR 7, $FF	Branch if A is plus
BRSET 7, $FF	Branch if A is minus
BRCLR 7, $80	Branch if X is plus (BXPL)
BRSET 7, $80	Branch if X is minus (BXMI)
BRCLR 7, $81	Branch if Y is plus (BYPL)
BRSET 7, $81	Branch if Y is minus (BYMI)

OPCODE MAP

Table 1 is a listing of all the instruction set opcodes applicable to the MC68HC04J3 MCU.

Table 1. Opcode Map

Hi / Low	0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111	8 1000	9 1001	A 1010	B 1011	C 1100	D 1101	E 1110	F 1111	Hi / Low
	Branch Instructions								Register/Memory, Control, and Read/Modify/Write Instructions				Bit Manipulation Instructions		Register/Memory and Read/Modify/Write		
0 0000	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	4 MVI 3 IMM	5 BRCLR0 3 BTB	4 BCLR0 2 BSC	4 LDA 1 R IND	4 LDA 1 R IND	0 0000
1 0001	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	*	5 BRCLR1 3 BTB	4 BCLR1 2 BSC	4 STA 1 R IND	4 STA 1 R IND	1 0001
2 0010	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	2 RTI 1 INH	5 BRCLR2 3 BTB	4 BCLR2 2 BSC	4 ADD 1 R IND	4 ADD 1 R IND	2 0010
3 0011	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	2 RTS 1 INH	5 BRCLR3 3 BTB	4 BCLR3 2 BSC	4 SUB 1 R IND	4 SUB 1 R IND	3 0011
4 0100	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	4 COMA 1 INH	5 BRCLR4 3 BTB	4 BCLR4 2 BSC	4 CMP 1 R IND	4 CMP 1 R IND	4 0100
5 0101	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	4 ROLA 1 INH	5 BRCLR5 3 BTB	4 BCLR5 2 BSC	4 AND 1 R IND	4 AND 1 R IND	5 0101
6 0110	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	2 STOP 1 INH	5 BRCLR6 3 BTB	4 BCLR6 2 BSC	4 INC 1 R IND	4 INC 1 R IND	6 0110
7 0111	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	*	2 WAIT 1 INH	5 BRCLR7 3 BTB	4 BCLR7 2 BSC	4 DEC 1 R IND	4 DEC 1 R IND	7 0111
8 1000	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	4 INC 1 S D	4 DEC 1 S D	5 BRSET0 3 BTB	4 BSET0 2 BSC	4 LDA 2 IMM	4 LDA 2 DIR	8 1000
9 1001	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	4 INC 1 S D	4 DEC 1 S D	5 BRSET1 3 BTB	4 BSET1 2 BSC	#	4 STA 2 DIR	9 1001
A 1010	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	4 INC 1 S D	4 DEC 1 S D	5 BRSET2 3 BTB	4 BSET2 2 BSC	4 ADD 2 IMM	4 ADD 2 DIR	A 1010
B 1011	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	4 INC 1 S D	4 DEC 1 S D	5 BRSET3 3 BTB	4 BSET3 2 BSC	SUB IMM	4 SUB 2 DIR	B 1011
C 1100	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	4 LDA 1 S D	4 STA 1 S D	5 BRSET4 3 BTB	4 BSET4 2 BSC	4 CMP 2 IMM	CMP DIR	C 1100
D 1101	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	4 LDA 1 S D	4 STA 1 S D	5 BRSET5 3 BTB	4 BSET5 2 BSC	4 AND 2 IMM	4 AND 2 DIR	D 1101
E 1110	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JRSn 2 EXT	4 JMPn 2 EXT	4 LDA 1 S D	4 STA 1 S D	5 BRSET6 3 BTB	4 BSET6 2 BSC	#	4 INC 2 DIR	E 1110
F 1111	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL	4 JSRn 2 EXT	4 JMPn 2 EXT	4 LDA 1 S D	4 STA 1 S D	5 BRSET7 3 BTB	4 BSET7 2 BSC	#	4 DEC 2 DIR	F 1111

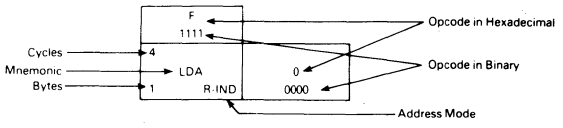
Abbreviations for Address Modes

INH	Inherent
S-D	Short Direct
B-T-B	Bit Test and Branch
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
R-IND	Register Indirect

* Indicates Instruction Reserved for Future Use

\# Indicates Illegal Instruction

LEGEND



ADDRESSING MODES

The MCU has nine different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. It deals with objects in three different address spaces: program space, data space, and stack space. The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is located in program ROM. It is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution, such as a constant used to initialize a loop counter.

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the 256 bytes in memory with a single two-byte instruction.

SHORT DIRECT

In the short direct addressing mode, the MCU has four locations in data space RAM which it can use, ($80, $81, $82, and $83). The opcode determines the data space RAM location, and the instruction is only one byte. Short direct addressing is a subset of the direct addressing mode. The X and Y registers are at locations $80 and $81, respectively.

EXTENDED

In the extended addressing mode, the effective address of the argument is obtained by concatenating the four least-significant bits of the opcode with the byte following the opcode to form a 12-bit address. Instructions using the extended addressing mode, such as JMP or JSR, are capable of branching anywhere in program space. An extended addressing mode instruction is two bytes long.

RELATIVE

The relative addressing mode is only used in conditional branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −15 to +16 from the opcode address. The programmer need not calculate the offset when using the Motorola assembler, since it calculates the proper offset and checks to see that it is within the span of the branch.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte in which the specified bit is to be set or cleared. Thus, any bit in the 256 locations of data space memory that can be written to can be set or cleared with a single two-byte instruction.

CAUTION

The corresponding DDRs for ports A and B are write only registers (registers at $04, $05). A read operation on these registers is undefined. Since BSET and BCLR are read-modify-write functions, they cannot be used to set or clear a DDR bit; all "unaffected" bits would be set. Write all DDR bits in a port using a single-store instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit that is to be tested and its condition (set or clear) is included in the opcode. The data space address of the byte to be tested is in the single byte immediately following the opcode byte. The third byte is sign extended to twelve bits and becomes the offset added to the PC if the condition is true. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry flag.

REGISTER-INDIRECT

In the register-indirect addressing mode, the operand is at the address in data space pointed to by the contents of one of the indirect registers, X or Y. The particular indirect register is selected by bit 4 of the opcode. Bit 4 decodes into an address that represents the register, $80 or $81. A register-indirect instruction is one byte long.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. These instructions are one byte long.

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{DD}	−0.3 to +7.0	V
Input Voltage	V_{in}	$V_{SS}-0.3$ to $V_{DD}+0.3$	V
Current Drain per Pin Excluding V_{DD} and V_{SS}	I	10	mA
Total Current for Sink Ports A, B, C EXTAL, TIM Source	I I	30 15	mA
Operating Temperature Range (Comm.)	T_A	0 to 70	°C
Operating Temperature Range (Ind.)	T_A	−40 to +85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature Plastic	T_J	 150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields. However, it is advised that normal precautions be taken to avoid applications of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}) \leq V_{DD}$. Reliability of operation is enhanced if unused inputs except EXTAL are connected to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

3

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic	θ_{JA}	 70	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

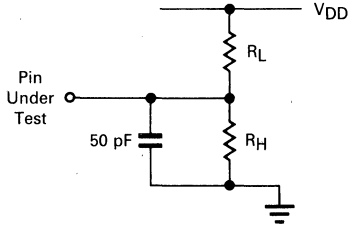
An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.



V_{DD} = +4.5V
I_{OL}/I_{OH} = 800 µA
$R_L = R_H$ = 4.6 kΩ

V_{DD} = +2.7V
I_{OL}/I_{OH} = 200 µA
$R_L = R_H$ = 10.5 kΩ

V_{DD} = +2.0V
I_{OL}/I_{OH} = 100 µA
$R_L = R_H$ = 16 kΩ

Figure 9. Equivalent Test Load

CONTROL TIMING CHARACTERISTICS

Characteristic	Symbol	Min	Typ	Max	Unit
(V_{DD} = +5 Vdc ±10%, V_{SS} = O Vdc; T_A = 0°C to 70°C					
Oscillator Frequency	f_{osc}	0	—	11.0	MHz
PHI1 Clock Frequency	f_{CL}	0	—	5.5	MHz
Cycle Time (Min)	t_{cyc}	2.2	—	—	µs
$\overline{IRQ}$ Pulse Width	t_{IWL}	2×t_{cyc}	—	—	µs
$\overline{RESET}$ Pulse Width	RWL	2×t_{cyc}	—	—	µs
Oscillator Clock Pulse Width	t_{OL}, t_{OH}	45	—	—	ns
V_{DD} = +3 Vdc ±10%, V_{SS} = 0 Vdc, T_A = 0°C to 70°C					
Oscillator Frequency	f_{osc}	—	—	11	MHz
PHI1 Clock Frequency	f_{CL}	—	—	4.2	MHz
Cycle Time (Min)	t_{cyc}	2.9	—	—	µs
$\overline{IRQ}$ Pulse Width	t_{IWL}	2×t_{cyc}	—	—	µs
$\overline{RESET}$ Pulse Width	t_{RWL}	2×t_{cyc}	—	—	µs
Oscillator Clock Pulse Width	t_{OL},t_{OH}	45	—	—	ns
V_{DD} = +2.2 Vdc ±10%, V_{SS} = 0 Vdc, T_A = 0°C to 70°C					
Oscillator Frequency	f_{osc}	0	—	8.4	MHz
PHI1 Clock Frequency	f_{CL}	0	—	2.1	MHz
Cycle Time (Min)	t_{cyc}	5.7	—	—	µs
$\overline{IRQ}$ Pulse Width	t_{IWL}	2×t_{cyc}	—	—	µs
$\overline{RESET}$ Pulse Width	t_{RWL}	2×t_{cyc}	—	—	µs
Oscillator Clock Pulse Width	t_{OL},t_{OH}	45	—	—	ns

NOTE: 2 V operation is a user-selectable option only. Prior consultation with the factory is required.

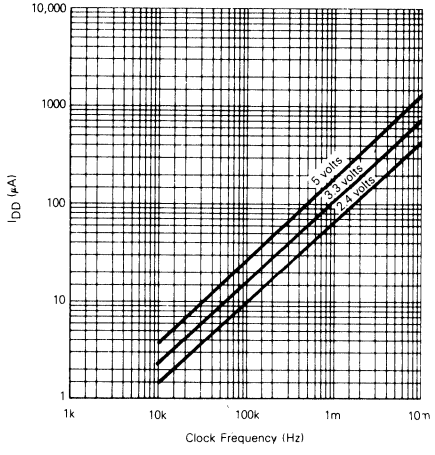


Figure 10. Typical RUN Current vs Clock Frequency

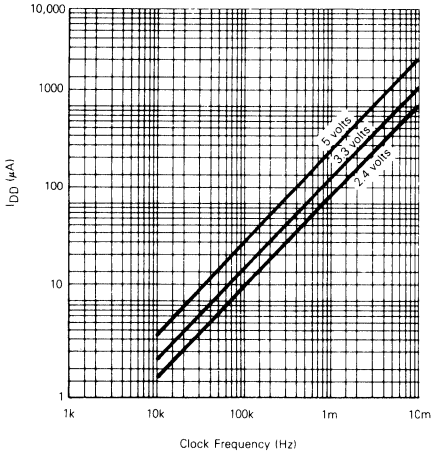


Figure 11. Typical WAIT Current vs Clock Frequency

DC ELECTRICAL CHARACTERISTICS (Typical pull-down sink current for $V_{out} = V_{DD}$ is 50 μA.)

Characteristic		Symbol	Min	Typ	Max	Unit
V_{DD} = +5 Vdc ±10%, V_{SS} = 0 Vdc, T_A = 0°C to 70° C						
Output Voltage, I_{Load} ⟨10.0 μA		V_{OL} V_{OH}	— $V_{DD}-0.1$	— —	0.1 —	V
Output High Voltage, I_{Load} = +800 μA) Output Low Voltage, I_{Load} = +800 μA)	Ports, TIM Ports, TIM	V_{OH} V_{OL}	$V_{DD}-0.4$ —	— —	— 0.4	V
Input High Voltage	Ports, TIM, XTAL, MDS IRQ, RESET	V_{IH} V_{IH}	$0.7 \times V_{DD}$ $0.8 \times V_{DD}$	— —	V_{DD} V_{DD}	V
Input Low Voltage	Ports, TIM, XTAL, MDS IRQ, RESET	V_{IL} V_{IL}	V_{SS} V_{SS}	— —	$0.3 \times V_{DD}$ $0.2 \times V_{DD}$	V
Total Supply Current C_L = 50 pF, Ports, TIM, No dc load, $t_{cyc} = 1/f_{CL}$ (max), V_{IL} = 0.2 V, $V_{IH} = V_{DD} - 0.2$ V	RUN WAIT* STOP*	I_{DD} I_{DD} I_{DD}	— — —	2 0.5 3	3 1 5	mA mA μA
I/O Ports Input Leakage $V_{SS} \langle V_I \langle V_{DD}$		I_{IL}	—	—	±1	μA
Input Current	RESET, IRQ, TIM	I_{in}	—	—	±1	μA
Capacitance per Pin	PORTS (as Input or Output) RESET, IRQ, TIM, XTAL, MDS	C_{out} C_{in}	— —	— —	12 8	pF
V_{DD} = +3 Vdc ±10%, V_{SS} = 0 Vdc, T_A = 0°C to 70°C						
Output Voltage, I_{Load} ⟨10.0 μA		V_{OL} V_{OH}	— $V_{DD}-0.1$	— —	0.1 —	V
Output High Voltage, I_{Load} = −200 μA) Output Low Voltage, I_{Load} = +200 μA)	Ports, TIM Ports, TIM	V_{OH} V_{OL}	$V_{DD}-0.3$ —	— —	— 0.3	V
Input High Voltage	Ports, TIM, XTAL, MDS IRQ, RESET	V_{IH} V_{IH}	$0.7 \times V_{DD}$ $0.8 \times V_{DD}$	— —	V_{DD} V_{DD}	V
Input Low Voltage	Ports, TIM, MDS, XTAL IRQ, RESET	V_{IL} V_{IL}	V_{SS} V_{SS}	— —	$0.3 \times V_{DD}$ $0.2 \times V_{DD}$	V
Total Supply Current C_L = 50 pF, Ports, TIM, No dc load, $t_{cyc} = 1/f_{CL}$(Max), V_{IL} = 0.2 V, $V_{IH} = V_{DD} - 0.2$ V	RUN WAIT* STOP*	I_{DD} I_{DD} I_{DD}	— — —	0.8 0.3 1.5	1.5 0.5 4	mA mA μA
I/O Ports Input Leakage $V_{SS} \langle V_I \langle V_{DD}$		I_{IL}	—	—	±1	μA
Input Current	RESET, IRQ, TIM	I_{in}	—	—	±1	μA
Capacitance per Pin	PORTS (as Input or Output) RESET, IRQ, TIM, XTAL, MDS	C_{out} C_{in}	— —	— —	12 8	pF
V_{DD} = +2.2 Vdc ±10%, V_{SS} = 0 Vdc, T_A = 0°C to 70°C						
Output Voltage, I_{Load} ⟨10.0 μA		V_{OL} V_{OH}	— $V_{DD}-0.1$	— —	0.1 —	V
Output High Voltage, I_{Load} = −100μA) Output Low Voltage, I_{Load} = +100 μA)	Ports, TIM Ports, TIM	V_{OH} V_{OL}	$V_{DD}-0.3$ —	— —	— 0.3	V
Input High Voltage	Ports, TIM, XTAL, MDS IRQ, RESET	V_{IH} V_{IH}	$0.7 \times V_{DD}$ $0.8 \times V_{DD}$	— —	V_{DD} V_{DD}	V
Input Low Voltage	Ports, TIM, MDS, XTAL IRQ, RESET	V_{IL} V_{IL}	V_{SS} V_{SS}	— —	$0.3 \times V_{DD}$ $0.2 \times V_{DD}$	V
Total Supply Current C_L = 50 pF, Ports, TIM, No dc load, $t_{cyc} = 1/f_{CL}$(Max), V_{IL} = 0.2v, $V_{IH} = V_{DD} - 0.2$ V,	RUN WAIT* STOP*	I_{DD} I_{DD} I_{DD}	— — —	0.6 0.2 1	1 0.3 3	mA mA μA
I/O Ports Input Leakage $V_{SS} \langle V_I \langle V_{DD}$		I_{IL}	—	—	±1	μA
Input Current	REST, IRQ, TIM	I_{in}	—	—	±1	μA
Capacitance per Pin	PORTS (as Input or Output) RESET, IRQ, TIM, XTAL, MDS	C_{out} C_{in}	— —	— —	12 8	pF

*Measured under the following conditions:
- All ports and timer pin are configured as input
- XTAL is driven by a square wave input
- EXTAL is open circuit
- port pull downs not enabled

NOTE: Typical pull-down sink current for $V_{out} = V_{DD}$ is 50 μA.

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola using the following media:

MDOS, disk file
MS-DOS disk file (360K)
EPROM(s) 2516, 2716, 2532, 2732

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, a sales person, or a Motorola representative.

FLEXIBLE DISKS

Several types of flexible disks (MDOS™ or MS™-DOS disk file), programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. The diskette should be clearly labeled with the customer's name, date, project or product name, and the filename containing the pattern.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to expedite the process in case of any difficulty with the pattern file.

MDOS Disk File

MDOS is Motorola's Disk Operating System available on the EXORciser® development system. The disk media submitted must be a single-sided, single-density, 8-inch MDOS compatible floppy diskette. The diskette must contain the minimum set of MDOS system files in addition to the pattern file.

The .LO output of the M6804 cross assembler should be furnished. In addition, the file must be produced (using the ROLLOUT command) containing the absolute image of the M6804 memory. It is necessary to include the entire memory image of both program and data space. All unused bytes, including those in the user space, must be set to logic zero.

MS-DOS/PC-DOS Disk File

MS-DOS is Microsoft's Disk Operating System. Disk media submitted must be standard density (360K), double-sided 5 1/4 inch compatible floppy diskette. The diskette must contain the object file code in Motorola's S–record format. The S–record format is a character-based object file format generated by M6804 cross assemblers and linkers on IBM® PC style machines.

EPROMS

Four K of EPROM are necessary to contain the entire MC68HC04J3 program. Two 2516 or 2716 type EPROMs or a single 2532 or 2732 type EPROM can be submitted for pattern generation. The EPROM is programmed with the customer program using positive logic sense for address and data. Submissions on two EPROMs must be clearly marked. All unused bytes, including the user's space, must be set to zero.

If the MC68HC04J3 MCU ROM pattern is submitted on one 2532 or 2732 EPROM, or on two 2516 or 2716 type EPROMs, memory map addressing is one-for-one. The data space ROM runs from EPROM address $018 to $05F and program space ROM runs from EPROM address $970 to $FF7, with vectors from $FFC to $FFF.

For shipment to Motorola, EPROMs should be placed in a conductive IC carrier and packed securely. Styrofoam is not acceptable for shipment.

Verification Media

All original pattern media, EPROMs or floppy disks, are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked, and the verification form should be completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for the creation of the customer mask. To aid in the verification process, Motorola will program *customer supplied* blank EPROM(s) or DOS disks from the data file used to create the custom mask.

ROM Verification Units (RVUs)

Ten MCUs containing the customer's ROM pattern will be sent for program verification. These units will have been made using the custom mask but are for the purpose of ROM verification only. For expediency, the MCUs are unmarked, packaged in ceramic, and tested with five volts at room temperature. These RVUs are free with the minimum order quantity but are not production parts. These RVUs are not guaranteed by Motorola Quality Assurance.

Ordering Information

The following table provides generic information pertaining to the package type, temperature, and order numbers for the MC68HC04J3.

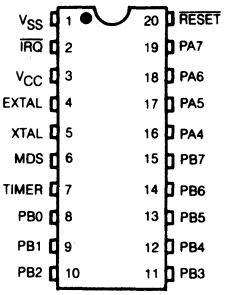
Ordering Information

Package Type	Temperature	Order Number
Plastic (P Suffix)	0°C to 70°C	MC68HC04J3P
	−40°C to +85°C	MC68HC04J3CP

MDOS is a trademark of Motorola Inc.
MS-DOS is a trademark of Microsoft, Inc.
EXORciser is a registered trademark of Motorola Inc.
IBM is a registered trademark of Internal Business Machines Corporation.

MECHANICAL DATA

PIN ASSIGNMENTS



3

MC68HC04P4

Product Preview

8-Bit HCMOS Microcontroller Unit

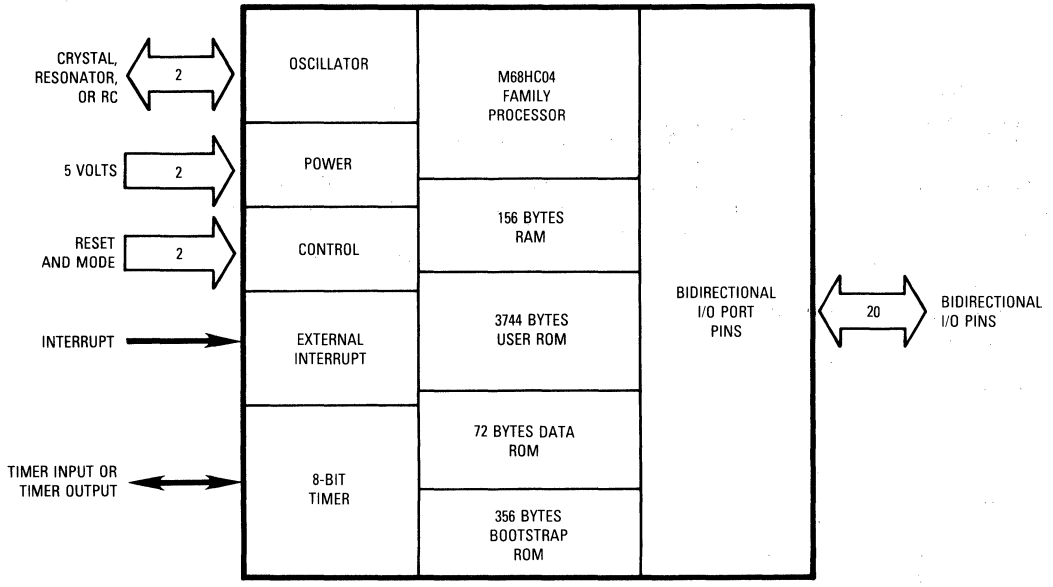
The MC68HC04P4 Microcontroller Unit (MCU) device is a member of the M68HC04 Family of low-cost, low-power, single-chip microcontrollers. It is designed for the user who needs an economical MCU with the proven capabilities of the M6805-based instruction set.

The following are some of the hardware and software features of the MC68HC04P4.

- HCMOS Technology
- Power Saving STOP and WAIT Modes
- 156 Bytes of RAM
- 3744 Bytes User ROM (including $\overline{\text{RESET}}$ and $\overline{\text{IRQ}}$ vectors)
- 72 Bytes of Data ROM
- 352 Bytes of Selfcheck ROM
- 20 Bidirectional Memory Mapped I/O
- Versatile Indirect X and Y Registers
- On-Chip Clock Generator
- Master and Power-Up Reset
- External and Timer Interrupts
- Single 2-6 Volt Supply
- Byte Efficient Instruction Set
- True Bit Manipulation
- Bit Test and Branch Instruction
- Conditional Branch
- Single Instruction Memory Examine/Change
- Timer Pin is Software Programmable as Clock Input or Timer Output
- 9 Powerful Addressing Modes
- STOP and WAIT Instructions
- Crystal/Ceramic Resonator
- User's Selectable Options
  - — Pull-ups on $\overline{\text{IRQ}}$ and $\overline{\text{RESET}}$
  - — Crystal or Ceramic Resonator Oscillator or Special Resistor and Capacitor
  - — Interrupt, Edge Sensitive, or Special Level-Sensitive, or Edge and Level Sensitive
- 28-Pin DIP and PLCC Package

3

This document contains information on a new product. Specifications and information herein are subject to change without notice.


CRYSTAL, RESONATOR, OR RC
2
5 VOLTS
2
RESET AND MODE
2
INTERRUPT
TIMER INPUT OR TIMER OUTPUT
OSCILLATOR
POWER
CONTROL
EXTERNAL INTERRUPT
8-BIT TIMER
M68HC04 FAMILY PROCESSOR
156 BYTES RAM
3744 BYTES USER ROM
72 BYTES DATA ROM
356 BYTES BOOTSTRAP ROM
BIDIRECTIONAL I/O PORT PINS
20
BIDIRECTIONAL I/O PINS

3

MC68HC704P4

Product Preview

One Time Programmable ROM (OTPROM) or Standard Eraseable Programmable Read-Only Memory (EPROM) Microcomputer

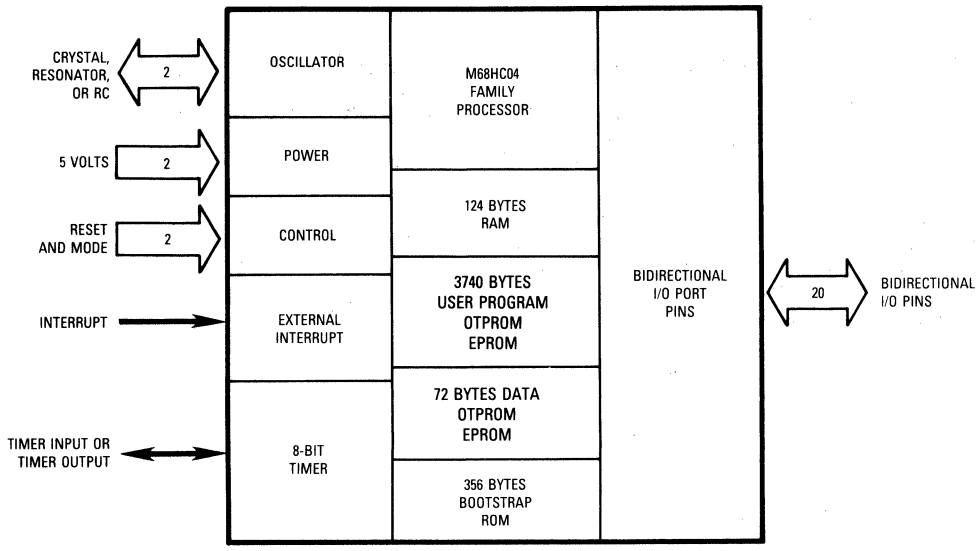
The MC68HC704P4 Microcomputer Unit (MCU) device is similar to the MC68HC04P4 with the following exceptions. The exceptions incorporate 3740 bytes of One Time Programmable Read-Only Memory (OTPROM) or Standard Eraseable Programmable Read-Only Memory (EPROM) and 124 bytes of RAM (refer to the block diagram).

The following are some of the hardware and software features of the MC68HC704P4.

- HCMOS Technology
- Power Saving STOP and WAIT Modes
- 124 Bytes of RAM
- 3740 Bytes of OTPROM or EPROM (including RESET and IRQ vectors)
- 72 Bytes of Data PROM/EPROM
- 352 Bytes of Selfcheck ROM
- 20 Bidirectional Memory Mapped I/O
- Versatile Indirect X and Y Registers
- On-Chip Clock Generator
- Master Reset
- External and Timer Interrupts
- Single 2-6 Volt Supply
- Byte Efficient Instruction Set
- True Bit Manipulation
- Bit Test and Branch Instruction
- Conditional Branch
- Single Instruction Memory Examine/Change
- Timer Pin is Software Programmable as Clock Input or Timer Output
- 9 Powerful Addressing Modes
- STOP and WAIT Instructions
- Crystal/Ceramic Resonator
- OTPROM/EPROM Option Register User's Options
  - –Edge or Level IRQ
  - –Oscillator Divide Ratio (by 1, 2, or 4)
  - –OTPROM/EPROM Content Protection
- OTPROM/EPROM Self-Programming Mode
- OTPROM/EPROM Contents Verification by On-Chip Signature Analysis
- 28-Pin DIP (OTPROM) Package
- 28-Pin DIP (EPROM Window) Package

This document contains information on a new product. Specifications and information herein are subject to change without notice.

BLOCK DIAGRAM



3

MC6805P2

Technical Summary
8-Bit Microcomputer Unit

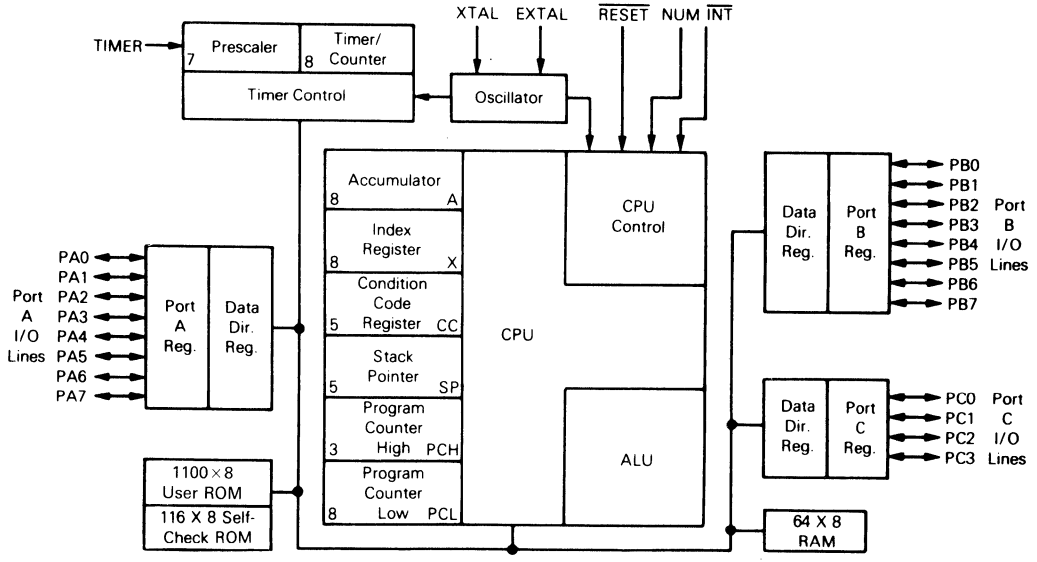
The MC6805P2 (HMOS) Microcomputer Unit (MCU) is a member of the MC6805 Family of microcomputers. This low cost and high speed MCU has parallel I/O capability with pins programmable as input or output. This publication contains condensed information on the MCU; for detailed information, refer to *M6805 HMOS, M146805 CMOS Family User's Manual* (M6805UM(AD2)) or contact your local Motorola sales office.

Refer to the block diagram for the hardware features and to the below list for additional features available on the MCU.

- Internal 8-Bit Timer with 7-Bit Programmable Prescaler
- On-chip Oscillator
- Memory Mapped I/O
- Versatile Interrupt Handling
- True Bit Manipulation
- Bit Test and Branch Instruction
- 20 I/O Ports
- Vectored Interrupts
- 64 Bytes RAM
- Low Voltage Inhibit Option
- Self-Check Mode
- Master Reset
- 1100 Bytes ROM

3

BLOCK DIAGRAM



SIGNAL DESCRIPTION

V_{CC} AND V_{SS}

Power is supplied to the microcomputer using these two pins. V_{CC} is +5.25 volts (±0.5Δ) power, and V_{SS} is ground.

$\overline{INT}$

This pin provides the capability for asynchronously applying an external interrupt to the MCU. Refer to INTERRUPTS for additional information.

NUM (Non-User Mode)

This pin is not for user applications and must be connected to V_{SS}.

EXTAL, XTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a ceramic resonator, a resistor/capacitor combination, or an external signal (depending upon selected manufacturing mask options) is connected to these pins to provide a system clock.

RC Oscillator

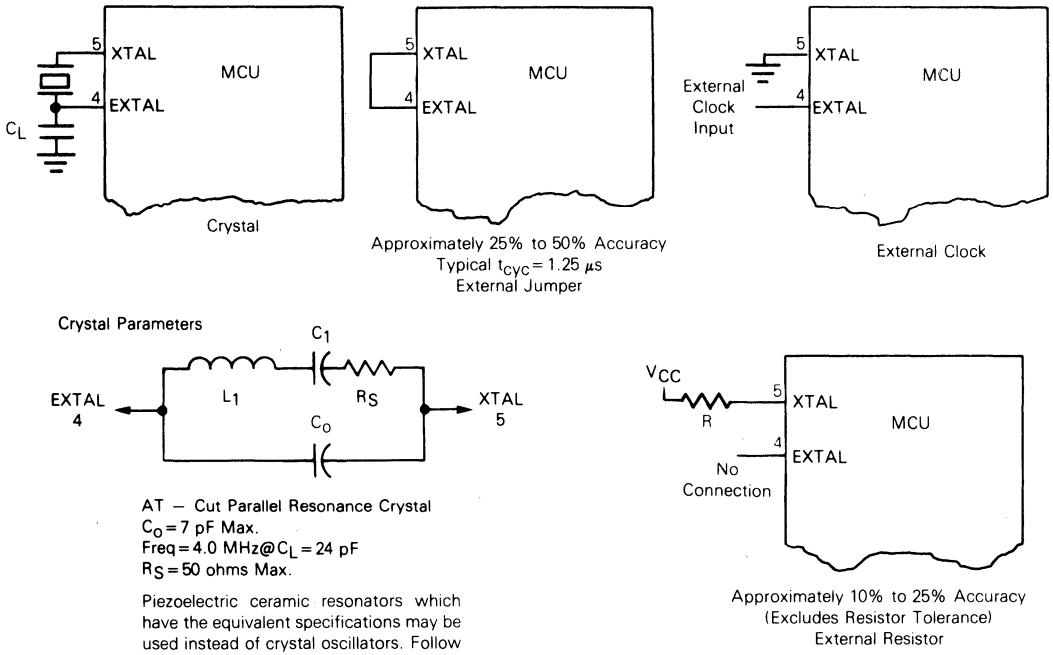
With this option, a resistor is connected to the oscillator pins as shown in Figure 1. The relationship between R and f_{osc} is shown in Figure 2.

Crystal

The circuit shown in Figure 1 is recommended when using a crystal. Using an external CMOS oscillator is recommended when crystals outside the specified ranges are to be used. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time. Refer to **ELECTRICAL SPECIFICATIONS** for V_{CC} specifications.

External Clock

An external clock should be applied to the EXTAL input with the XTAL input connected to ground, as shown in Figure 1.



NOTE: The recommended C_L value with a 4.0 MHz crystal is 27 pF maximum including system distributed capacitance. There is an internal capacitance of approximately 25 pF on the XTAL pin. For crystal frequencies other than 4 MHz, the total capacitance on each pin should be scaled as the inverse of the frequence ratio. For example, with a 2 MHz crystal, use approximately 50 pF on EXTAL and approximately 25 pF on XTAL. The exact value depends on the motional-arm parameters of the crystal used.

Figure 1. Oscillator Connections

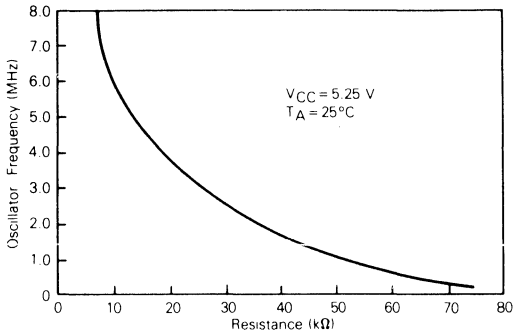


Figure 2. Typical Frequency vs Resistance for RC Oscillator Option only

TIMER

This pin can be used as an external input to control the internal timer/counter circuitry or gating $\phi 2$ input to timer, depending on mask option.

$\overline{\text{RESET}}$

This pin has a Schmitt trigger input and an on-chip pullup. The MCU can be reset by pulling $\overline{\text{RESET}}$ low.

INPUT/OUTPUT LINES (PA0-PA7, PB0-PB7, PC0-PC3)

These 20 lines are arranged into two 8-bit ports (A and B) and one 4-bit port (C). Ports A, B, and C are programmable as either inputs or outputs under software control of the data direction registers. Refer to **PROGRAMMING** for additional information.

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

Any port pin is programmable as either input or output under software control of the corresponding data direction register (DDR). The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic 1 for output and a logic 0 for input. On reset, all the DDRs are initialized to a logic 0 state to put the ports in the input mode. The port output registers are not initialized on reset and should be written to before setting the DDR bits.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This may be used to initialize the data registers and avoid undefined outputs. Care must be exercised when using read-modify-write instructions since the data read corresponds to the pin level if the DDR is an input (0) and also to the latched output when the DDR is an output (1). Refer to Table 1 for I/O functions and to Figure 3 for typical port circuitry.

Table 1. I/O Pin Functions

Data Direction Register Bit	Output Data Bit	Output State	Input To MCU
1	0	0	0
1	1	1	1
0	X	Hi-Z	Pin

MEMORY

The MCU is capable of addressing 2048 bytes of memory and I/O registers. The memory map is shown in Figure 4. The locations consist of 1092 bytes of user ROM, self-check ROM, user RAM, a timer control register, and I/O. The user interrupt vectors are located from $7F8 to $7FF.

The stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer decrements during pushes and increments during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

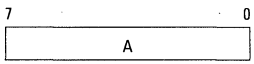
Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

REGISTERS

The MCU contains the registers described in the following paragraphs.

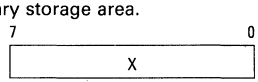
ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



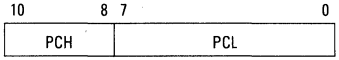
INDEX REGISTER (X)

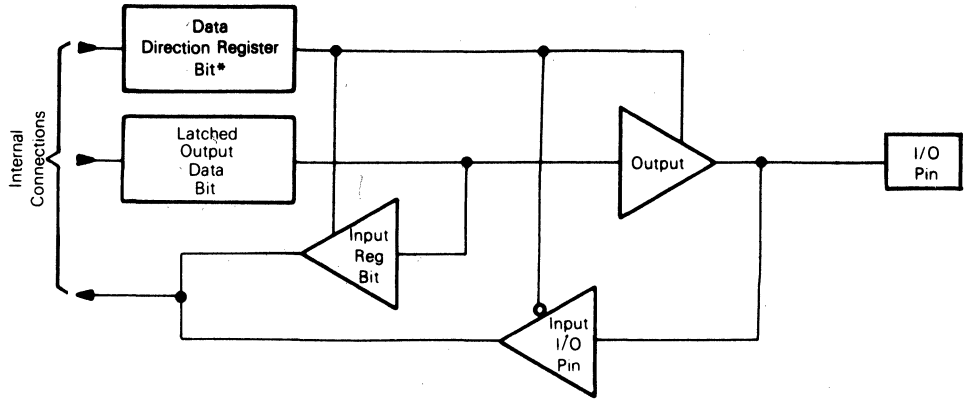
The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16-bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



PROGRAM COUNTER (PC)

The program counter is an 11-bit register that contains the address of the next byte to be fetched.





*DDR is a write-only register and reads as all "1s".

PORT DATA REGISTER

7 0

Port A Addr = $000
Port B Addr = $001
Port C Addr = $002 (Bits 0→3)

PORT DATA DIRECTION REGISTER (DDR)

7 0

(1) Write Only; reads as all "1s"
(2) 1 = Output; 0 = Input. Cleared to 0 by reset.
(3) Port A Addr = $004
Port B Addr = $005
Port C Addr = $006 (Bits 0→3)

Figure 3. Typical Port I/O Circuitry and Register Configuration

STACK POINTER (SP)

The stack pointer is an 11-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set at location $07F. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

The six most-significant bits of the stack pointer are permanently set at 000011. Subroutines and interrupts may be nested down to location $061 (31 bytes maximum), which allows the programmer to use up to 15 levels of subroutine calls (less if interrupts are allowed).

10					5	4	0
0	0	0	0	1	1	SP	

CONDITION CODE REGISTER (CC)

The condition code register is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program, and specifications can be taken as a result of their state. Each bit is explained in the following paragraphs.

4				0
H	I	N	Z	C

Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, the timer and external interrupt is masked (disabled). If an external interrupt occurs while this bit is set, the interrupt is latched and is processed as soon as the interrupt bit is cleared.

Negative (N)

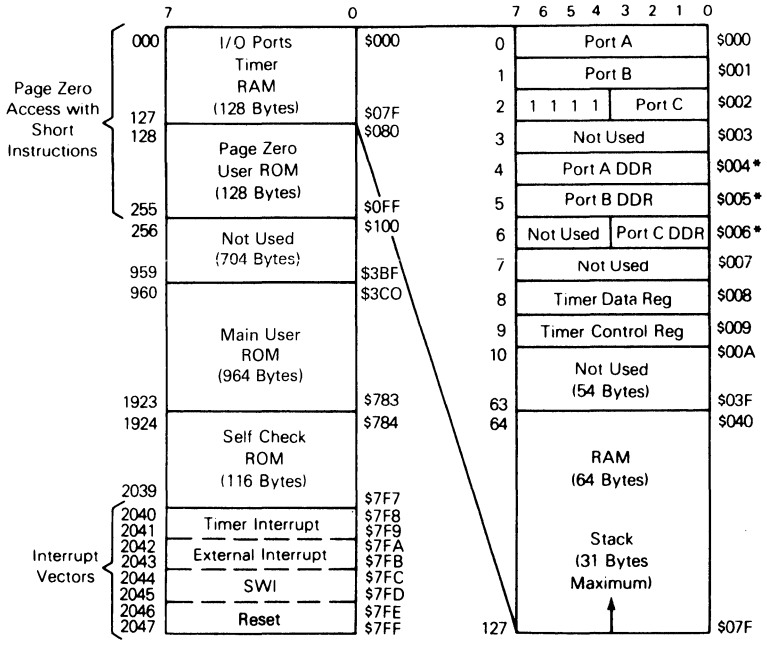
When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic 1).

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/Borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions, and during shifts and rotates.



*Caution: Data direction registers (DDRs) are write-only, set to $FF.

Figure 4. Memory Map

SELF CHECK

The self check is initiated by connecting the MCU as shown in Figure 5 and then monitoring the output of port C (bit 3) for an oscillation of approximately 7 Hz. The following tests are executed automatically:

I/O — functionally exercise I/O ports
RAM — walking bit test
ROM — exclusive OR with ODD "1s" parity result
Timer — functionally exercise timer
Interrupts — functionally exercise external and timer interrupts

Table 2 shows the status of the LEDs as a result of a failure. Port C is tested only once (just after reset). If port C fails, no lights will appear.

Table 2. Self-Check Error Patterns

PC1	PC0	Problem
0	0	Interrupt Failure
0	1	Bad Port A or Port B
1	0	Bad RAM
1	1	Bad RAM
All 4 LEDs Flashing		Good Device

RESETS

The MCU can be reset three ways: (1) by initial power-up, (2) by the external reset input ($\overline{\text{RESET}}$), and (3) by an optional, internal, low-voltage detect circuit. The $\overline{\text{RESET}}$ input consists mainly of a Schmitt trigger that senses the $\overline{\text{RESET}}$ line logic level.

POWER-ON-RESET (POR)

An internal reset is generated on power-up that allows the internal clock generator to stabilize. The power-on reset is used strictly for power turn-on conditions and should not be used to detect any drop in the power supply voltage. A delay of t_{RHL} milliseconds is required before allowing the $\overline{\text{RESET}}$ input to go high. Connecting a capacitor to the $\overline{\text{RESET}}$ input (Figure 6) typically provides sufficient delay.

EXTERNAL RESET INPUT

The MCU is reset when a logic zero is applied to the $\overline{\text{RESET}}$ input for a period longer than one machine cycle (t_{cyc}). Under this type of reset, the Schmitt trigger switches off at V_{IRES-} to provide an internal reset voltage.

LOW-VOLTAGE INHIBIT (LVI)

The optional low-voltage detection circuit causes a reset of the MCU if the power supply voltage falls below a certain level (V_{LVI}). The only requirement being that V_{CC}

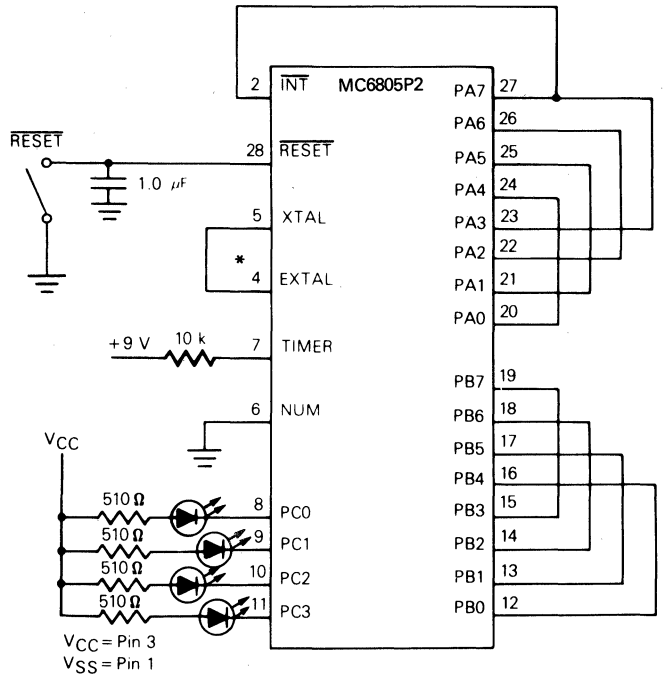


Figure 5. Self-Check Connections

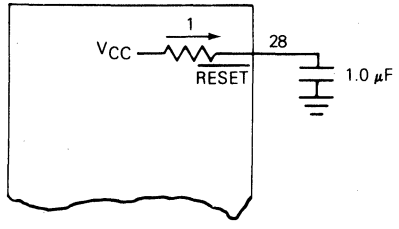


Figure 6. Power-Up RESET Delay Circuit

must remain at or below the V_{LVI} threshold for one t_{cyc} minimum.

In typical applications, the V_{CC} bus filter capacitor will eliminate negative-going voltage glitches of less than one t_{cyc}. The ouput from the low-voltage detector is connected directly to the internal reset circuitry. It also forces the $\overline{RESET}$ pin low via a strong discharge device through a resistor. The internal reset is removed once the power supply voltage rises above a recovery level (V_{LVR}), at which time a normal power-on reset occurs.

INTERRUPTS

The MCU can be interrupted three different ways: (1) through the external interrupt $\overline{INT}$ input pin, (2) with the internal timer interrupt request, or (3) using the software interrupt instruction (SWI).

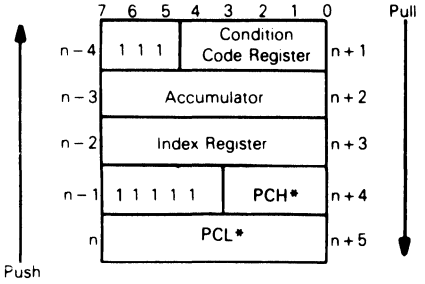
Interrupts cause the processor registers to be saved on the stack and the interrupt mask (I bit) set to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack and then normal processing resumes. The stacking order is shown in Figure 7.

Unlike $\overline{RESET}$, hardware interrupts do not cause the current instruction execution to be halted but are considered pending until the current instruction is complete.

NOTE

The current instruction is considered to be the one already fetched and being operated on.

When the current instruction is complete, the processor checks all pending hardware interrupts and if unmasked (I bit clear) proceeds with interrupt processing; otherwise, the next instruction is fetched and executed. Masked interrupts are latched for later interrupt service. If the



*For subroutine calls, only PCH and PCL are stacked.

Figure 7. Interrupt Stacking Order

timer interrupt status bit is cleared before unmasking the interrupt, then the interrupt is not latched.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction regardless of the setting of the I bit. Refer to Figure 8 for the reset and interrupt instruction processing sequence.

TIMER INTERRUPT

If the timer mask bit (TCR6) is cleared, then each time the timer decrements to zero (transitions from $01 to $00) an interrupt request is generated. The actual processor interrupt is generated only if the interrupt mask bit of the condition code register (CCR) is also cleared. When the

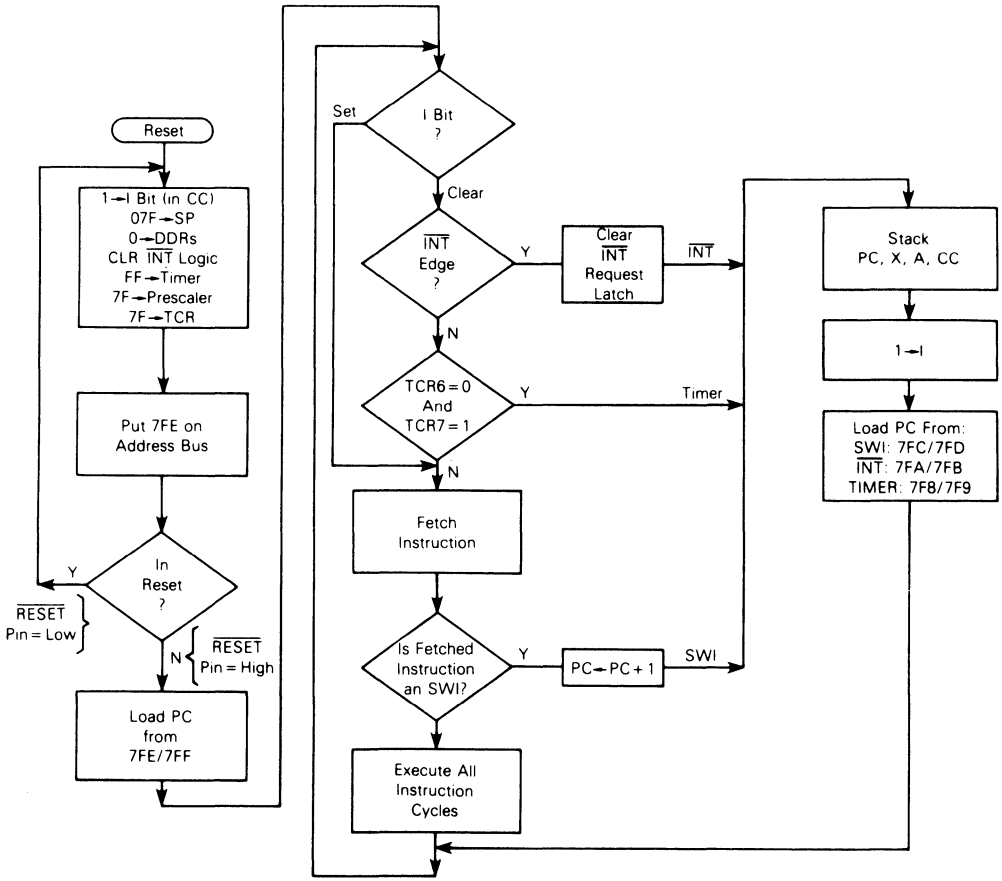


Figure 8. Reset and Interrupt Processing Flowchart

interrupt is recognized, the current state of the machine is pushed onto the stack and the I bit in the CCR is set, masking further interrupts until the present one is serviced. The contents of the timer interrupt vector, containing the location of the timer interrupt service routine, is then loaded into the program counter. At the end of the timer interrupt service routine, the software normally executes an RTI instruction which restores the machine state and starts executing the interrupted program.

EXTERNAL INTERRUPT

The external interrupt is internally synchronized and then latched on the falling edge of $\overline{INT}$. Clearing the I bit enables the external interrupt. The MC6805P2 only requires negative edge-sensitive trigger interrupts. The following paragraphs describe two typical external interrupt circuits.

Zero-Crossing

A sinusoidal input signal (f_{INT} maximum) can be used to generate an external interrupt (see Figure 9a) for use as a zero-crossing detector (for negative transitions of the ac sinusoid). This type of circuit allows applications such as servicing time-of-day routines and engaging/disengaging ac power control devices. Off-chip, full-wave rectification provides an interrupt at every zero crossing of the ac signal and thereby provides a 2f clock.

Digital-Signal Interrupt

With this type of circuit (Figure 9b), the $\overline{INT}$ pin can be driven by a digital signal. The maximum frequency of a signal that can be recognized by the TIMER or $\overline{INT}$ pin logic is dependent on the parameter labeled t_{WL}, t_{WH}. Refer to **TIMER** for additional information.

SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. If the I bit is zero, SWI executes after the other interrupts. The SWI execution is similar to the hardware interrupts.

TIMER

The MCU consists of an 8-bit software programmable counter driven by a 7-bit software programmable prescaler. The timer source is made during manufacturing as a mask option. The 8-bit counter may be loaded under program control and is decremented toward zero. When the timer reaches zero, the timer interrupt request bit (bit 7) in the timer control register (TCR) is set. Refer to Figure 10 for timer block diagram.

The timer interrupt can be masked (disabled) by setting the timer interrupt mask bit (bit 6) in the TCR. When the I bit in the condition code register is cleared, the processor receives the interrupt. The MCU responds to this interrupt by 1) saving the present CPU state on the stack, 2) fetching the timer interrupt vector, and 3) executing the interrupt routine. The timer interrupt request bit must be cleared by software. Refer to **RESETS** and **INTERRUPTS** for additional information.

The prescaler is a 7-bit divider which is used to extend the maximum length of the timer. To avoid truncation errors, the prescaler is cleared when TCR bit 3 is set to a logic 1; however, the TCR bit 3 always reads as a logic 0 to ensure proper operation with read-modify-write instructions.

The timer continues to count past zero, falling from $00 through $FF, and continues the countdown. The counter can be read at any time by reading the timer data register (TDR). This allows a program to determine the length of time since a timer interrupt has occurred without disturbing the counting process. Three machine cycles are required for a change in state of the TIMER pin to decrement the timer prescaler.

Clock input to the timer can be from an external source or from the internal phase two signal. Clock source is one of the mask options available. A prescaler mask option is also available that can provide up to a maximum of 128 counts to the clock input.

NOTE

If $\phi 2$ is used timer input should be tied to V_{CC}. If low, it will gate $\phi 2$ off.

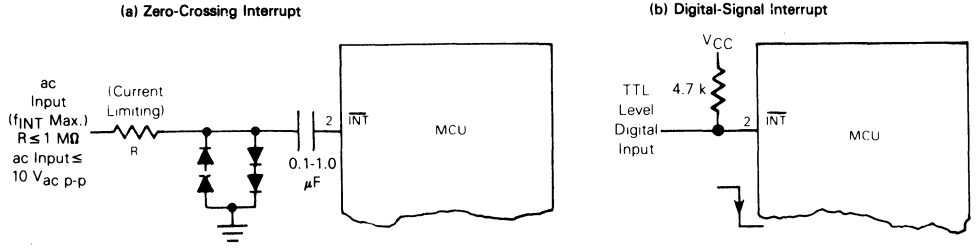


Figure 9. Typical Interrupt Circuits

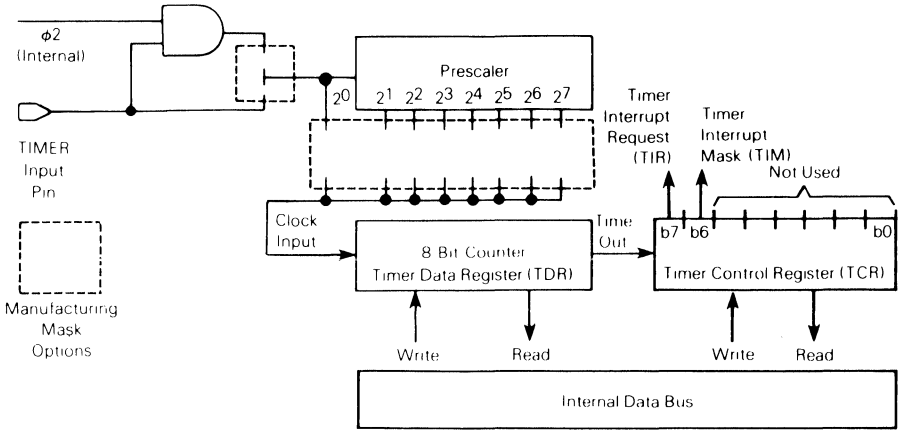


Figure 10. Timer Block Diagram

3

TIMER CONTROL REGISTER (TCR) $009

This 8-bit register controls timer interrupt request and inhibit signals.

7	6	5	4	3	2	1	0
TIR	TIM	1	1	1	1	1	1
RESET:							
0	1	U	U	U	U	U	U

TIR — Timer Interrupt Request
Used to indicate the timer interrupt when it is logic one
1 = Set when the timer data register changes to all zeros
0 = Cleared by external reset, power-on reset, or under program control

TIM — Timer Interrupt Mask
Used to inhibit the timer interrupt
1 = Interrupt inhibited
0 = Interrupt enabled

Bits 5 through 0
Not used

INSTRUCTION SET

The MCU has a set of 59 basic instructions. They can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following list of instructions.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the first 256 bytes of the memory space, where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear and bit test, and branch functions are all implemented with a single instruction. For test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition

code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch if Bit n is Set	BRSET n (n=0 . . . 7)
Branch if Bit n is Clear	BRCLR n (n=0 . . . 7)
Set Bit n	BSET n (n=0 . . . 7)
Clear Bit n	BCLR n (n=0 . . . 7)

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (2's Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Higher	BHI
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
(Branch if Higher or Same)	(BHS)
Branch if Carry Set	BCS
(Branch if Lower)	(BLO)
Branch if Not Equal	BNE
Branch if Equal	BEQ

— Continued —

Function	Mnemonic
Branch if Half Carry Clear	BHCC
Branch if Half Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Bit is Clear	BMC
Branch if Interrupt Mask Bit is Set	BMS
Branch if Interrupt Line is Low	BIL
Branch if Interrupt Line is High	BIH
Branch to Subroutine	BSR

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No-Operation	NOP

OPCODE MAP SUMMARY

Table 3 is an opcode map for the instructions used on the MCU.

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single byte instructions, while the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. Two byte direct addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

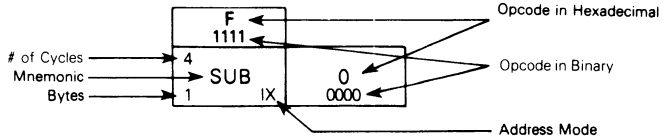
Table 3. Opcode Map

	Bit Manipulation		Branch	Read-Modify-Write					Control		Register/Memory						
	BTB	BSC	REL	DIR	INH	INH	IX1	IX	INH	INH	IMM	DIR	EXT	IX2	IX1	IX	
Hi / Low	0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111	8 1000	9 1001	A 1010	B 1011	C 1100	D 1101	E 1110	F 1111	Hi / Low
0 0000	10 BRSET0 3 BTB	7 BSET0 2 BSC	4 BRA 2 REL	6 NEG 2 DIR	4 NEG 1 INH	4 NEG 1 INH	7 NEG 2 IX1	6 NEG 1 IX	9 RTI 1 INH		2 SUB 2 IMM	4 SUB 2 DIR	5 SUB 3 EXT	6 SUB 3 IX2	5 SUB 2 IX1	4 SUB 1 IX	0 0000
1 0001	10 BRCLR0 3 BTB	7 BCLR0 2 BSC	4 BRN 2 REL						6 RTS 1 INH		2 CMP 2 IMM	4 CMP 2 DIR	5 CMP 3 EXT	6 CMP 3 IX2	5 CMP 2 IX1	4 CMP 1 IX	1 0001
2 0010	10 BRSET1 3 BTB	7 BSET1 2 BSC	4 BHI 2 REL								2 SBC 2 IMM	4 SBC 2 DIR	5 SBC 3 EXT	6 SBC 3 IX2	5 SBC 2 IX1	4 SBC 1 IX	2 0010
3 0011	10 BRCLR1 3 BTB	7 BCLR1 2 BSC	4 BLS 2 REL	6 COM 2 DIR	4 COMA 1 INH	4 COMX 1 INH	7 COM 2 IX1	6 COM 1 IX	11 SWI 1 INH		2 CPX 2 IMM	4 CPX 2 DIR	5 CPX 3 EXT	6 CPX 3 IX2	5 CPX 2 IX1	4 CPX 1 IX	3 0011
4 0100	10 BRSET2 3 BTB	7 BSET2 2 BSC	4 BCC 2 REL	6 LSR 2 DTR	4 LSRA 1 INH	4 LSRX 1 INH	7 LSR 2 IX1	6 LSR 1 IX			2 AND 2 IMM	4 AND 2 DIR	5 AND 3 EXT	6 AND 3 IX2	5 AND 2 IX1	4 AND 1 IX	4 0100
5 0101	10 BRCLR2 3 BTB	7 BCLR2 2 BSC	4 BCS 2 REL								2 BIT 2 IMM	4 BIT 2 DIR	5 BIT 3 EXT	6 BIT 3 IX2	5 BIT 2 IX1	4 BIT 1 IX	5 0101
6 0110	10 BRSET3 3 BTB	7 BSET3 2 BSC	4 BNE 2 REL	6 ROR 2 DIR	4 RORA 1 INH	4 RORX 1 INH	7 ROR 2 IX1	6 ROR 1 IX			2 LDA 2 IMM	4 LDA 2 DIR	5 LDA 3 EXT	6 LDA 3 IX2	5 LDA 2 IX1	4 LDA 1 IX	6 0110
7 0111	10 BRCLR3 3 BTB	7 BCLR3 2 BSC	4 BEQ 2 REL	6 ASR 2 DIR	4 ASRA 1 INH	4 ASRX 1 INH	7 ASR 2 IX1	6 ASR 1 IX		2 TAX 1 INH		5 STA 2 DIR	6 STA 3 EXT	7 STA 3 IX2	6 STA 2 IX1	5 STA 1 IX	7 0111
8 1000	10 BRSET4 3 BTB	7 BSET4 2 BSC	4 BHCC 2 REL	6 LSL 2 DIR	4 LSLA 1 INH	4 LSLX 1 INH	7 LSL 2 IX1	6 LSL 1 IX		2 CLC 1 INH	2 EOR 2 IMM	4 EOR 2 DIR	5 EOR 3 EXT	6 EOR 3 IX2	5 EOR 2 IX1	4 EOR 1 IX	8 1000
9 1001	10 BRCLR4 3 BTB	7 BCLR4 2 BSC	4 BHCS 2 REL	6 ROL 2 DIR	4 ROLA 1 INH	4 ROLX 1 INH	7 ROL 2 IX1	6 ROL 1 IX		2 SEC 1 INH	2 ADC 2 IMM	4 ADC 2 DIR	5 ADC 3 EXT	6 ADC 3 IX2	5 ADC 2 IX1	4 ADC 1 IX	9 1001
A 1010	10 BRSET5 3 BTB	7 BSET5 2 BSC	4 BPL 2 REL	6 DEC 2 DIR	4 DECA 1 INH	4 DECX 1 INH	7 DEC 2 IX1	6 DEC 1 IX		2 CLI 1 INH	2 ORA 2 IMM	4 ORA 2 DIR	5 ORA 3 EXT	6 ORA 3 IX2	5 ORA 2 IX1	4 ORA 1 IX	A 1010
B 1011	10 BRCLR5 3 BTB	7 BCLR5 2 BSC	4 BMI 2 REL							2 SEI 1 INH	2 ADD 2 IMM	4 ADD 2 DIR	5 ADD 3 EXT	6 ADD 3 IX2	5 ADD 2 IX1	4 ADD 1 IX	B 1011
C 1100	10 BRSET6 3 BTB	7 BSET6 2 BSC	4 BMC 2 REL	6 INC 2 DIR	4 INCA 1 INH	4 INCX 1 INH	7 INC 2 IX1	6 INC 1 IX		2 RSP 1 INH		3 JMP 2 DIR	4 JMP 3 EXT	5 JMP 3 IX2	4 JMP 2 IX1	3 JMP 1 IX	C 1100
D 1101	10 BRCLR6 3 BTB	7 BCLR6 2 BSC	4 BMS 2 REL	6 TST 2 DIR	4 TSTA 1 INH	4 TSTX 1 INH	7 TST 2 IX1	6 TST 1 IX		2 NOP 1 INH	8 BSR 2 REL	7 JSR 2 DIR	8 JSR 3 EXT	9 JSR 3 IX2	8 JSR 2 IX1	7 JSR 1 IX	D 1101
E 1110	10 BRSET7 3 BTB	7 BSET7 2 BSC	4 BIL 2 REL								2 LDX 2 IMM	4 LDX 2 DIR	5 LDX 3 EXT	6 LDX 3 IX2	5 LDX 2 IX1	4 LDX 1 IX	E 1110
F 1111	10 BRCLR7 3 BTB	7 BCLR7 2 BSC	4 BIH 2 REL	6 CLR 2 DIR	4 CLRA 1 INH	4 CLRX 1 INH	7 CLR 2 IX1	6 CLR 1 IX		2 TXA 1 INH		5 STX 2 DIR	6 STX 3 EXT	7 STX 3 IX2	6 STX 2 IX1	5 STX 1 IX	F 1111

Abbreviations for Address Modes

INH	Inherent
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction.

RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −126 to +129 from the opcode address.

INDEX, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. This addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this 2-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This addressing mode can be used in a manner similar to indexed, 8-bit offset except that this 3-byte instruction allows tables to be anywhere in memory.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte to which the specified bit is to be set or cleared. Any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single 2-byte instruction.

CAUTION

The corresponding DDRs for ports, A, B, and C are write only registers (registers at $004, $005, and $006). A read operation on these registers always returns a "1". Since BSET and BCLR are read-modify-write functions, these instructions cannot be used to set or clear a DDR bit (all "unaffected" bits would be set). It is recommended that all DDR bits in a port be written using a single-store instruction.

BIT TEST and BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit that is to be tested and its condition (set or clear) is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single 3-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltages (Except Timer in Self-Check Mode) Self-Check Mode (TIMER Pin Only)	V_{in}	 −0.3 to +7.0 −0.3 to +15.0	V
Operating Temperature Range	T_A	T_L to T_H 0 to 70 −40 to +85°C*	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature Plastic Cerdip	T_J	 150 175	°C/W

*Available at additional cost

These devices contain circuitry to protect the inputs against damage due to high static voltages or electric fields; however, normal precautions should be taken to avoid application of any voltage higher than the maximum rated voltages to this high-impedance circuit. For proper operation, V_{in} and V_{out} should be constrained to the range $V_{SS} \leq (V_{in}$ and $V_{out}) \leq V_{CC}$. Reliability of operation is enhanced if unused inputs except EXTAL are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Cerdip Plastic	θ_{JA}	 60 72	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
$P_D = P_{INT} + P_{I/O}$
$P_{INT} = I_{CC} \times V_{CC}$, Watts – Chip Internal Power
$P_{I/O}$ = Power Dissipation on Input and Output Pins – User Determined

For most applications $P_{I/O} < P_{INT}$ and can be neglected.

The following is an approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected):

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.25 ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0° to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage	V_{IH}				V
$\overline{RESET}$ (4.75 ≤ V_{CC} ≤ 5.75)		4.0	—	V_{CC}	
(V_{CC} < 4.75)		V_{CC} −0.5	—	V_{CC}	
$\overline{INT}$ (4.75 ≤ V_{CC} ≤ 5.75)		4.0	*	V_{CC}	
(V_{CC} < 4.75)		V_{CC} −0.5	*	V_{CC}	
All Other		2.0	—	V_{CC}	
Input High Voltage Timer	V_{IH}				V
Timer Mode		2.0	—	V_{CC} +1	
Self-Check Mode		9.0	10.0	15.0	
Input Low Voltage	V_{IL}				V
$\overline{RESET}$		V_{SS}	—	0.8	
$\overline{INT}$		V_{SS}	*	1.5	
All Other		V_{SS}	—	0.8	
$\overline{RESET}$ Hysteresis Voltage					V
"Out of Reset"	V_{IRES-}	2.1	—	4.0	
"Into Reset"	V_{IRES+}	0.8	—	2.0	
$\overline{INT}$ Zero Crossing Input Voltage, Through a Capacitor	V_{INT}	2.0	—	4.0	$V_{ac\ p\text{-}p}$
Internal Power Dissipation - No Port Loading V_{CC} = 5.75 V, T_A = 0°C	P_{INT}	—	400	690	mW
Input Capacitance	C_{in}				pF
XTAL		—	25	—	
All Other		—	10	—	
Low Voltage Recover	V_{LVR}	—	—	4.75	V
Low Voltage Inhibit 0°C to 70°C	V_{LVI}	2.75	3.5	—	V
−40°C to +85°C		3.1	3.5	—	
Input Current	I_{in}				μA
$\overline{TIMER}$ (V_{in} = 0.4 V)		—	—	20	
$\overline{INT}$ (V_{in} = 2.4 V to V_{CC})		—	20	50	
EXTAL (V_{in} = 2.4 V to V_{CC}, Crystal Option)		—	—	10	
(V_{in} = 0.4 V, Crystal Option)		—	—	−1600	
$\overline{RESET}$ (V_{in} = 0.8 V)		−4.0	—	−40	
(External Capacitor Charging Current)					

*Due to internal biasing, this input (when unused) floats to approximately 2.0 Vdc.

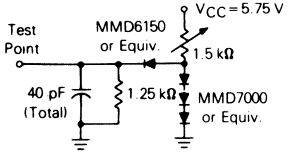


Figure 11. TTL Equivalent Test Load (Port B)

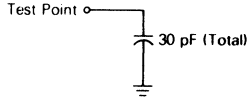


Figure 12. CMOS Equivalent Test Load (Port A)

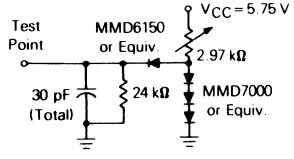


Figure 13. TTL Equivalent Test Load (Ports A and C)

PORT DC ELECTRICAL CHARACTERISTICS

(V_{CC} = 5.25 ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0° to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Port A with CMOS Drive Enabled					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Output High Voltage, I_{Load} = −10 μA	V_{OH}	V_{CC} − 1	—	—	V
Input High Voltage, I_{Load} = −300 μA (max.)	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage, I_{Load} = −500 μA (max.)	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current (V_{in} = 2.0 V to V_{CC})	I_{IH}	—	—	−300	μA
Hi-Z State Input Current (V_{in} = 0.4 V)	I_{IL}	—	—	−500	μA
Port B					
Output Low Voltage, I_{Load} = 3.2 mA	V_{OL}	—	—	0.4	V
Output Low Voltage, I_{Load} = 10 mA (sink)	V_{OL}	—	—	1.0	V
Output High Voltage, I_{Load} = −200 μA	V_{OH}	2.4	—	—	V
Darlington Current Drive (Source), V_O = 1.5 V	I_{OH}	−1.0	—	−10	mA
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	2	10	μA
Port C and Port A with CMOS Drive Disabled					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	2	10	μA

SWITCHING CHARACTERISTICS

(V_{CC} = +5.25 ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0° to 70°C, unless other noted)

Characteristic		Symbol	Min	Typ	Max	Unit
Oscillator Frequency	MC6805P2	f_{osc}	0.4	—	4.2	MHz
	MC68A05P2		0.4	—	6.0	
	MC68B05P2		0.4	—	8.0	
Cycle Time ($4/f_{osc}$)		t_{cyc}	0.95	—	10	μs
$\overline{INT}$ and TIMER Pulse Width (see Interrupt Section)		t_{WL}, t_{WH}	t_{cyc} + 250	—	—	ns
$\overline{RESET}$ Pulse Width		t_{RWL}	t_{cyc} + 250	—	—	ns
$\overline{RESET}$ Delay Time (External Capacitance = 1.0 μF)		t_{RHL}	—	100	—	ms
$\overline{INT}$ Zero Crossing Detection Input Frequency		f_{INT}	0.03	—	1.0	kHz
External Clock Input Duty Cycle (EXTAL)		—	40	50	60	%

3

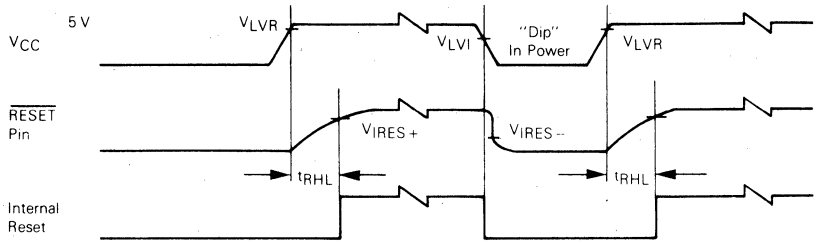


Figure 14. Power and Reset Timing

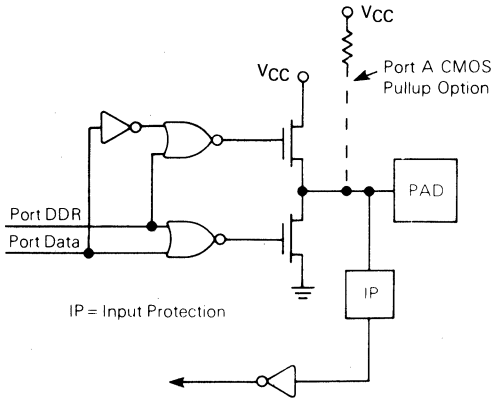


Figure 15. Ports A and C Logic Diagram

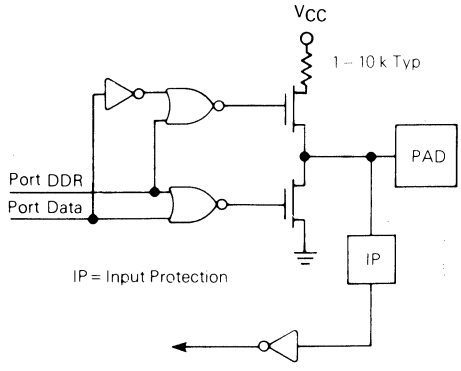


Figure 16. Port B Logic Diagram

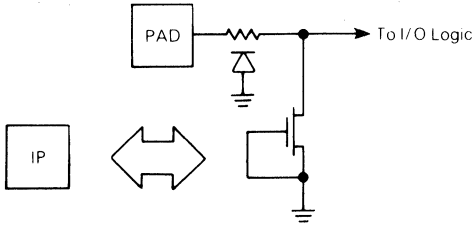


Figure 17. Typical Input Protection

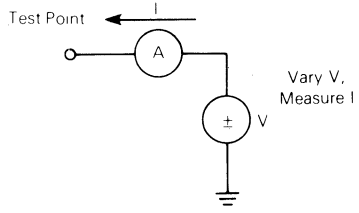


Figure 18. I/O Characteristic Measurement Circuit

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola in the following media:

MDOS, disk file
MS-DOS/PC-DOS disk file
EPROM(s) 2516, 2716, or 68705P3

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, sales person, or Motorola representative.

NOTE

The low cost resistor oscillator option is not available on B54F mask.

FLEXIBLE DISKS

Several types of flexible disks (MDOS™ or MS™-DOS/PC-DOS disk file), programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. In either case, the diskette should be clearly labeled with the customer's name, date, project or product name, and the name of the file containing the pattern.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to expedite the process in case of any difficulty with the pattern file.

MDOS Disk File

MDOS is Motorola's Disk Operating System available on the EXORciser® development system. The disk media submitted must be a single-sided, single density, 8-inch MDOS compatible floppy diskette. The diskette must contain the minimum set of MDOS system files in addition to the pattern file.

The .LO output of the M6805 cross assembler should be furnished. In addition, the file must be produced (using the ROLLOUT command) containing the absolute image of the M6805 memory. It is necessary to include the entire memory image of both data and program space. All unused bytes, including those in the user space, must be set to zero.

MS-DOS/PC-DOS Disk File

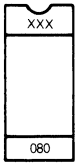
MS-DOS is Microsoft's Disk Operating System. PC-DOS is IBM® Personal Computer (PC) Disk Operating System. Disk media submitted must be a standard density (360K) double-sided 5 1/4 inch compatible floppy diskette. The diskette must contain object file code in Motorola's S-record format. The S-record format is a character-based object file format generated by M6805 cross assemblers and linkers on IBM PC style machines.

EPROMs

A 2516, 2716, or 68705P3 type EPROM, programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. Since all program and data space information will fit on one of these EPROM devices, the EPROM must be programmed as described in the following paragraphs.

The program space ROM must start at EPROM address $080. If the customer program starts at any other address, the EPROM must be marked accordingly. All unused bytes, including the user's space, must be set at zero. For shipment to Motorola, EPROMs should be placed in a conductive IC carrier and packed securely. Styrofoam is not acceptable for shipment.

EPROM MARKING



XXX = Customer ID

VERIFICATION MEDIA

All original pattern media (EPROMs or floppy disk) are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked, and the verification form should be completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for creation of the customer mask. To aid in the verification process, Motorola will program *customer supplied* blank EPROM(s) or DOS disks from the data file used to create the custom mask.

ROM VERIFICATION UNITS (RVUs)

Ten MCUs containing the customer's ROM pattern will be sent for program verification. These units will have been made using the custom mask but are for the purpose of ROM verification only. For expediency, the MCUs are usually unmarked, packaged in ceramic, and tested with five volts at room temperature. These RVUs are free with a minimum order quantity but are not production parts. These RVUs are not guaranteed by Motorola Quality Assurance.

MDOS is a trademark of Motorola Inc.
MS-DOS is a trademark of Microsoft, Inc.
EXORciser is a registered trademark of Motorola Inc.
IBM is a registered trademark of International Business Machines Corporation.

The following table provides generic information pertaining to the package type, temperature, and MC order numbers for the MC6805P2.

Table 4. Generic Information

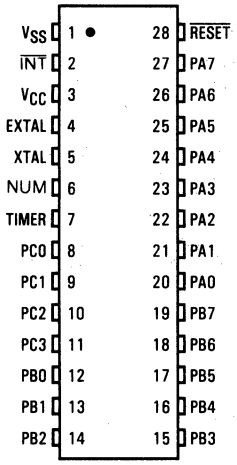
Package Type	Internal Clock Frequency (MHz)	Temperature	Order Number
Plastic (P Suffix)	1.0	0° to 70°C	MC6805P2P
	1.5	0° to 70°C	MC68A05P2P
	2.0	0° to 70°C	MC68B05P2P
	1.0	−40° to +85°C	MC6805P2CP
Cerdip (S Suffix)	1.0	0° to 70°C	MC6805P2S
	1.5	0° to 70°C	MC68A05P2S
	2.0	0° to 70°C	MC68B05P2S
PLCC (FN Suffix)	1.0	0° to 70°C	MC6805P2FN
	1.0	−40° to +85°C	MC6805P2CFN

MECHANICAL DATA

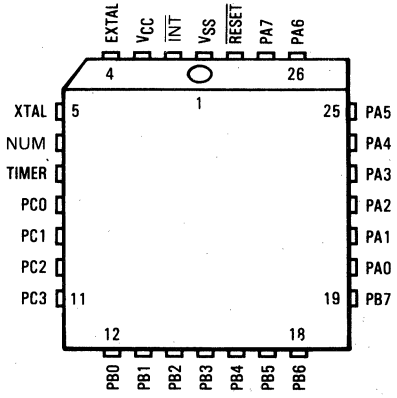
3

PIN ASSIGNMENTS

28-PIN DUAL-IN-LINE PACKAGE



28-LEAD PLCC PACKAGE



MC6805P6

Technical Summary
8-Bit Microcontroller Unit

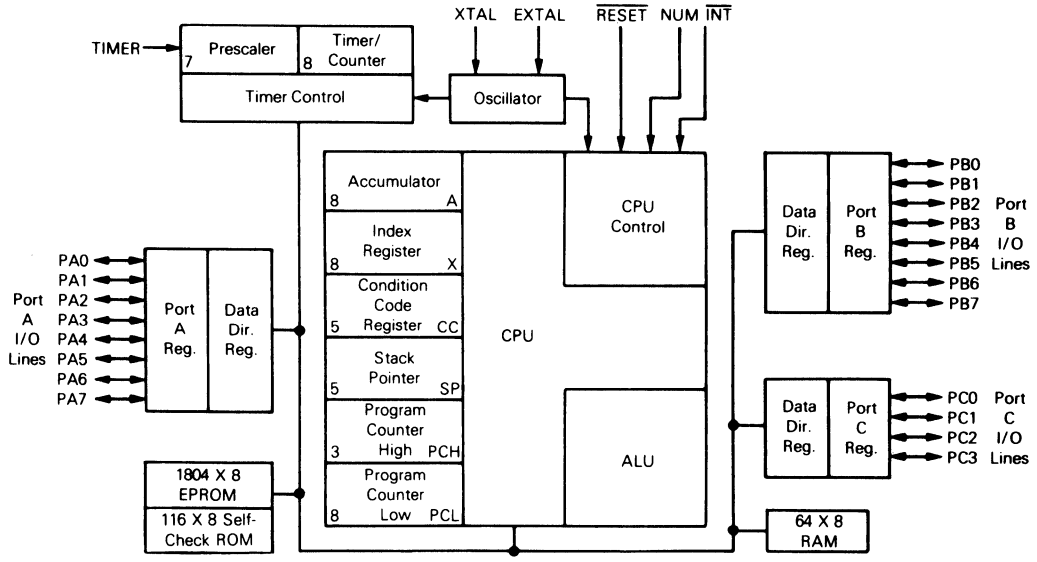
The MC6805P6 (HMOS) Microcontroller Unit (MCU) is a member of the MC6805 Family of microcontrollers. This low cost and high-speed MCU has parallel I/O capability with pins programmable as input or output. This publication contains condensed information on the MCU; for detailed information, refer to *M6805 HMOS, M146805 CMOS Family User's Manual* (M6805UM(AD2)) or contact your local Motorola sales office.

Refer to the block diagram for the hardware features and to the list below for additional features available on the MCU.

- Internal 8-Bit Timer with 7-Bit Programmable Prescaler
- On-chip Oscillator
- Memory Mapped I/O
- Versatile Interrupt Handling
- True Bit Manipulation
- Bit Test and Branch Instruction
- Vectored Interrupts
- Low Voltage Inhibit Option
- Self-Check Mode
- Master Reset
- 1804 Bytes ROM
- 64 Bytes RAM
- 20 I/O Ports

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

V_{CC} AND V_{SS}

Power is supplied to the microcontroller using these two pins. V_{CC} is 5.25 volts (±0.5Δ) power, and V_{SS} is ground.

NUM (Non-User Mode)

This pin is not for user applications and must be connected to V_{SS}.

$\overline{INT}$

This pin provides the capability for asynchronously applying an external interrupt to the MCU. Refer to **INTERRUPTS** for more detailed information.

EXTAL, XTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a ceramic resonator, a resistor/capacitor combination, or an external signal (depending upon selected manufacturing mask options) can be connected to these pins to provide a system clock.

RC Oscillator

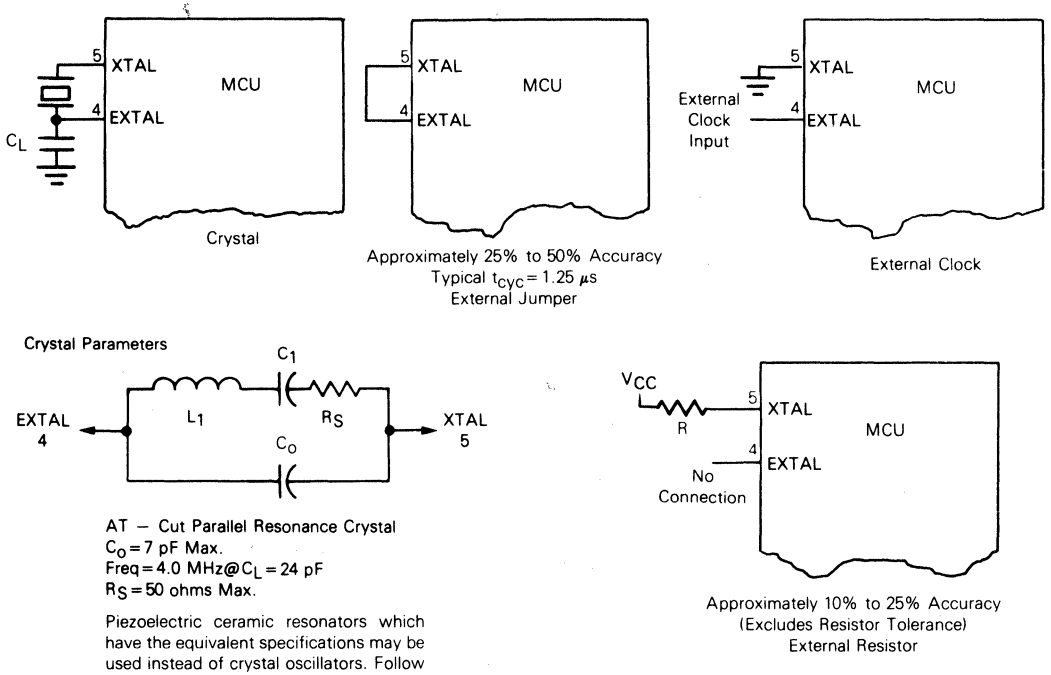
With this option, a resistor is connected to the oscillator pins as shown in Figure 1. The relationship between R and f_{osc} is shown in Figure 2.

Crystal

The circuit shown in Figure 1 is recommended when using a crystal. Using an external CMOS oscillator is recommended when crystals outside the specified ranges are to be used. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time. Refer to **ELECTRICAL SPECIFICATIONS** for V_{CC} specifications.

External Clock

An external clock should be applied to the EXTAL input with the XTAL input connected to ground, as shown in Figure 1.



NOTE: The recommended C_L value with a 4.0 MHz crystal is 27 pF maximum including system distributed capacitance. There is an internal capacitance of approximately 25 pF on the XTAL pin. For crystal frequencies other than 4 MHz, the total capacitance on each pin should be scaled as the inverse of the frequence ratio. For example, with a 2 MHz crystal, use approximately 50 pF on EXTAL and approximately 25 pF on XTAL. The exact value depends on the motional-arm parameters of the crystal used.

Figure 1. Oscillator Connections

TIMER

This pin can be used as an external input to control the internal timer/counter circuitry or for gating $\phi 2$ input to timer, depending on mask option.

$\overline{\text{RESET}}$

This pin has a Schmitt trigger input and an on-chip pullup. The MCU can be reset by pulling $\overline{\text{RESET}}$ low.

INPUT/OUTPUT LINES (PA0-PA7, PB0-PB7, PC0-PC3)

These 20 lines are arranged into two 8-bit ports (A and B) and one 4-bit port (C). Ports A, B, and C are programmable as either inputs or outputs under software control of the data direction registers. Refer to **PROGRAMMING** for additional information.

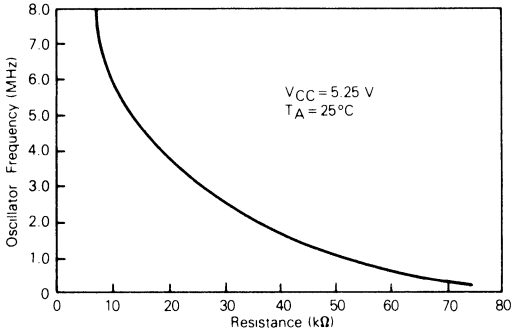


Figure 2. Typical Frequency vs Resistance for RC Oscillator Option Only

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

Any port pin is programmable as either input or output under software control of the corresponding write-only data direction register (DDR). The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic 1 for output and a logic 0 for input. On reset, all the DDRs are initialized to a logic 0 state to put the ports in the input mode. The port output registers are not initialized on reset, and should be written to before setting the DDR bits.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This port write may be used to initialize the data registers and avoid undefined outputs. Care must be exercised when using read-modify-write instructions since the data read corresponds to the pin level if the DDR is an input (0) and also to the latched output when the DDR is an output (1). Refer to Table 1 for I/O functions and to Figure 3 for typical port circuitry.

Table 1. I/O Pin Functions

Data Direction Register Bit	Output Data Bit	Output State	Input To MCU
1	0	0	0
1	1	1	1
0	X	Hi-Z	Pin

MEMORY

The MCU is capable of addressing 2048 bytes of memory and I/O registers. The memory map is shown in Figure 4. The locations consist of 1668 bytes of user ROM, user self-check ROM, user RAM, a timer control register, and I/O. The user interrupt vectors are located from $7F8 to $7FF.

The stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer decrements during pushes and increments during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

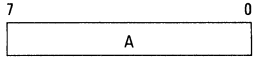
Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

REGISTERS

The MCU contains the registers described in the following paragraphs.

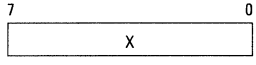
ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



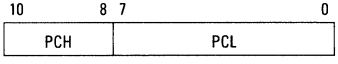
INDEX REGISTER (X)

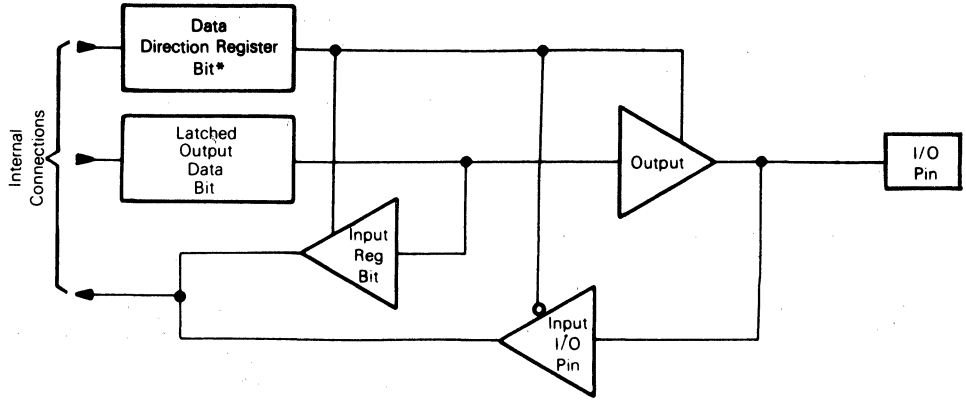
The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16-bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



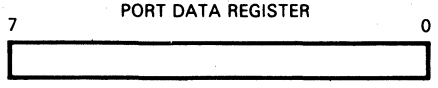
PROGRAM COUNTER (PC)

The program counter is an 11-bit register that contains the address of the next byte to be fetched.

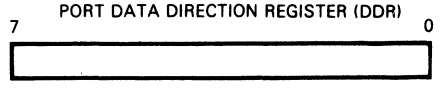




*DDR is a write-only register and reads as all "1s".



Port A Addr = $000
Port B Addr = $001
Port C Addr = $002 (Bits 0→3)



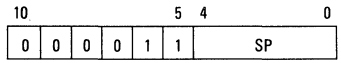
(1) Write Only; reads as all "1s"
(2) 1 = Output; 0 = Input. Cleared to 0 by reset.
(3) Port A Addr = $004
Port B Addr = $005
Port C Addr = $006 (Bits 0→3)

Figure 3. Typical Port I/O Circuitry and Register Configuration

STACK POINTER (SP)

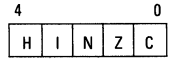
The stack pointer is an 11-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set at location $07F. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

The six most-significant bits of the stack pointer are permanently set at 000011. Subroutines and interrupts may be nested down to location $061 (31 bytes maximum), which allows the programmer to use up to 15 levels of subroutine calls (less if interrupts are allowed).



CONDITION CODE REGISTER (CC)

The condition code register is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program, and specifications can be taken as a result of their state. Each bit is explained in the following paragraphs.



Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, the timer and external interrupt is masked (disabled). If an external interrupt occurs while this bit is set, the interrupt is latched and is processed as soon as the interrupt bit is cleared.

Negative (N)

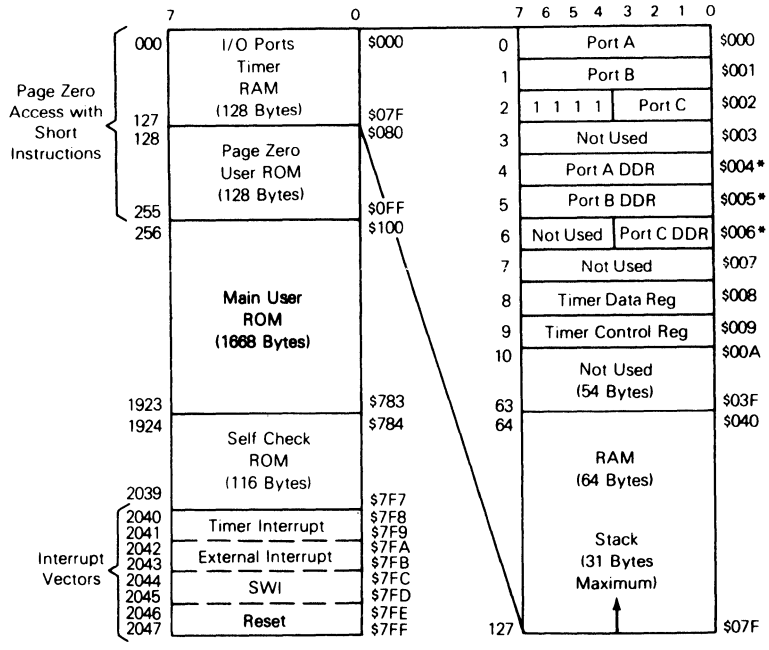
When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic 1).

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/Borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions, and during shifts and rotates.



*Caution: Data direction registers (DDRs) are write-only, set to $FF.

Figure 4. Memory Map

SELF CHECK

The self check is initiated by connecting the MCU as shown in Figure 5 and then monitoring the output of port C (bit 3) for an oscillation of approximately 7 Hz. The following tests are executed automatically:

I/O — functionally exercise I/O ports
RAM — walking bit test
ROM — exclusive OR with ODD "1s" parity result
TIMER — functionally exercise timer
Interrupts — functionally exercise external and timer interrupts

Table 2 shows the status of the LEDs as a result of a failure. Port C is tested only once (just after reset). If port C fails, no lights will appear.

Table 2. Self-Check Error Patterns

PC1	PC0	Function
0	0	Interrupt Failure
0	1	Bad Port A or Port B
1	0	Bad RAM
1	1	Bad RAM
All 4 LEDs Flasing		Good Device

RESETS

The MCU can be reset three ways: (1) by initial power-up, (2) by the external reset input ($\overline{\text{RESET}}$), and (3) by an optional, internal, low-voltage detect circuit. The $\overline{\text{RESET}}$ input consists mainly of a Schmitt trigger that senses the $\overline{\text{RESET}}$ line logic level.

POWER-ON-RESET (POR)

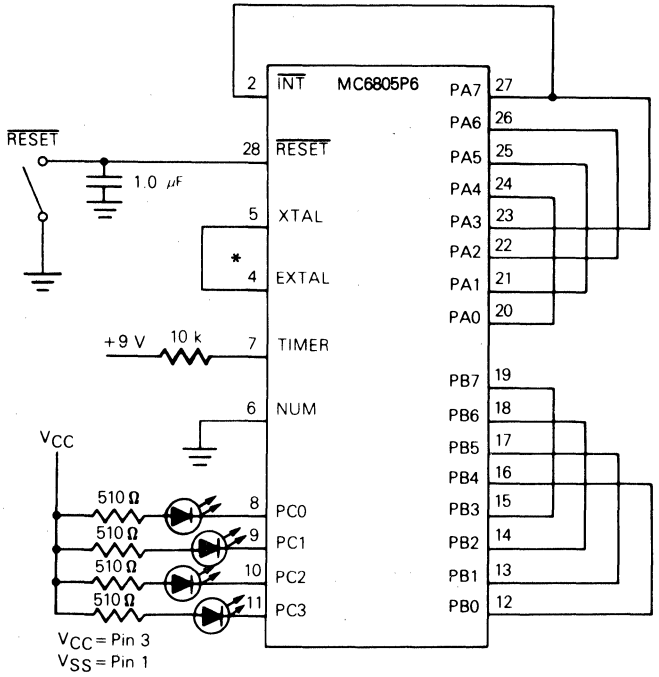
An internal reset is generated on power-up that allows the internal clock generator to stabilize. The power-on reset is used strictly for power turn-on conditions and should not be used to detect any drop in the power supply voltage. A delay of t_{RHL} milliseconds is required before allowing the $\overline{\text{RESET}}$ input to go high. Connecting a capacitor to the $\overline{\text{RESET}}$ input (Figure 6) typically provides sufficient delay.

EXTERNAL RESET INPUT

The MCU is reset when a logic zero is applied to the $\overline{\text{RESET}}$ input for a period longer than one machine cycle (t_{cyc}). Under this type of reset, the Schmitt trigger switches off at V_{IRES-} to provide an internal reset voltage.

LOW-VOLTAGE INHIBIT (LVI)

The optional low-voltage detection circuit causes a reset of the MCU if the power supply voltage falls below a



*This connection depends on the clock oscillator user selectable mask option. Use crystal if crystal option is selected.

Figure 5. Self-Check Connections

certain level (V_{LVI}). The only requirement being that V_{CC} must remain at or below the V_{LVI} threshold for one t_{cyc} minimum.

In typical applications, the V_{CC} bus filter capacitor will eliminate negative-going voltage glitches of less than one t_{cyc}. The output from the low-voltage detector is connected directly to the internal reset circuitry. It also forces the RESET pin low via a strong discharge device through a resistor. The internal reset is removed once the power supply voltage rises above a recovery level (V_{LVR}), at which time a normal power-on reset occurs.

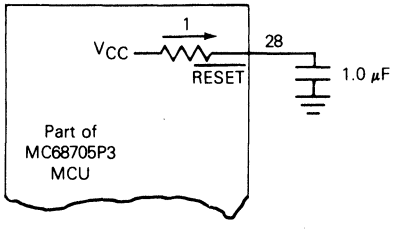


Figure 6. Power-up $\overline{\text{RESET}}$ Delay Circuit

INTERRUPTS

The MCU can be interrupted three different ways: (1) through the external interrupt $\overline{\text{INT}}$ input pin, (2) with the internal timer interrupt request, or (3) using the software interrupt instruction (SWI).

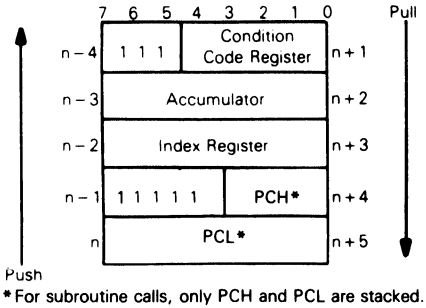
Interrupts cause the processor registers to be saved on the stack and the interrupt mask (I bit) set to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack and then normal processing resumes. The stacking order is shown in Figure 7.

Unlike $\overline{\text{RESET}}$, hardware interrupts do not cause the current instruction execution to be halted but are considered pending until the current instruction is complete.

NOTE

The current instruction is considered to be the one already fetched and being operated on.

When the current instruction is complete, the processor checks all pending hardware interrupts and, if unmasked, (I bit clear) proceeds with interrupt processing; otherwise, the next instruction is fetched and executed. Masked interrupts are latched for later interrupt service. If the



*For subroutine calls, only PCH and PCL are stacked.

Figure 7. Interrupt Stacking Order

timer interrupt status bit is cleared before unmasking the interrupt, then the interrupt is not latched.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction regardless of the setting of the I bit. Refer to Figure 8 for the reset and interrupt instruction processing sequence.

TIMER INTERRUPT

If the timer mask bit (TCR6) is cleared, then each time the timer decrements to zero (transitions from $01 to $00) an interrupt request is generated. The actual processor interrupt is generated only if the interrupt mask bit of the condition code register (CCR) is also cleared. When the

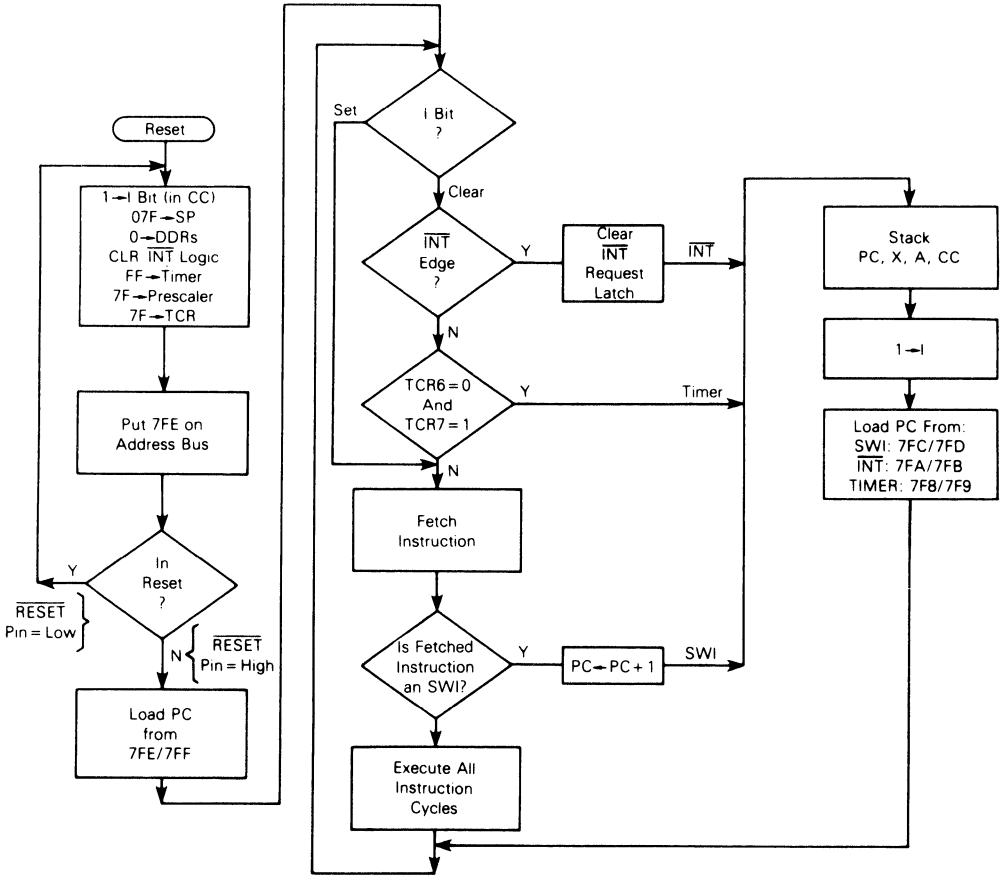


Figure 8. Reset and Interrupt Processing Flowchart

interrupt is recognized, the current state of the machine is pushed onto the stack, and the I bit in the CCR is set, masking further interrupts until the present one is serviced. The contents of the timer interrupt vector, containing the location of the timer interrupt service routine, is then loaded into the program counter. At the end of the timer interrupt service routine, the software normally executes an RTI instruction which restores the machine state and starts executing the interrupted program.

EXTERNAL INTERRUPT

The external interrupt is internally synchronized and then latched on the falling edge of $\overline{INT}$. Clearing the I bit enables the external interrupt. The MC6805P6 only requires negative edge-sensitive trigger interrupts. The following paragraphs describe two typical external interrupt circuits.

Zero-Crossing

A sinusoidal input signal (f_{INT} maximum) can be used to generate an external interrupt (see Figure 9a) for use as a zero-crossing detector (for negative transitions of the ac sinusoid). This type of circuit allows applications such as servicing time-of-day routines and engaging/disengaging ac power control devices. Off-chip, full-wave rectification provides an interrupt at every zero crossing of the ac signal and thereby provides a 2f clock.

Digital-Signal Interrupt

With this type of circuit (Figure 9b), the INT pin can be driven by a digital signal. The maximum frequency of a signal that can be recognized by the TIMER or INT pin logic is dependent on the parameter labeled t_{WL}, t_{WH}. Refer to **TIMER** for additional information.

SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. If the I bit is zero, SWI executes after the other interrupts. The SWI execution is similar to the hardware interrupts.

TIMER

The MCU consists of an 8-bit software programmable counter driven by a 7-bit software programmable prescaler. The timer source is made during manufacturing as a mask option. The 8-bit counter may be loaded under program control and is decremented toward zero. When the timer reaches zero, the timer interrupt request bit (bit 7) in the timer control register (TCR) is set. Refer to Figure 10 for timer block diagram.

The timer interrupt can be masked (disabled) by setting the timer interrupt mask bit (bit 6) in the TCR. When the I bit in the condition code register is cleared, the processor receives the interrupt. The MCU responds to this interrupt by 1) saving the present CPU state on the stack, 2) fetching the timer interrupt vector, and 3) executing the interrupt routine. The timer interrupt request bit must be cleared by software. Refer to **RESETS** and **INTERRUPTS** for additional information.

The prescaler is a 7-bit divider which is used to extend the maximum length of the timer. To avoid truncation errors, the prescaler is cleared when TCR bit 3 is set to a logic 1; however, the TCR bit 3 always reads as a logic 0 to ensure proper operation with read-modify-write instructions.

The timer continues to count past zero, falling from $00 through $FF, and continues the countdown. The counter can be read at any time by reading the timer data register (TDR). This allows a program to determine the length of time since a timer interrupt has occurred without disturbing the counting process. Three machine cycles are required for a change in state of the TIMER pin to decrement the timer prescaler.

Clock input to the timer can be from an external source or from the internal phase two signal. Clock source is one of the mask options available. A prescaler mask option is also available that can provide up to a maximum of 128 counts to the clock input.

NOTE

If $\phi 2$ is used, Timer input should be tied to V_{CC}. If low, it will gate $\phi 2$ off.

TIMER CONTROL REGISTER (TCR) $009

This 8-bit register controls the timer interrupt request and inhibit signals. All bits are read/write except bit 3.

7	6	5	4	3	2	1	0
TIR	TIM	1	1	1	1	1	1
RESET:							
0	1	U	U	U	U	U	U

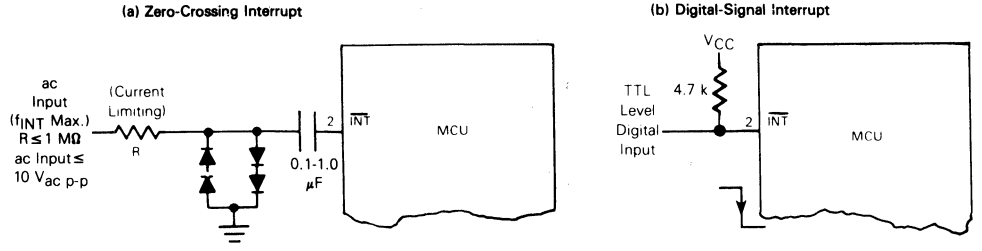


Figure 9. Typical Interrupt Circuits

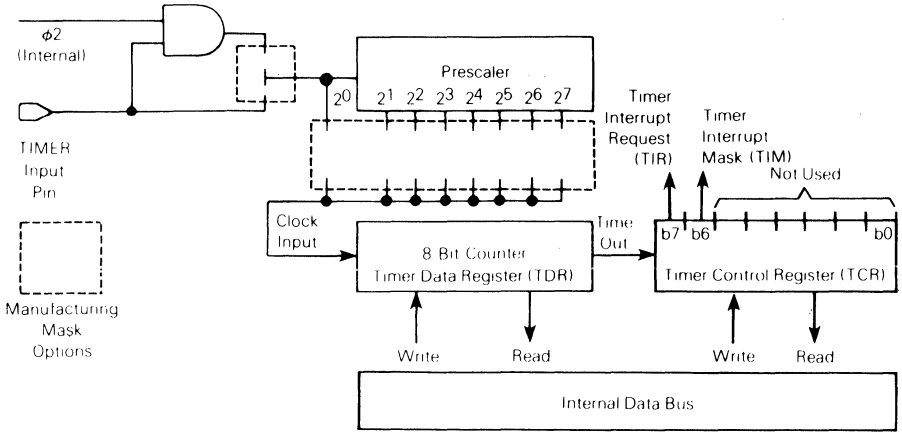


Figure 10. Timer Block Diagram

TIR — Timer Interrupt Request
Used to indicate the timer interrupt when it is logic one
1 = Set when the timer data register changes to all zeros
0 = Cleared by external reset, power-on reset, or under program control
TIM — Timer Interrupt Mask
Used to inhibit the timer interrupt.
1 = Interrupt inhibited
0 = Interrupt enabled
Bits 5 through 0
Not used

INSTRUCTION SET

The MCU has a set of 59 basic instructions. They can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following list of instructions.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the first 256 bytes of the memory space, where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear and bit test, and branch functions are all implemented with a single instruction. For test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch if Bit n is Set	BRSET n (n = 0 . . . 7)
Branch if Bit n is Clear	BRCLR n (n = 0 . . . 7)
Set Bit n	BSET n (n = 0 . . . 7)
Clear Bit n	BCLR n (n = 0 . . . 7)

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (2's Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Higher	BHI
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
(Branch if Higher or Same)	(BHS)
Branch if Carry Set	BCS
(Branch if Lower)	(BLO)
Branch if Not Equal	BNE
Branch if Half Carry Clear	BHCC
Branch if Half Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Bit is Clear	BMC
Branch if Interrupt Mask Bit is Set	BMS
Branch if Interrupt Line is Low	BIL
Branch if Interrupt Line is High	BIH
Branch to Subroutine	BSR

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No-Operation	NOP

OPCODE MAP SUMMARY

Table 3 is an opcode map for the instructions used on the MCU.

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single byte instructions while the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. Two byte direct addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the

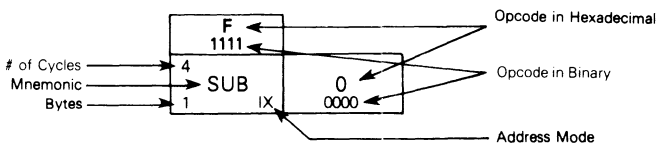
Table 3. Opcode Map

Hi / Low	Bit Manipulation BTB 0 0000	Bit Manipulation BSC 1 0001	Branch REL 2 0010	Read-Modify-Write DIR 3 0011	Read-Modify-Write INH 4 0100	Read-Modify-Write INH 5 0101	Read-Modify-Write IX1 6 0110	Read-Modify-Write IX 7 0111	Control INH 8 1000	Control INH 9 1001	Register/Memory IMM A 1010	Register/Memory DIR B 1011	Register/Memory EXT C 1100	Register/Memory IX2 D 1101	Register/Memory IX1 E 1110	Register/Memory IX F 1111	Hi / Low
0 0000	10 BRSET0 3 BTB	7 BSET0 2 BSC	4 BRA 2 REL	6 NEG 2 DIR	4 NEG 1 INH	4 NEG 1 INH	7 NEG 2 IX1	6 NEG 1 IX	9 RTI 1 INH		2 SUB 2 IMM	4 SUB 2 DIR	5 SUB 3 EXT	6 SUB 3 IX2	5 SUB 2 IX1	4 SUB 1 IX	0 0000
1 0001	10 BRCLR0 3 BTB	7 BCLR0 2 BSC	4 BRN 2 REL						6 RTS 1 INH		2 CMP 2 IMM	4 CMP 2 DIR	5 CMP 3 EXT	6 CMP 3 IX2	5 CMP 2 IX1	4 CMP 1 IX	1 0001
2 0010	10 BRSET1 3 BTB	7 BSET1 2 BSC	4 BHI 2 REL								2 SBC 2 IMM	4 SBC 2 DIR	5 SBC 3 EXT	6 SBC 3 IX2	5 SBC 2 IX1	4 SBC 1 IX	2 0010
3 0011	10 BRCLR1 3 BTB	7 BCLR1 2 BSC	4 BLS 2 REL	6 COM 2 DIR	4 COMA 1 INH	4 COMX 1 INH	7 COM 2 IX1	6 COM 1 IX	11 SWI 1 INH		2 CPX 2 IMM	4 CPX 2 DIR	5 CPX 3 EXT	6 CPX 3 IX2	5 CPX 2 IX1	4 CPX 1 IX	3 0011
4 0100	10 BRSET2 3 BTB	7 BSET2 2 BSC	4 BCC 2 REL	6 LSR 2 DTR	4 LSRA 1 INH	4 LSRX 1 INH	7 LSR 2 IX1	6 LSR 1 IX			2 AND 2 IMM	4 AND 2 DIR	5 AND 3 EXT	6 AND 3 IX2	5 AND 2 IX1	4 AND 1 IX	4 0100
5 0101	10 BRCLR2 3 BTB	7 BCLR2 2 BSC	4 BCS 2 REL								2 BIT 2 IMM	4 BIT 2 DIR	5 BIT 3 EXT	6 BIT 3 IX2	5 BIT 2 IX1	4 BIT 1 IX	5 0101
6 0110	10 BRSET3 3 BTB	7 BSET3 2 BSC	4 BNE 2 REL	6 ROR 2 DIR	4 RORA 1 INH	4 RORX 1 INH	7 ROR 2 IX1	6 ROR 1 IX			2 LDA 2 IMM	4 LDA 2 DIR	5 LDA 3 EXT	6 LDA 3 IX2	5 LDA 2 IX1	4 LDA 1 IX	6 0110
7 0111	10 BRCLR3 3 BTB	7 BCLR3 2 BSC	4 BEQ 2 REL	6 ASR 2 DIR	4 ASRA 1 INH	4 ASRX 1 INH	7 ASR 2 IX1	6 ASR 1 IX		2 TAX 1 INH		5 STA 2 DIR	6 STA 3 EXT	7 STA 3 IX2	6 STA 2 IX1	5 STA 1 IX	7 0111
8 1000	10 BRSET4 3 BTB	7 BSET4 2 BSC	4 BHCC 2 REL	6 LSL 2 DIR	4 LSLA 1 INH	4 LSLX 1 INH	7 LSL 2 IX1	6 LSL 1 IX		2 CLC 1 INH	2 EOR 2 IMM	4 EOR 2 DIR	5 EOR 3 EXT	6 EOR 3 IX2	5 EOR 2 IX1	4 EOR 1 IX	8 1000
9 1001	10 BRCLR4 3 BTB	7 BCLR4 2 BSC	4 BHCS 2 REL	6 ROL 2 DIR	4 ROLA 1 INH	4 ROLX 1 INH	7 ROL 2 IX1	6 ROL 1 IX		2 SEC 1 INH	2 ADC 2 IMM	4 ADC 2 DIR	5 ADC 3 EXT	6 ADC 3 IX2	5 ADC 2 IX1	4 ADC 1 IX	9 1001
A 1010	10 BRSET5 3 BTB	7 BSET5 2 BSC	4 BPL 2 REL	6 DEC 2 DIR	4 DECA 1 INH	4 DECX 1 INH	7 DEC 2 IX1	6 DEC 1 IX		2 CLI 1 INH	2 ORA 2 IMM	4 ORA 2 DIR	5 ORA 3 EXT	6 ORA 3 IX2	5 ORA 2 IX1	4 ORA 1 IX	A 1010
B 1011	10 BRCLR5 3 BTB	7 BCLR5 2 BSC	4 BMI 2 REL							2 SEI 1 INH	2 ADD 2 IMM	4 ADD 2 DIR	5 ADD 3 EXT	6 ADD 3 IX2	5 ADD 2 IX1	4 ADD 1 IX	B 1011
C 1100	10 BRSET6 3 BTB	7 BSET6 2 BSC	4 BMC 2 REL	6 INC 2 DIR	4 INCA 1 INH	4 INCX 1 INH	7 INC 2 IX1	6 INC 1 IX		2 RSP 1 INH		3 JMP 2 DIR	4 JMP 3 EXT	5 JMP 3 IX2	4 JMP 2 IX1	3 JMP 1 IX	C 1100
D 1101	10 BRCLR6 3 BTB	7 BCLR6 2 BSC	4 BMS 2 REL	6 TST 2 DIR	4 TSTA 1 INH	4 TSTX 1 INH	7 TST 2 IX1	6 TST 1 IX		2 NOP 1 INH	8 BSR 2 REL	7 JSR 2 DIR	8 JSR 3 EXT	9 JSR 3 IX2	8 JSR 2 IX1	7 JSR 1 IX	D 1101
E 1110	10 BRSET7 3 BTB	7 BSET7 2 BSC	4 BIL 2 REL								2 LDX 2 IMM	4 LDX 2 DIR	5 LDX 3 EXT	6 LDX 3 IX2	5 LDX 2 IX1	4 LDX 1 IX	E 1110
F 1111	10 BRCLR7 3 BTB	7 BCLR7 2 BSC	4 BIH 2 REL	6 CLR 2 DIR	4 CLRA 1 INH	4 CLRX 1 INH	7 CLR 2 IX1	6 CLR 1 IX		2 TXA 1 INH		5 STX 2 DIR	6 STX 3 EXT	7 STX 3 IX2	6 STX 2 IX1	5 STX 1 IX	F 1111

Abbreviations for Address Modes

INH	Inherent
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction.

RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −126 to +129 from the opcode address.

3

INDEX, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. This addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this 2-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This addressing mode can be used in a manner similar to indexed, 8-bit offset except that this 3-byte instruction allows tables to be anywhere in memory.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte to which the specified bit is to be set or cleared. Any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single 2-byte instruction.

CAUTION

The corresponding DDRs for ports, A, B, and C are write only registers (registers at $004, $005, and $006). A read operation on these registers is undefined. Since BSET and BCLR are read-modify-write functions, these instructions cannot be used to set or clear a DDR bit (all "unaffected" bits would be set). It is recommended that all DDR bits in a port be written using a single-store instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit that is to be tested and its condition (set or clear) is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single 3-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltages Except Timer in Self-Check Mode Self-Check Mode (TIMER Pin Only)	V_{in}	 −0.3 to +7.0 −0.3 to +15.0	V
Operating Temperature Range	T_A	T_L to T_H 0 to 70 −40 to +85*	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature Plastic Cerdip	T_J	 150 175	°C/W

*Available at additional cost

These devices contain circuitry to protect the inputs against damage due to high static voltages or electric fields; however, normal precautions should be taken to avoid application of any voltage higher than the maximum rated voltages to this high-impedance circuit. For proper operation, V_{in} and V_{out} should be constrained to the range $V_{SS} \leq (V_{in}$ and $V_{out}) \leq V_{CC}$. Reliability of operation is enhanced if unused inputs except EXTAL are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Cerdip Plastic	θ_{JA}	 60 72	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
$P_D = P_{INT} + P_{I/O}$
$P_{INT} = I_{CC} \times V_{CC}$, Watts – Chip Internal Power
$P_{I/O}$ = Power Dissipation on Input and Output Pins – User Determined

For most applications $P_{I/O} < P_{INT}$ and can be neglected.

The following is an approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected):

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.25 ±5 Vdc, V_{SS} = 0 Vdc, T_A = 0°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage	V_{IH}				V
$\overline{RESET}$ (4.75 ≤ V_{CC} ≤ 5.75)		4.0	—	V_{CC}	
(V_{CC} < 4.75)		V_{CC} − 0.5	—	V_{CC}	
$\overline{INT}$ (4.75 ≤ V_{CC} ≤ 5.75)		4.0	*	V_{CC}	
(V_{CC} < 4.75)		V_{CC} − 0.5	*	V_{CC}	
All Other		2.0	—	V_{CC}	
Input High Voltage Timer	V_{IH}				V
Timer Mode		2.0	—	V_{CC} + 1	
Self-Check Mode		9.0	10.0	15.0	
Input Low Voltage	V_{IL}				V
$\overline{INT}$		V_{SS}	*	1.5	
All Other		V_{SS}	—	0.8	
$\overline{RESET}$ Hysteresis Voltage					V
"Out of Reset"	V_{IRES+}	2.1	—	4.0	
"Into Reset"	V_{IRES-}	0.8	—	2.0	
$\overline{INT}$ Zero Crossing Input Voltage, Through a Capacitor	V_{INT}	2.0	—	4.0	$V_{ac\ p-p}$
Internal Power Dissipation - No Port Loading V_{CC} = 5.75 V, T_A = 0°C	P_{INT}	—	400	690	mW
Input Capacitance	C_{in}				pF
XTAL		—	25	—	
All Other		—	10	—	
Low Voltage Recover	V_{LVR}	—	—	4.75	V
Low Voltage Inhibit 0°C to 70°C	V_{LVI}	2.75	3.5	—	V
−40°C to +85°C		3.1	3.5	—	
Input Current (External Capacitor Charging Current)	I_{in}				μA
$\overline{TIMER}$ (V_{in} = 0.4 V)		—	—	20	
$\overline{INT}$ (V_{in} = 2.4 V to V_{CC}		—	20	50	
EXTAL (V_{in} = 2.4 V to V_{CC}, Crystal Option)		—	—	10	
(V_{in} = 0.4 V, Crystal Option)		—	—	−1600	
$\overline{RESET}$ (V_{in} = 0.8 V)		−4.0	—	−40	

*Due to internal biasing, this input (when unused) floats to approximately 2.0 Vdc.

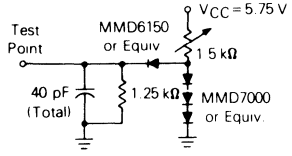


Figure 11. TTL Equivalent Test Load (Port B)

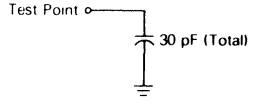


Figure 12. CMOS Equivalent Test Load (Port A)

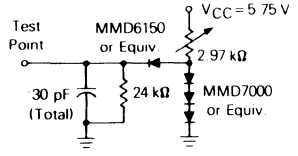


Figure 13. TTL Equivalent Test Load (Ports A and C)

PORT DC ELECTRICAL CHARACTERISTICS

(V_{CC} = 5.25 ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0° to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Port A with CMOS Drive Enabled					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Output High Voltage, I_{Load} = −10 μA	V_{OH}	V_{CC} −1	—	—	V
Input High Voltage, I_{Load} = −300 μA (max.)	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage, I_{Load} = −500 μA (max.)	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current (V_{in} = 2.0 V to V_{CC})	I_{IH}	—	—	−300	μA
Hi-Z State Input Current (V_{in} = 0.4 V)	I_{IL}	—	—	−500	μA
Port B					
Output Low Voltage, I_{Load} = 3.2 mA	V_{OL}	—	—	0.4	V
Output Low Voltage, I_{Load} = 10 mA (sink)	V_{OL}	—	—	1.0	V
Output High Voltage, I_{Load} = −200 μA	V_{OH}	2.4	—	—	V
Darlington Current Drive (Source), V_O = 1.5 V	I_{OH}	−1.0	—	−10	mA
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	2	10	μA
Port C and Port A with CMOS Drive Disabled					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	2	10	μA

SWITCHING CHARACTERISTICS

(V_{CC} = +5.25 ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0° to 70°C, unless other noted)

Characteristic		Symbol	Min	Typ	Max	Unit
Oscillator Frequency	MC6805P6	f_{osc}	0.4	—	4.2	MHz
	MC68A05P6		0.4	—	6.0	
	MC68B05P6		0.4	—	8.0	
Cycle Time ($4/f_{osc}$)		t_{cyc}	0.95	—	10	μs
$\overline{INT}$ and TIMER Pulse Width (See INTERRUPTS)		t_{WL}, t_{WH}	t_{cyc} + 250	—	—	ns
$\overline{RESET}$ Pulse Width		t_{RWL}	t_{cyc} + 250	—	—	ns
$\overline{RESET}$ Delay Time (External Capacitance = 1.0 μF)		t_{RHL}	—	100	—	ms
$\overline{INT}$ Zero Crossing Detection Input Frequency		f_{INT}	0.03	—	1.0	kHz
External Clock Input Duty Cycle (EXTAL)		—	40	50	60	%

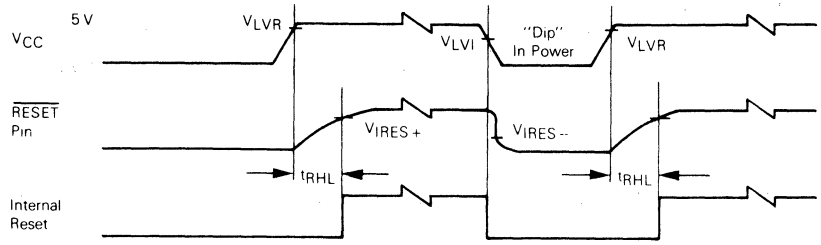


Figure 14. Power and Reset Timing

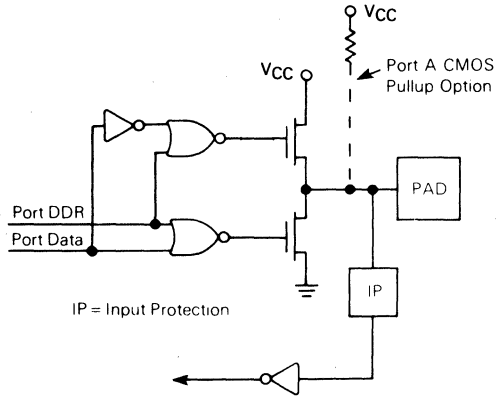


Figure 15. Ports A and C Logic Diagram

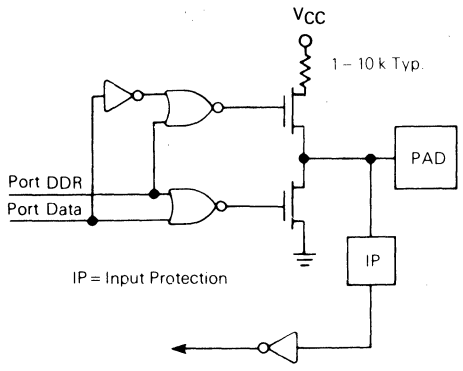


Figure 16. Port B Logic Diagram

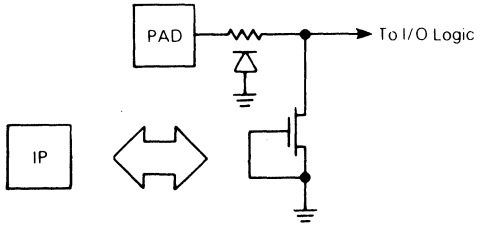


Figure 17. Typical Input Protection

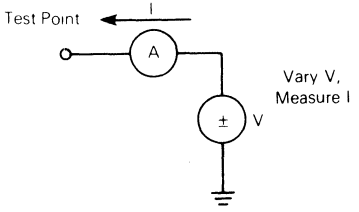


Figure 18. I/O Characteristic Measurement Circuit

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola in the following media:

MDOS, disk file
MS-DOS/PC-DOS disk file
EPROM(s) 2516, 2716, or 68705P3

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, a sales person, or Motorola representative.

FLEXIBLE DISKS

Several types of flexible disks (MDOS™ or MS™-DOS/PC-DOS disk file), programmed with the customer program (positive logic sense for address and data), may be submitted for pattern generation. In either case, the diskette should be clearly labeled with the customers name, date, project or product name, and the name of the file containing the pattern.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to expedite the process in case of any difficulty with the pattern file.

MDOS Disk File

MDOS is Motorola's Disk Operating System available on the EXORciser® development system. The disk media submitted must be a single-sided, single density, 8-inch MDOS compatible floppy diskette. The diskette must contain the minimum set of MDOS system files in addition to the pattern file.

The .LO output of the M6805 cross assembler should be furnished. In addition, the file must be produced (using the ROLLOUT command) containing the absolute image of the M6805 memory. Include the entire memory image of both data and program space. All unused bytes, including those in the user space, must be set to zero.

MS-DOS/PC-DOS Disk File

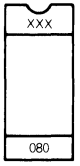
MS-DOS is Microsoft's Disk Operating System. PC-DOS is IBM® Personal Computer (PC) Disk Operating System. Disk media submitted must be a standard density (360K) double-sided 5 1/4 inch compatible floppy diskette. The diskette must contain object file code in Motorola's S-record format. The S-record format is a character-based object file format generated by M6805 cross assemblers and linkers on IBM PC style machines.

EPROMs

A 2516, 2716, or 68705P3 type EPROM, programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. The EPROM must be clearly marked to indicated which EPROM corresponds to which address space.

All unused bytes, including the user's space, must be set to zero. For shipment to Motorola, EPROMs should be placed in a conductive IC carrier and packed securely. Styrofoam is not acceptable for shipment.

EPROM MARKING



XXX = Customer ID

VERIFICATION MEDIA

All original pattern media (EPROMs or floppy disk) are filled for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked and the verification form should be completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for creation of the customer mask. To aid in the verification process, Motorola will program *customer supplied* blank EPROM(s) or DOS disk from the data file used to create the custom mask.

ROM VERIFICATION UNITS (RVUs)

Ten MCUs containing the customer's ROM pattern will be sent for program verification. These units will have been made using the custom mask but are for the purpose of ROM verification only. For expediency, the MCUs are unmarked, packaged in ceramic, and tested with five volts at room temperature. These RVUs are free with the minimum order quantity but are not production parts. These RVUs are not guaranteed by Motorola Quality Assurance.

3

MDOS is a trademark of Motorola Inc.
MS is a trademark of Microsoft, Inc.
EXORciser is a registered trademark of Motorola Inc.
IBM is a registered trademark of International Business Machines Corporation.

The following table provides generic information pertaining to the package type, temperature, and MC order numbers for the MC6805P6.

Table 4. Generic Information

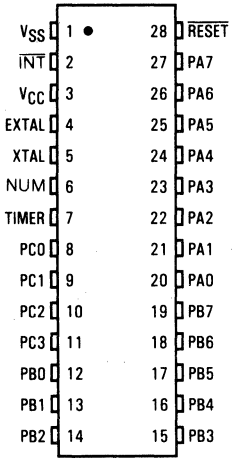
Package Type	Internal Clock Frequency (MHz)	Temperature	Order Number
Plastic (P Suffix)	1.0	0° to 70°C	MC6805P6P
	1.0	−40° to +85°C	MC6805P6CP
	1.5	0° to 70°C	MC68A05P6P
	2.0	0° to 70°C	MC68B05P6P
Cerdip (S Suffix)	1.0	0° to 70°C	MC6805P6S
	1.5	0° to 70°C	MC68A05P6S
	2.0	0° to 70°C	MC68B05P6S
PLCC (FN Suffix)	1.0	0° to 70°C	MC6805P6FN
	1.0	−40° to +85°C	MC6805P6CFN

MECHANICAL DATA

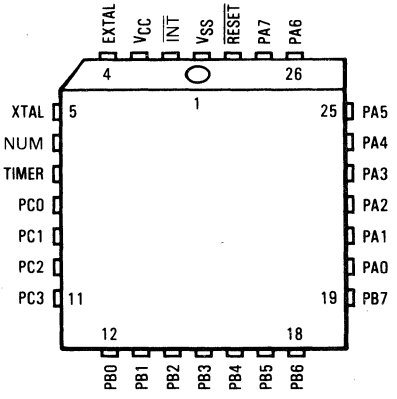
3

PIN ASSIGNMENTS

28-PIN DUAL-IN-LINE PACKAGE



28-LEAD PLCC PACKAGE



MC6805R2

Technical Summary
8-Bit Microcontroller Unit

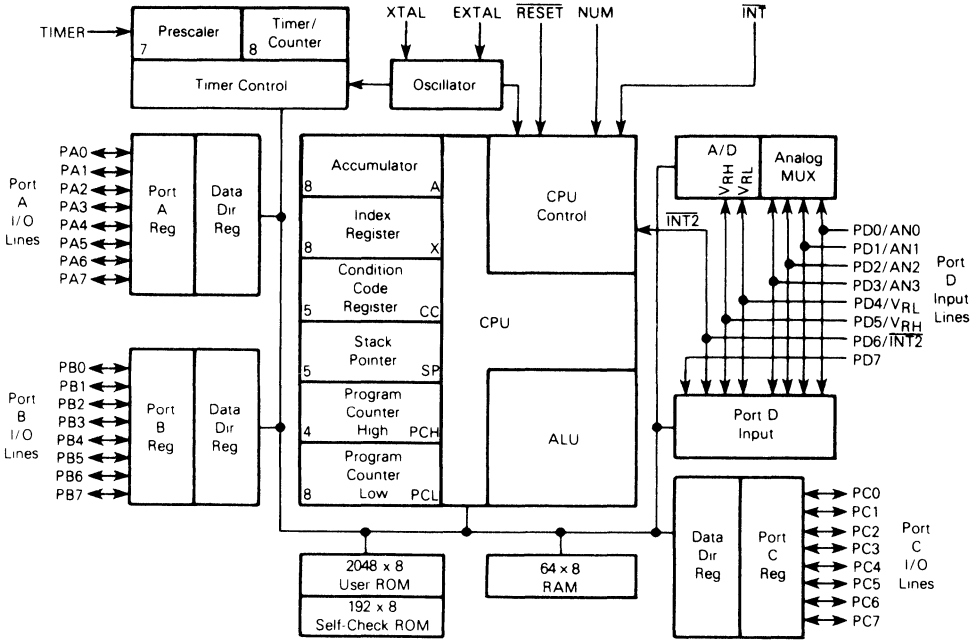
The MC6805R2 (HMOS) Microcontroller Unit (MCU) is a member of the MC6805 Family of microcontrollers. This low cost and high-speed MCU has parallel I/O capability with pins programmable as input or output. This publication contains condensed information on the MCU; for detailed information, refer to *M6805 HMOS, M146805 CMOS Family User's Manual* (M6805UM(AD2)) or contact your local Motorola sales office.

Refer to the block diagram for the hardware features and to the list below for additional features available on the MCU.

- Internal 8-Bit Timer with 7-Bit Prescaler
- On-chip Oscillator
- Memory Mapped I/O
- Versatile Interrupt Handling
- True Bit Manipulation
- Bit Test and Branch Instruction
- Vectored Interrupts
- Self-Check Mode
- 2048 Bytes of ROM
- 64 Bytes of RAM
- 24 Bidirectional I/O Ports
- A/D Converter

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

V_{CC} AND V_{SS}

Power is supplied to the microcomputer using these two pins. V_{CC} is +5.25 volts (±0.5Δ) power, and V_{SS} is ground.

NUM

This pin is not for user applications and must be connected to V_{SS}.

$\overline{INT}$

This pin provides the capability for asynchronously applying an external interrupt to the MCU. Refer to **INTERRUPTS** for more detailed information.

EXTAL, XTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a resistor/capacitor combination, or an external signal (depending upon selected manufacturing mask option) is connected to these pins to provide a system clock.

RC Oscillator

With this option, a resistor is connected to the oscillator pins as shown in Figure 1. The relationship between R and f_{OSC} is shown in Figure 2.

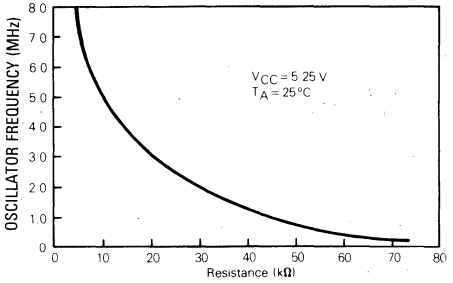
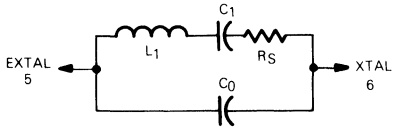


Figure 2. Typical Frequency vs Resistance for RC Oscillator Option Only

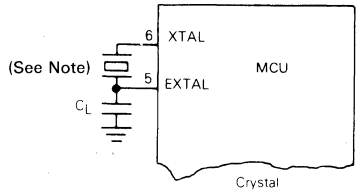
Crystal

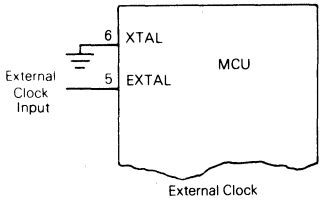
The circuit shown in Figure 1 is recommended when using a crystal. Using an external CMOS oscillator is recommended when crystals outside the specified ranges are to be used. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time. Refer to **ELECTRICAL SPECIFICATIONS** for V_{CC} specifications.

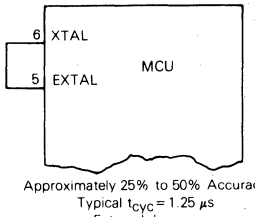


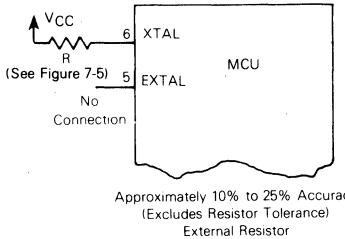
AT – Cut Parallel Resonance Crystal
C_0 = 7 pF Max
Freq. = 4.0 MHz @ C_L = 24 pF
R_S = 50 ohms Max.

Piezoelectric ceramic resonators which have the equivalent specifications may be used instead of crystal oscillators. Follow ceramic resonator manufacturer's suggestions for C_0, C_1, and R_S values.









NOTE: The recommended C_L value with a 4.0 MHz crystal is 27 pF maximum, including system distributed capacitance. There is an internal capacitance of approximately 25 pF on the XTAL pin. For crystal frequencies other than 4 MHz, the total capacitance on each pin should be scaled as the inverse of the frequency ratio. For example, with a 2 MHz crystal, use approximately 50 pF on EXTAL and approximately 25 pF on XTAL. The exact value depends on the Motional-Arm parameters of the crystal used.

Figure 1. Oscillator Connections

External Clock

An external clock should be applied to the EXTAL input with the XTAL input connected to ground, as shown in Figure 1. This option may only be used with the crystal oscillator option selected in the mask option register. The t_{OXOV} or t_{ILCH} specifications do not apply when using an external clock input.

TIMER

This pin is used as an external input to control the internal timer/counter circuitry. This pin also detects a higher voltage level used to initiate the self-test program.

$\overline{\text{RESET}}$

This pin has a Schmitt trigger input and an on-chip pullup. The MCU can be reset by pulling $\overline{\text{RESET}}$ low.

INPUT/OUTPUT LINES (PA0-PA7, PB0-PB7, PC0-PC7,PD0-PD7)

These 32 lines are arranged into four 8-bit ports (A, B, C, and D). Ports A, B, and C are programmable as either inputs or outputs under software control of the data direction registers. Port D is a fixed input port and not controlled by any data direction register. Port D has up to four analog inputs, plus two voltage reference inputs when the A/D converter is used (PD5/V_{RH}, PD4/V_{RL}) and an $\overline{\text{INT2}}$ input. All Port D lines can be read directly and used as binary inputs. If any analog input is used, then PD5/V_{RH} and PD4/V_{RL} must be used in the analog mode. Refer to **PROGRAMMING** and **ANALOG-TO-DIGITAL CONVERTER** for additional information.

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

Port A, B, and C pins are programmable as either input or output under software control of the corresponding data direction register (DDR). Port D lines are input only. The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic one for output and a logic zero for input. On reset, all the DDRs are initialized to a logic zero state to put the ports in the input mode. The port output registers are not initialized on reset and should be written to before setting the DDR bits.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This port write may be used to initialize the data registers and avoid undefined outputs. Care must be exercised when using read-modify-write instructions since the data read corresponds to the pin level if the DDR is an input (zero) and also to the latched output when the DDR is an output (one). Refer to Table 1 for I/O functions and to Figure 3 for typical port circuitry.

Port D provides reference voltage ($\overline{\text{INT2}}$) and multiplexed analog inputs. Port D can always be used as digital input and may be used for analog if V_{RH} and V_{RL} are connected to the appropriate reference voltage. The V_{RH} (PD5) and V_{RL} (PD4) are internally connected to the A/D resistor.

Table 1. I/O Pin Functions

Data Direction Register Bit	Latched Output Data Bit	Output State	Input To MCU
1	0	0	0
1	1	1	1
0	X	Hi-Z**	Pin

**Ports B and C are three-state ports. Port A has optional internal pullup devices to provide CMOS data drive capability.

MEMORY

The MCU is capable of addressing 4096 bytes of memory and I/O registers. The memory map is shown in Figure 4. The locations consist of user ROM, self-check ROM, user RAM, A/D registers, a miscellaneous control register,

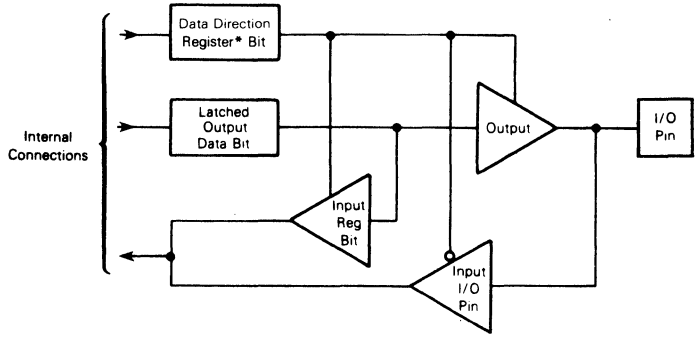
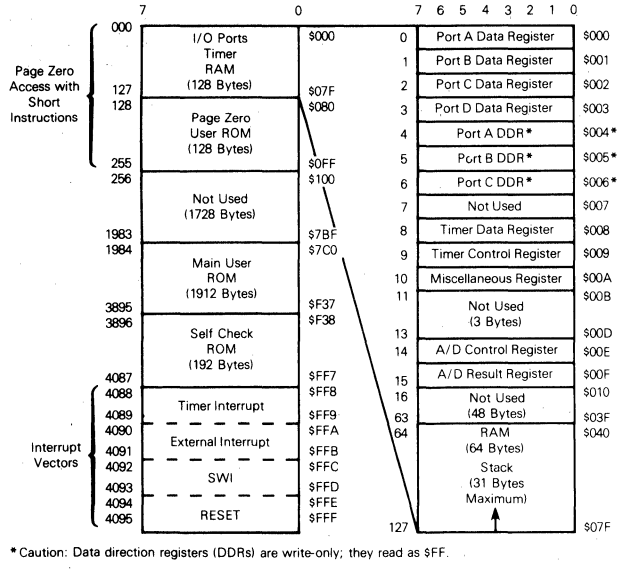


Figure 3. Typical Port I/O Circuitry and Register Configuration



*Caution: Data direction registers (DDRs) are write-only; they read as $FF.

Figure 4. Memory Map

and I/O. The interrupt and reset vectors are located from $FF8 to $FFF.

The stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer decrements during pushes and increments during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

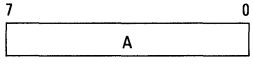
Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

REGISTERS

The MCU contains the registers described in the following paragraphs.

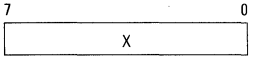
ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



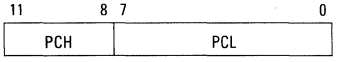
INDEX REGISTER (X)

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16-bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



PROGRAM COUNTER (PC)

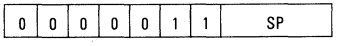
The program counter is an 12-bit register that contains the address of the next byte to be fetched.



STACK POINTER (SP)

The stack pointer is an 12-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set at location $07F. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

The seven most-significant bits of the stack pointer are permanently set at 0000011. Subroutines and interrupts may be nested down to location $061 (31 bytes maximum), which allows the programmer to use up to 15 levels of subroutine calls (less if interrupts are allowed).



CONDITION CODE REGISTER (CC)

The condition code register is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a

program, and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.

4				0
H	I	N	Z	C

Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, the timer and external interrupt is masked (disabled). If an external interrupt occurs while this bit is set, the interrupt is latched and is processed as soon as the interrupt bit is cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic one).

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/Borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions, and during shifts and rotates.

SELF-CHECK

The self-check is initiated by connecting the MCU as shown in Figure 5 and then monitoring the output of port C (bit 3) for an oscillation of approximately 7 Hz. The following test are executed automatically:

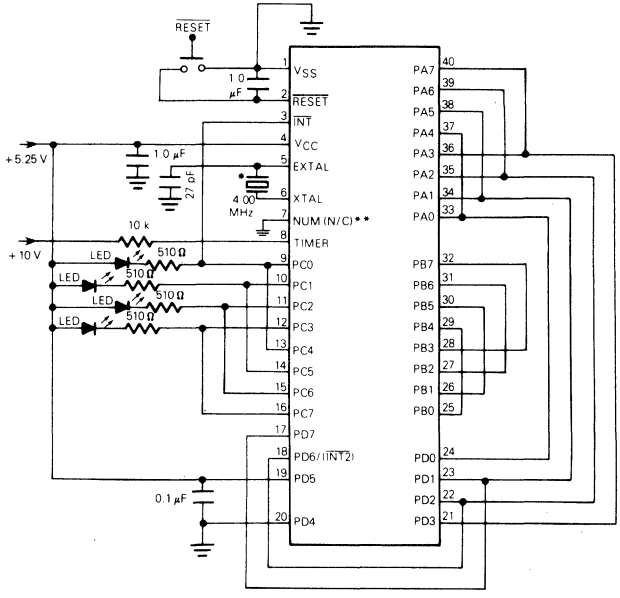
I/O — Functionally exercise I/O ports.
RAM — Walking bit test.
ROM — Exclusive OR with ODD "1st" parity result.
Timer — functionally exercise timer.
Interrupts — Functionally exercise external and timer interrupts.
A/D Converter — Functionally test the Analog-to-Digital Converter.

The RAM, ROM, and the A/D test can be called by a user program. The timer test may be called if the timer input is the internal clock. Table 2 shows the status of the LEDs as a result of a failure. Port C is tested only once (just after reset). If port C fails, no lights will appear.

Table 2. Self-Check Error Patterns

PC0	PC1	PC2	PC3	Remarks (1: LED ON; 0: LED OFF)
1	0	1	0	Bad I/O
0	0	1	0	Bad Timer
1	1	0	0	Bad RAM
0	1	0	0	Bad ROM
1	0	0	0	Bad A/D
0	0	0	0	Bad Interrupts or Request Flag
All Flashing				Good Device

Anything else Bad Part, Bad Port C, etc.



*This connection depends on clock oscillator user selectable mask option. Use jumper if the RC mask option is selected.
**Pin 7 is not for user application and must be connected to V_{SS}.

Figure 5. Self-Check Connections

RESETS

The MCU can be reset three ways: (1) by initial power-up, (2) by the external reset input ($\overline{\text{RESET}}$), and (3) by an optional, internal, low-voltage detect circuit. The $\overline{\text{RESET}}$ input consists mainly of a Schmitt trigger that senses the $\overline{\text{RESET}}$ line logic level.

POWER-ON-RESET (POR)

An internal reset is generated on power-up that allows the internal clock generator to stabilize. The power-on reset is used strictly for power turn-on conditions and should not be used to detect any drop in the power supply voltage. A delay of t_{RHL} milliseconds is required before allowing the $\overline{\text{RESET}}$ input to go high. Connecting a capacitor to the $\overline{\text{RESET}}$ input (Figure 6) typically provides sufficient delay.

EXTERNAL RESET INPUT

The MCU is reset when a logic zero is applied to the $\overline{\text{RESET}}$ input for a period longer than one machine cycle (t_{cyc}). Under this type of reset, the Schmitt trigger switches off at V_{IRES-} to provide an internal reset voltage.

LOW-VOLTAGE INHIBIT (LVI)

The optional low-voltage detection circuit causes a reset of the MCU if the power supply voltage falls below a certain level (V_{LVI}). The only requirement is that the V_{CC} must remain at or below the V_{LVI} threshold for one t_{cyc} minimum.

In typical applications, the V_{CC} bus filter capacitor will eliminate negative-going voltage glitches of less than one t_{cyc}. The output from the low-voltage detector is connected directly to the internal reset circuitry. It also forces the $\overline{\text{RESET}}$ pin low via a strong discharge device through a resistor. The internal reset is removed once the power supply voltage rises above a recovery level (V_{LVR}) at which time a normal power-on reset occurs.

INTERRUPTS

The MCU can be interrupted four different ways: (1) through the external interrupt $\overline{\text{INT}}$ input pin, (2) with the internal timer interrupt request, (3) using the software interrupt instruction (SWI) or (4) the external port D bit 6 ($\overline{\text{INT2}}$) input pin.

Interrupts cause the processor registers to be saved on the stack and the interrupt mask (I bit) set to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack, and then normal processing resumes. The stacking order is shown in Figure 7.

Unlike $\overline{\text{RESET}}$, hardware interrupts do not cause the current instruction execution to be halted but are considered pending until the current instruction is complete.

NOTE

The current instruction is considered to be the one already fetched and being operated on.

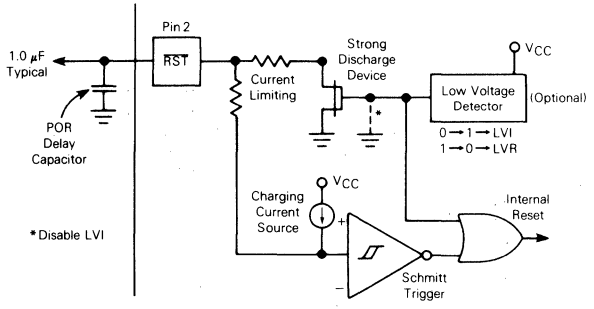
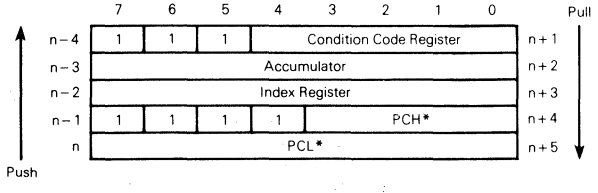


Figure 6. $\overline{\text{RESET}}$ Configuration



*For subroutine calls, only PCH and PCL are stacked.

Figure 7. Interrupt Stacking Order

When the current instruction is complete, the processor checks all pending hardware interrupts and, if unmasked (I bit clear), proceeds with interrupt processing; otherwise, the next instruction is fetched and executed. Masked interrupts are latched for later interrupt service. If the timer interrupt status bit is cleared before unmasking the interrupt, then the interrupt is not latched.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction regardless of the setting of the I bit. Refer to Figure 8 for the reset and interrupt instruction processing sequence.

TIMER INTERRUPT

If the timer mask bit (TCR6) is cleared, then, each time the timer decrements to zero (transitions from $01 to $00), an interrupt request is generated. The actual processor interrupt is generated only if the interrupt mask bit of the condition code register (CCR) is also cleared. When the interrupt is recognized, the current state of the machine is pushed onto the stack and the I bit in the CCR is set, masking further interrupts until the present one is serviced. The contents of the timer interrupt vector, containing the location of the timer interrupt service routine, is then loaded into the program counter. At the end of the timer interrupt service routine, the software normally executes an RTI instruction which restores the machine state and starts executing the interrupted program.

EXTERNAL INTERRUPT

The external interrupt is internally synchronized and then latched on the falling edge of $\overline{INT}$ and $\overline{INT2}$. Clearing the I bit enables the external interrupt. The $\overline{INT2}$ interrupt has an interrupt request bit (bit 7) and a mask bit (bit 6) in the miscellaneous register (MR). The $\overline{INT2}$ interrupt is inhibited when the mask bit is set. The $\overline{INT2}$ is always read as a digital input on port D. The $\overline{INT2}$ and timer interrupt request bits, if set, cause the MCU to process and interrupt when the condition code I bit is clear. The following paragraphs describe two typical external interrupt circuits.

Zero-Crossing Interrupt

A sinusoidal input signal (f_{INT} maximum) can be used to generate an external interrupt (see Figure 9a) for use as a zero-crossing detector (for negative transitions of the ac sinusoid). This type of circuit allows applications

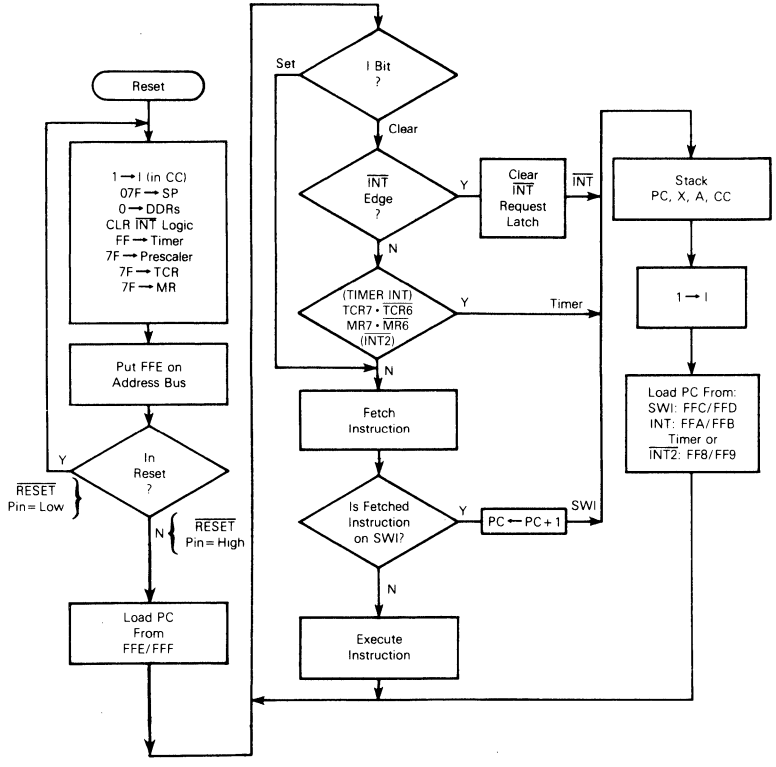


Figure 8. Reset and Interrupt Processing Flowchart

such as servicing time-of-day routines and engaging/disengaging ac power control devices. Off-chip, full-wave rectification provides an interrupt at every zero crossing of the ac signal and thereby provides a 2f clock.

Digital-Signal Interrupt

With this type of circuit (Figure 9b), the $\overline{INT}$ pin can be driven by a digital signal. The maximum frequency of a signal that can be recognized by the TIMER or $\overline{INT}$ pin logic is dependent on the parameter labeled t_{WL}, t_{WH}. Refer to **TIMER** for additional information.

SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. If the I bit is zero, SWI executes after the other interrupts. The SWI execution is similar to the hardware interrupts.

3

TIMER

The MCU consists of an 8-bit software programmable counter driven by a 7-bit prescaler. The timer source is made during manufacturing as a mask option. The 8-bit counter may be loaded under program control and is decremented toward zero. When the timer reaches zero, the timer interrupt request bit (bit 7) in the timer control register (TCR) is set. Refer to Figure 10 for timer block diagram.

The timer interrupt can be masked (disabled) by setting the timer interrupt mask bit (bit 6) in the TCR. When the I bit in the condition code register is cleared, the processor receives the interrupt. The MCU responds to this interrupt by 1) saving the present CPU state on the stack, 2) fetching the timer interrupt vector, and 3) executing the interrupt routine. The timer interrupt request bit must be cleared by software. The TIMER and $\overline{INT2}$ share the same interrupt vector, therefore the interrupt routine must check the request bits to determine the source of the interrupt. Refer to **RESETS** and **INTERRUPTS** for additional information.

The prescaler is a 7-bit divider which is used to extend the maximum length of the timer. To avoid truncation errors, the prescaler is cleared when TCR bit 3 is set to a logic one; however, the TCR bit 3 always reads as a logic zero to ensure proper operation with read-modify-write instructions.

The timer continues to count past zero, falling from $00 through $FF, and continues the countdown. The counter can be read at any time by reading the timer data register (TDR). This allows a program to determine the length of time since a timer interrupt has occurred without disturbing the counting process. Three machine cycles are required for a change in state of the TIMER pin to decrement the timer prescaler.

Clock input to the timer can be from an external source or from the internal phase two signal. Clock source is one of the mask options. A prescaler mask option is available to select a divide option of a power of two up to 128.

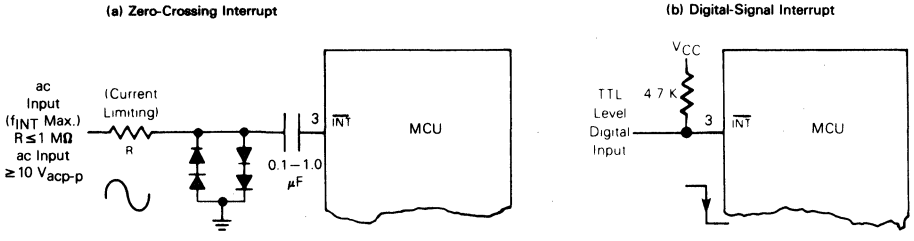


Figure 9. Typical Interrupt Circuits

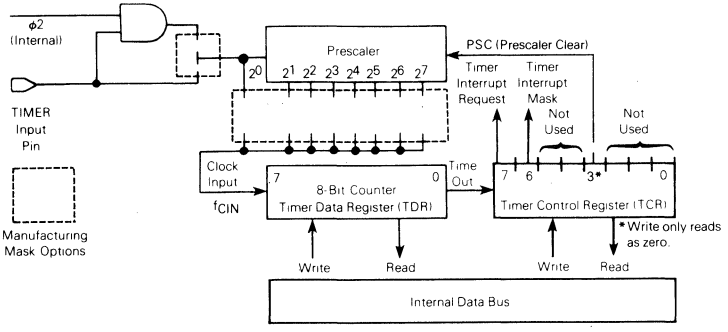


Figure 10. Timer Block Diagram

TIMER CONTROL REGISTER (TCR) ($009)

This 8-bit register controls various functions such as write timer interrupt request, timer interrupt inhibit, and prescaler clear. Bit 3 is write only.

7	6	5	4	3	2	1	0
TIR	TIM	1	1	PSC	1	1	1
RESET:							
0	1	U	U	U	U	U	U

TIR — Timer Interrupt Request
Used to indicate the timer interrupt when it is logic one.
1 = Set when the timer data register changes to all zeros.
0 = Cleared by external reset, power-on reset, or under program control.

TIM — Timer Interrupt Mask
Used to inhibit the timer interrupt.
1 = Interrupt inhibited.
0 = Interrupt enabled.

PSC — Prescaler Clear
Write only bit. Writing a one to this bit resets the prescaler to zero. A read of this location always indicates a zero.

Bits 5, 4, 2, 1, 0 — Not Used.

ANALOG-TO-DIGITAL CONVERTER

The chip resident 8-bit analog-to-digital (A/D) converter uses a successive approximation technique as shown in Figure 11. Four external analog inputs can be connected to the A/D through a multiplexer via Port D. Four internal analog channels (V_{RH}–V_{RL}, V_{RH}–$V_{RL}/2$, V_{RH}–$V_{RL}/4$, and V_{RL}) may be selected for calibration. The accuracy of these internal channels may not meet the accuracy specifications of the external channels.

Multiplexer selection is controlled by the A/D control register (ACR) bits 0, 1, and 2. Refer to Table 3 for multiplexer selection. The ACR is shown in Figure 11. The converter uses 30 machine cycles to complete a conversion of a sampled analog input. When the conversion is complete, the digital value is placed in the A/D result register (ARR), the conversion flag set, selected input is sampled again, and a new conversion starts. When ACR7 is cleared, the conversion in progress is aborted and the selected input is sampled for five machine cycles and held internally.

Table 3. A/D Input MUX Selection

A/D Control Register			Input Selected	A/D Output (Hex)		
ACR2	ACR1	ACR0		Min	Typ	Max
0	0	0	AN0			
0	0	1	AN1			
0	1	0	AN2			
0	1	1	AN3			
1	0	0	V_{RH}*	FE	FF	FF
1	0	1	V_{RL}*	00	00	01
1	1	0	$V_{RH}/4$*	3F	40	41
1	1	1	$V_{RH}/2$*	7F	80	81

*Internal (Calibration) Levels

The converter uses V_{RH} and V_{RL} as reference voltages. An input voltage equal to or greater than V_{RH} converts to $FF. An input voltage equal to or less than V_{RL}, but greater than V_{SS}, converts to $00. Maximum and minimum ratings must not be exceeded. Each analog input source should use V_{RH} as the supply voltage and be referenced to V_{RL} for the ratiometric conversion. To maintain full accuracy of the A/D, three requirements should be followed: (1) V_{RH} should be equal to or less than V_{DD}, (2) V_{RL} should be equal to or greater than V_{SS} but less than maximum specifications, and (3) V_{RH}–V_{RL} should be equal to or greater than 4 volts.

The A/D has a built-in 1/2 LSB offset intended to reduce the magnitude of the quantizing error to ±1/2 LSB, rather than +0, −1 LSB with no offset. This implies that, ignoring errors, the transition point from $00 to $01 occurs at 1/2 LSB above V_{RL}. Similarly, the transition from $FE to $FF occurs 1-1/2 LSB below V_{RH}, ideally.

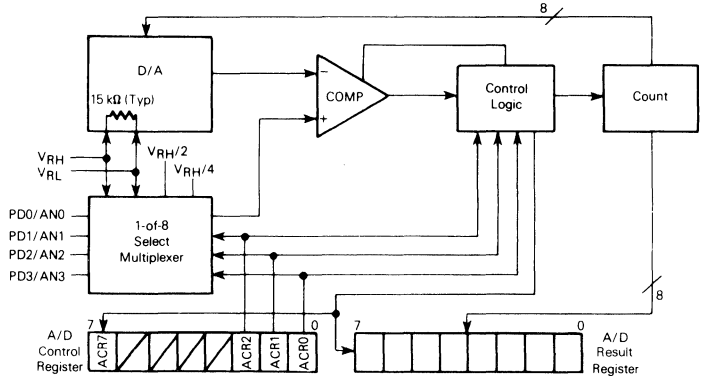


Figure 11. A/D Block Diagram

INSTRUCTION SET

The MCU has a set of 59 basic instructions which can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following list of instructions.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the first 256 bytes of the memory space, where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear and bit test, and branch functions are all implemented with a single instruction. For test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch if Bit n is Set	BRSET n (n=0 . . . 7)
Branch if Bit n is Clear	BRCLR n (n=0 . . . 7)
Set Bit n	BSET n (n=0 . . . 7)
Clear Bit n	BCLR n (n=0 . . . 7)

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (2's Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two-byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Higher	BHI
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
(Branch if Higher or Same)	(BHS)
Branch if Carry Set	BCS
(Branch if Lower)	(BLO)
Branch if Not Equal	BNE
Branch if Equal	BEQ
Branch if Half Carry Clear	BHCC
Branch if Half Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Bit is Clear	BMC
Branch if Interrupt Mask Bit is Set	BMS
Branch if Interrupt Line is Low	BIL
Branch if Interrupt Line is High	BIH
Branch to Subroutine	BSR

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No Operation	NOP

OPCODE MAP SUMMARY

Table 4 is an opcode map for the instructions used on the MCU.

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code coversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single-byte instructions, while the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. Two-byte direct-addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction.

RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −126 to +129 from the opcode address.

INDEX, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. This addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this two-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This addressing mode can be used in a manner similar to indexed, 8-bit offset except that this three-byte instruction allows tables to be anywhere in memory.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte to which the specified bit is to be set or cleared. Thus, any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single two-byte instruction.

CAUTION

The corresponding DDRs for ports A, B, and C are write-only registers (registers at $004, $005, and $006). A read operation on these registers is undefined. Since BSET and BCLR are read-modify-write

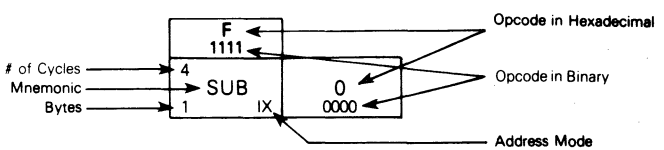
Table 4. Opcode Map

	Bit Manipulation		Branch	Read-Modify-Write					Control		Register/Memory						
Hi / Low	BTB 0 0000	BSC 1 0001	REL 2 0010	DIR 3 0011	INH 4 0100	INH 5 0101	IX1 6 0110	IX 7 0111	INH 8 1000	INH 9 1001	IMM A 1010	DIR B 1011	EXT C 1100	IX2 D 1101	IX1 E 1110	IX F 1111	Hi / Low
0 0000	10 BRSET0 3 BTB	7 BSET0 2 BSC	4 BRA 2 REL	6 NEG 2 DIR	4 NEG 1 INH	4 NEG 1 INH	7 NEG 2 IX1	6 NEG 1 IX	9 RTI 1 INH		2 SUB 2 IMM	4 SUB 2 DIR	5 SUB 3 EXT	6 SUB 3 IX2	5 SUB 2 IX1	4 SUB 1 IX	0 0000
1 0001	10 BRCLR0 3 BTB	BCLR0 2 BSC	4 BRN 2 REL						6 RTS 1 INH		2 CMP 2 IMM	4 CMP 2 DIR	5 CMP 3 EXT	6 CMP 3 IX2	5 CMP 2 IX1	4 CMP 1 IX	1 0001
2 0010	10 BRSET1 3 BTB	7 BSET1 2 BSC	4 BHI 2 REL								2 SBC 2 IMM	4 SBC 2 DIR	5 SBC 3 EXT	6 SBC 3 IX2	5 SBC 2 IX1	4 SBC 1 IX	2 0010
3 0011	10 BRCLR1 3 BTB	BCLR1 2 BSC	4 BLS 2 REL	6 COM 2 DIR	4 COMA 1 INH	4 COMX 1 INH	7 COM 2 IX1	6 COM 1 IX	11 SWI 1 INH		2 CPX 2 IMM	4 CPX 2 DIR	5 CPX 3 EXT	6 CPX 3 IX2	5 CPX 2 IX1	4 CPX 1 IX	3 0011
4 0100	10 BRSET2 3 BTB	BSET2 2 BSC	4 BCC 2 REL	6 LSR 2 DIR	4 LSRA 1 INH	4 LSRX 1 INH	7 LSR 2 IX1	6 LSR 1 IX			2 AND 2 IMM	4 AND 2 DIR	5 AND 3 EXT	6 AND 3 IX2	5 AND 2 IX1	4 AND 1 IX	4 0100
5 0101	10 BRCLR2 3 BTB	BCLR2 2 BSC	4 BCS 2 REL								2 BIT 2 IMM	4 BIT 2 DIR	5 BIT 3 EXT	6 BIT 3 IX2	5 BIT 2 IX1	4 BIT 1 IX	5 0101
6 0110	10 BRSET3 3 BTB	BSET3 2 BSC	4 BNE 2 REL	6 ROR 2 DIR	4 RORA 1 INH	4 RORX 1 INH	7 ROR 2 IX1	6 ROR 1 IX			2 LDA 2 IMM	4 LDA 2 DIR	5 LDA 3 EXT	6 LDA 3 IX2	5 LDA 2 IX1	4 LDA 1 IX	6 0110
7 0111	10 BRCLR3 3 BTB	BCLR3 2 BSC	4 BEQ 2 REL	6 ASR 2 DIR	4 ASRA 1 INH	4 ASRX 1 INH	7 ASR 2 IX1	6 ASR 1 IX		2 TAX 1 INH		5 STA 2 DIR	6 STA 3 EXT	7 STA 3 IX2	6 STA 2 IX1	5 STA 1 IX	7 0111
8 1000	10 BRSET4 3 BTB	BSET4 2 BSC	4 BHCC 2 REL	6 LSL 2 DIR	4 LSLA 1 INH	4 LSLX 1 INH	7 LSL 2 IX1	6 LSL 1 IX		2 CLC 1 INH	2 EOR 2 IMM	4 EOR 2 DIR	5 EOR 3 EXT	6 EOR 3 IX2	5 EOR 2 IX1	4 EOR 1 IX	8 1000
9 1001	10 BRCLR4 3 BTB	BCLR4 2 BSC	4 BHCS 2 REL	6 ROL 2 DIR	4 ROLA 1 INH	4 ROLX 1 INH	7 ROL 2 IX1	6 ROL 1 IX		2 SEC 1 INH	2 ADC 2 IMM	4 ADC 2 DIR	5 ADC 3 EXT	6 ADC 3 IX2	5 ADC 2 IX1	4 ADC 1 IX	9 1001
A 1010	10 BRSET5 3 BTB	BSET5 2 BSC	4 BPL 2 REL	6 DEC 2 DIR	4 DECA 1 INH	4 DECX 1 INH	7 DEC 2 IX1	6 DEC 1 IX		2 CLI 1 INH	2 ORA 2 IMM	4 ORA 2 DIR	5 ORA 3 EXT	6 ORA 3 IX2	5 ORA 2 IX1	4 ORA 1 IX	A 1010
B 1011	10 BRCLR5 3 BTB	BCLR5 2 BSC	4 BMI 2 REL							2 SEI 1 INH	2 ADD 2 IMM	4 ADD 2 DIR	5 ADD 3 EXT	6 ADD 3 IX2	5 ADD 2 IX1	4 ADD 1 IX	B 1011
C 1100	10 BRSET6 3 BTB	BSET6 2 BSC	4 BMC 2 REL	6 INC 2 DIR	4 INCA 1 INH	4 INCX 1 INH	7 INC 2 IX1	6 INC 1 IX		2 RSP 1 INH		3 JMP 2 DIR	4 JMP 3 EXT	5 JMP 3 IX2	4 JMP 2 IX1	3 JMP 1 IX	C 1100
D 1101	10 BRCLR6 3 BTB	BCLR6 2 BSC	4 BMS 2 REL	6 TST 2 DIR	4 TSTA 1 INH	4 TSTX 1 INH	7 TST 2 IX1	6 TST 1 IX		2 NOP 1 INH	8 BSR 2 REL	2 JSR 2 DIR	8 JSR 3 EXT	9 JSR 3 IX2	8 JSR 2 IX1	7 JSR 1 IX	D 1101
E 1110	10 BRSET7 3 BTB	BSET7 2 BSC	4 BIL 2 REL								2 LDX 2 IMM	4 LDX 2 DIR	5 LDX 3 EXT	6 LDX 3 IX2	5 LDX 2 IX1	4 LDX 1 IX	E 1110
F 1111	10 BRCLR7 3 BTB	BCLR7 2 BSC	4 BIH 2 REL	6 CLR 2 DIR	4 CLRA 1 INH	4 CLRX 1 INH	7 CLR 2 IX1	6 CLR 1 IX		2 TXA 1 INH		5 STX 2 DIR	6 STX 3 EXT	7 STX 3 IX2	6 STX 2 IX1	5 STX 1 IX	F 1111

Abbreviations for Address Modes

INH	Inherent
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



functions, these instructions cannot be used to set or clear a DDR bit (all "unaffected" bits would be set). It is recommended that all DDR bits in a port be written using a single-store instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit that is to be tested and its condition (set or clear) is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage Self-Check Mode (TIMER Pin Only)	V_{in}	−0.3 to +15.0	V
Operating Temperature Range MC6805R2 MC6805R2C MC6805R2V	T_A	T_L to T_H 0 to +70 −40 to 85 −40 to 105	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature Plastic PLCC Cerdip	T_J	 50 150 175	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation it is recommended the V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ and $V_{out}) \leq V_{CC}$. Reliability of operation is enhanced if unused inputs, except EXTAL, are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic (P Suffix) PLCC (FN Suffix) Cerdip (S Suffix)	θ_{JA}	 60 100 60	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

ELECTRICAL CHARACTERISTICS (V_{CC} = +5.25 Vdc ± 0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage	V_{IH}				V
$\overline{RESET}$ ($4.75 \leq V_{CC} \leq 5.75$)		4.0	—	V_{CC}	
V_{CC}(4.75)		V_{CC} – 0.5	—	V_{CC}	
$\overline{INT}$ ($4.75 \leq V_{CC} \leq 5.75$)		4.0	*	V_{CC}	
(V_{CC}(4.75)		V_{CC} – 0.5	*	V_{CC}	
All Other		2.0	—	V_{CC}	
Input High Voltage Timer	V_{IH}				V
Timer Mode		2.0	—	V_{CC} + 1.0	
Self-Check Mode		9.0	10.0	15.0	
Input Low Voltage	V_{IL}				V
$\overline{RESET}$		V_{SS}	—	0.8	
$\overline{INT}$		V_{SS}	*	1.5	
All Other (Except A/D Inputs)		V_{SS}	—	0.8	
$\overline{RESET}$ Hysteresis Voltages					V
"Out of Reset"	V_{IRES+}	2.1	—	4.0	
"Into Reset"	V_{IRES-}	0.8	—	2.0	
$\overline{INT}$ Zero-Crossing Input Voltage, Through a Capacitor	V_{INT}	2	—	4	$V_{ac\ p\text{-}p}$
Power Dissipation — (No Port Loading, V_{CC} = 5.75 V, T_A = 0°C	P_D	—	520	740	mW
for Steady-State Operation) T_A = –40°C		—	580	800	
Input Capacitance	C_{in}				pF
XTAL		—	25	—	
All Other Except Analog Inputs (See Note)		—	10	—	
Low Voltage Recover	V_{LVR}	—	—	4.75	V
Low Voltage Inhibit	V_{LVI}	2.75	3.75	4.70	V
Input Current					μA
$\overline{TIMER}$ (V_{in} = 0.4)		—	—	20	
$\overline{INT}$ (V_{in} = 2.4 V to V_{CC})		—	20	50	
EXTAL (V_{in} = 2.4 V to V_{CC} Crystal Option)	I_{in}	—	—	10	
(V_{in} = 0.4 V Crystal Option)		—	—	–1600	
$\overline{RESET}$ (V_{in} = 0.8 V)	I_{RES}	–4.0	—	–40	
(External Capacitor Charging Current)					

NOTE: Port D Analog Inputs, when selected C_{in} = 25 pF for the first 5 out of 30 cycles.
*Due to internal biasing this input (when unused) floats to approximately 2.0 V.

SWITCHING CHARACTERISTICS (V_{CC} = +5.25 Vdc ± 0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Characteristic	Symbol	Min	Typ	Max	Unit
Oscillator Frequency	f_{osc}	0.4	—	4.2	MHz
Cycle Time ($4/f_{osc}$)	t_{cyc}	0.95	—	10	μs
$\overline{INT}$, $\overline{INT2}$, and TIMER Pulse Width	t_{WL}, t_{WH}	t_{cyc} + 250	—	—	ns
$\overline{RESET}$ Pulse Width	t_{RWL}	t_{cyc} + 250	—	—	ns
$\overline{INT}$ Zero-Crossing Detection Input Frequency	f_{INT}	0.03	—	1	kHz
External Clock Input Duty Cycle (EXTAL)	—	40	50	60	%
Crystal Oscillator Start-Up Time	—	—	—	100	ms

A/D CONVERTER CHARACTERISTICS (V_{CC} = +5.25 Vdc ± 0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H), unless otherwise noted)

	Min	Typ	Max	Unit	Comments
Resolution	8	8	8	Bits	
Total Error	—	—	+/− 2.25*	LSB	Difference between ideal and actual transfer characteristics (includes non-linearity, zero offset and full scale errors)
Absolute Accuracy	—	—	+/− 2.75*	LSB	Difference between the actual input voltage and the full-scale weighted equivalent of the binary output code. All error sources included
Quantizing Error	—	—	+/− .5	LSB	Uncertainty due to converter resolution (inherent)
Conversion Range	V_{RL}	—	VRH	V	
V_{RH}	—	—	V_{CC}	V	A/D accuracy may decrease proportionately as VRH is reduced below 4.75 V. The sum of V_{RH} and V_{RL} must not exceed V_{CC}
V_{RL}	V_{SS}	—	0.2	V	
Conversion Time	30	30	30	t_{cyc}	Includes sample time
Monotonicity					Inherent with total error
Sample Time	5	5	5	t_{cyc}	
Sample/Hold Capacitance, Input	—	—	25	pF	
Analog Input Voltage	V_{RL}	—	V_{RH}	V	Negative transients on any analog lines (Pins 19-24) are not allowed at any time during conversion.

*Note: Accuracy may decrease at temperatures above T_A = 85°C or f_{osc} < 3.57 MHz.

3

PORT ELECTRICAL CHARACTERISTICS (V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Port A with CMOS Drive Enabled					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Output High Voltage, I_{Load} = −10 μA	V_{OH}	V_{CC} − 1.0	—	—	V
Input High Voltage, I_{Load} = −300 μA (max.)	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage, I_{Load} = −500 μA (max.)	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current (V_{in} = 2.0 V to V_{CC})	I_{IH}	—	—	−300	μA
Hi-Z State Input Current (V_{in} = 0.4 V)	I_{IL}	—	—	−500	μA
Port B					
Output Low Voltage, I_{Load} = 3.2 mA	V_{OL}	—	—	0.4	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.0	V
Output High Voltage, I_{Load} = −200 μA	V_{OH}	2.4	—	—	V
Darlington Current Drive (Source), V_O = 1.5 V	I_{OH}	−1.0	—	−10	mA
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port C and Port A with TTL Drive					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port C (Open-Drain Option)					
Input High Voltage	V_{IH}	2.0	—	13.0	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Input Leakage Current (V_{in} = 13.0 V)	I_{LOD}	—	<3	15	μA
Output Low Voltage I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Port D (Digital Inputs Only)					
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Input Current	I_{in}	—	<1	5	μA

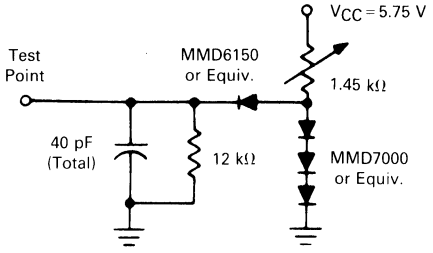


Figure 12. TTL Equivalent Test Load (Port B)

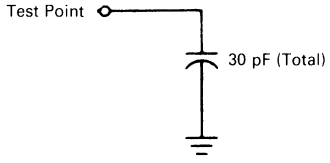


Figure 13. CMOS Equivalent Test Load (Port A)

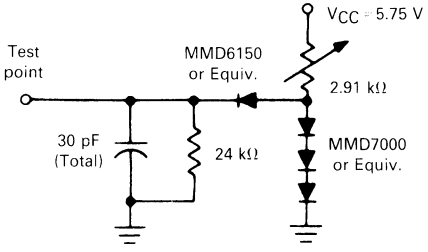


Figure 14. TTL Equivalent Test Load (Ports A and C)

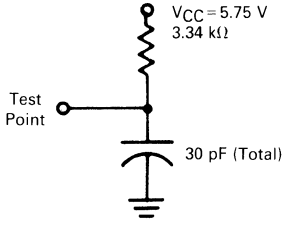


Figure 15. Open-Drain Equivalent Test Load (Port C)

3

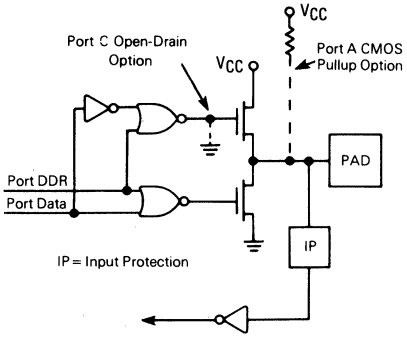


Figure 16. Ports A and C Logic Diagram

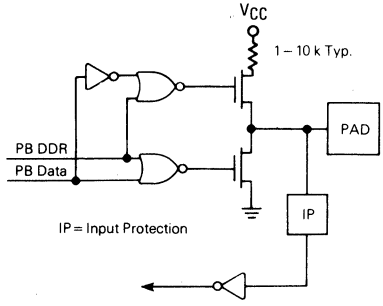


Figure 17. Port B Logic Diagram

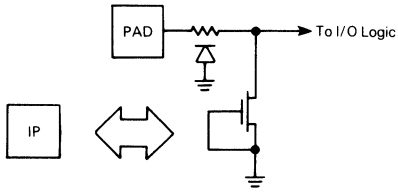


Figure 18. Typical Input Protection

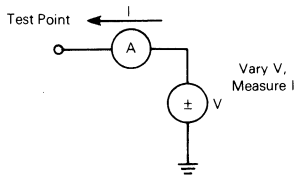


Figure 19. I/O Characteristic Measurement Circuit

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola in the following media:

MDOS™, disk file
MS™-DOS/PC-DOS disk file
EPROM(s) MC68705R3, 2532, 2732, or two 2516/2716

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, a sales person, or Motorola representative.

FLEXIBLE DISKS

Several types of flexible disks (MDOS or MS-DOS/PC-DOS disk file), programmed with the customer program (positive logic sense for address and data), may be submitted for pattern generation. In either case, the diskette should be clearly labeled with the customers name, date, project or product name, and the name of the file containing the pattern.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to speed up the process in case of any difficulty with the pattern file.

MDOS Disk File

MDOS is Motorola's Disk Operating System available on the EXORciser® development system. The disk media submitted must be a single-side, single-density, 8-inch MDOS compatible floppy diskette. The diskette must contain the minimum set of MDOS system files in addition to the pattern file.

The .LO output of the M6805 cross assembler should be furnished. In addition, the file must be produced (using the ROLLOUT command) containing the absolute image of the M6805 memory. Include the entire memory image of both data and program space. All unused bytes, including those in the user space, must be set to zero.

MS-DOS/PC-DOS Disk File

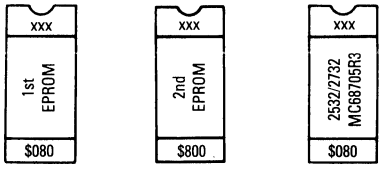
MS-DOS is Microsoft's Disk Operating System. PC-DOS is IBM® Personal Computer (PC) Disk Operating System. Disk media submitted must be a standard density (360K) double sided 5 1/4 inch compatible floppy diskette. The diskette must contain object file code in Motorola's S-record format. The S-record format is a character-based object file format generated by M6805 cross assemblers and linkers on IBM PC style machines.

EPROMs

An MC68705R3, 2532, 2732, 2516 (2), or 2716 (2) type EPROM(s), programmed with the customer program (positive logic sense for address and data) may be submitted for pattern generation. Since all program and data space information will fit on one MC68705R3/2532/2732 or two 2516/2716 type EPROM(s), the EPROM(s) must be programmed as described in the following paragraph.

For the 2532, 2732, or the MC68705R3, the ROM code should be located from $080 to $FF; and $700 to $F37 and the interrupt vectors from $FF8 to $FFF. For the 2516's or 2716's, the ROM code should be located from $080 to $FF and $7C0 to $7FF in the first EPROM and from $0 to $737 in the second EPROM. The interrupt vectors should be in the second EPROM from $7F8 to $7FF.

EPROM MARKING



xxx = Customer ID

VERIFICATION MEDIA

All original pattern media (EPROMs or floppy disk) are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked and the verification form completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for creation of the customer mask. To aid in the verification process, Motorola will program (customer supplied) blank EPROM(s) or DOS disk from the data file used to create the custom mask.

ROM VERIFICATION UNITS (RVUs)

Ten MCUs containing the customer's ROM pattern will be sent for program verification. These units will have been made using the custom mask but are for the purpose of ROM verification only. For expediency the MCUs are unmarked, packaged in ceramic, and tested at room temperature and five volts. These RVUs are free with the minimum order quantity but are not production parts. These RVUs are not guaranteed by Motorola Quality Assurance.

ORDERING INFORMATION

The following table provides generic information pertaining to the package type, temperature, and MC part numbers for the MC6805R2.

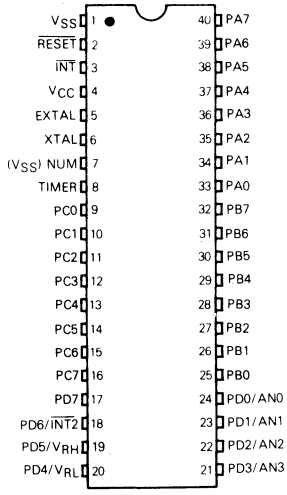
Package Type	Temperature	Part Number
Plastic (P Suffix)	0°C to 70°C −40°C to +85°C	MC6805R2P MC6805R2CP
Cerdip S Suffix	0°C to 70°C −40°C to +85°C	MC6805R2S MC6805R2CS
PLCC FN Suffix	0°C to 70°C −40°C to +85°C	MC6805R2FN MC6805R2CFN

MDOS is a trademark of Motorola Inc.
MS is a trademark of Microsoft, Inc.
EXORciser is a registered trademark of Motorola Inc.
IBM is a registered trademark of International Business Machines Corporation.

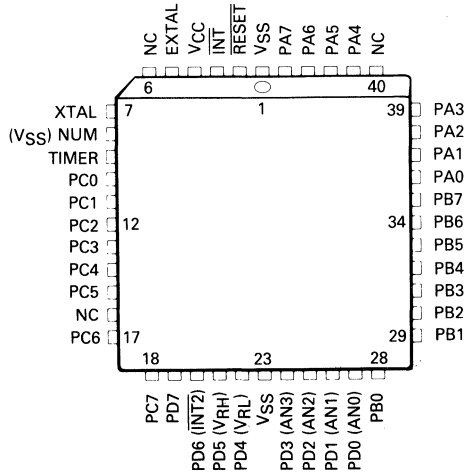
MECHANICAL DATA

PIN ASSIGNMENTS

Dual-in-Line Package



PLCC Package



MC6805R3

Technical Summary
8-Bit Microcontroller Unit

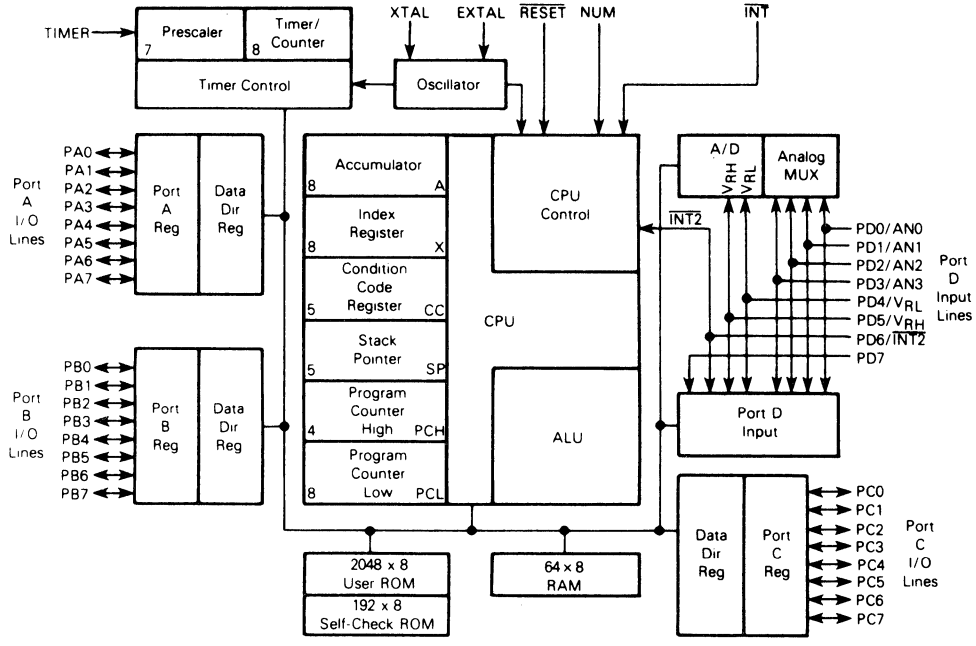
The MC6805R3 (HMOS) Microcontroller Unit (MCU) is a member of the MC6805 Family of microcomputers. This low cost and high-speed MCU has parallel I/O capability with pins programmable as input or output. This publication contains condensed information on the MCU; for detailed information, refer to *M6805 HMOS, M146805 CMOS Family User's Manual* (M6805UM(AD2)) or contact your local Motorola sales office.

Refer to the block diagram for the hardware features and to the below list for additional features available on the MCU.

- Internal 8-Bit Timer with 7-Bit Prescaler
- On-chip Oscillator
- Memory Mapped I/O
- Versatile Interrupt Handling
- True Bit Manipulation
- Bit Test and Branch Instruction
- Vectored Interrupts
- Self-Check Mode
- 3776 Bytes of ROM
- 112 Bytes of RAM
- 24 Bidirectional I/O Ports
- A/D Converter

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

V_{CC} AND V_{SS}

Power is supplied to the microcomputer using these two pins. V_{CC} is +5.25 volts (±0.5Δ) power, and V_{SS} is ground.

$\overline{INT}$

This pin provides the capability for asynchronously applying an external interrupt to the MCU. Refer to **INTERRUPTS** for more detailed information.

EXTAL, XTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a resistor/capacitor combination, or an external signal (depending upon selected manufacturing mask option) is connected to these pins to provide a system clock.

RC Oscillator

With this option, a resistor is connected to the oscillator pins as shown in Figure 1. The relationship between R and f_{osc} is shown in Figure 2.

Crystal

The circuit shown in Figure 1 is recommended when using a crystal. Using an external CMOS oscillator is recommended when crystals outside the specified ranges are to be used. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time. Refer to **ELECTRICAL SPECIFICATIONS** for V_{CC} specifications.

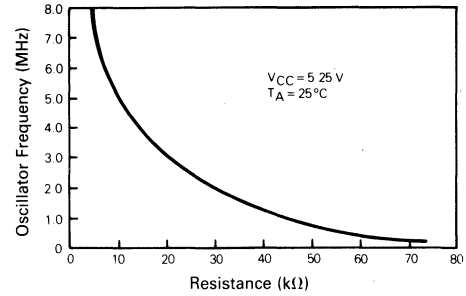
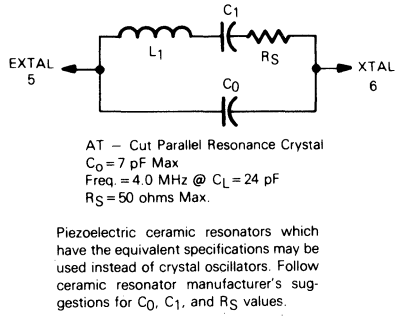
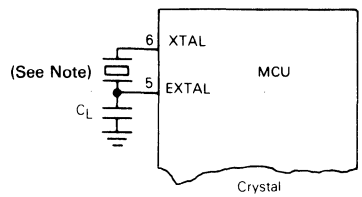


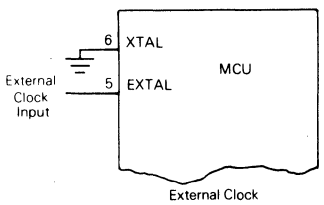
Figure 2. Typical Frequency vs Resistance for RC Oscillator Option Only

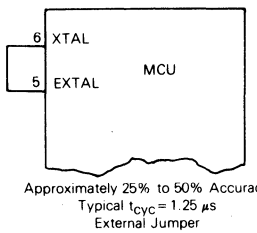
External Clock

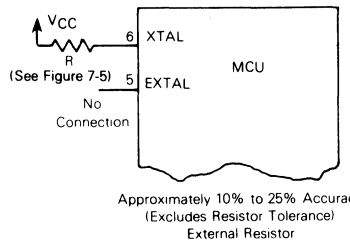
An external clock should be applied to the EXTAL input with the XTAL input connected to ground, as shown in











NOTE: The recommended C_L value with a 4.0 MHz crystal is 27 pF maximum, including system distributed capacitance. There is an internal capacitance of approximately 25 pF on the XTAL pin. For crystal frequencies other than 4 MHz, the total capacitance on each pin should be scaled as the inverse of the frequency ratio. For example, with a 2 MHz crystal, use approximately 50 pF on EXTAL and approximately 25 pF on XTAL. The exact value depends on the Motional-Arm parameters of the crystal used.

Figure 1. Oscillator Connections

Figure 1. This option may only be used with the crystal oscillator option selected in the mask option register. The t_{OXOV} or t_{ILCH} specifications do not apply when using an external clock input.

TIMER

This pin is used as an external input to control the internal timer/counter circuitry. This pin also detects a higher voltage level used to initiate the self-test program.

$\overline{\text{RESET}}$

This pin has a Schmitt trigger input and an on-chip pullup. The MCU can be reset by pulling $\overline{\text{RESET}}$ low.

INPUT/OUTPUT LINES (PA0-PA7, PB0-PB7, PC0-PC3)

These 32 lines are arranged into four 8-bit ports (A, B, C, and D). Ports A, B, and C are programmable as either inputs or outputs under software control of the data direction registers. Port D is a fixed input port and not controlled by any data register. Port D has up to four analog inputs, plus two voltage references inputs when the A/D converter is used (PD5/V_{RH}, PD4/V_{RL}), and an $\overline{\text{INT2}}$ input. All Port D lines can be read directly and used as binary input. If any analog input is used, then V_{RH} and V_{RL} must be used in the analog mode. Refer to **PROGRAMMING** and **ANALOG-TO-DIGITAL CONVERTER** for additional information.

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

Ports A, B, and C are programmable as either input or output under software control of the corresponding data direction register (DDR). Port D lines are input only. The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic one for output and a logic zero for input. On reset, all the DDRs are initialized to a logic zero state to put the ports in the input mode. The port output registers are not initialized on reset and should be written to before setting the DDR bits.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This port write may be used to initialize the data registers and avoid undefined outputs. Care must be exercised when using read-modify-write instructions since the data read corresponds to the pin level if the DDR is an input (zero) and, also, to the latched output when the DDR is an output (one). Refer to Table 1 for I/O functions and to Figure 3 for typical port circuitry.

Port D provides reference voltage ($\overline{\text{INT2}}$) and multiplexed analog inputs. Port D can always be used as digital input and may be used for analog if V_{RH} and V_{RL} are connected to the appropriate reference voltage. The V_{RH} (PD5) and V_{RL} (PD4) are internally connected to the A/D resistor.

Table 1. I/O Pin Functions

Data Direction Register Bit	Latched Output Data Bit	Output State	Input To MCU
1	0	0	0
1	1	1	1
0	X	Hi-Z**	Pin

**Ports B and C are three state ports. Port A has optional internal pullup devices to provide CMOS data drive capability.

MEMORY

The MCU is capable of addressing 4096 bytes of memory and I/O registers. The memory map is shown in Figure 4. The locations consist of user ROM, self-check ROM, user RAM, A/D registers, a miscellaneous register, and I/O. The interrupt and reset vectors are located from $FF8 to $FFF.

The stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer

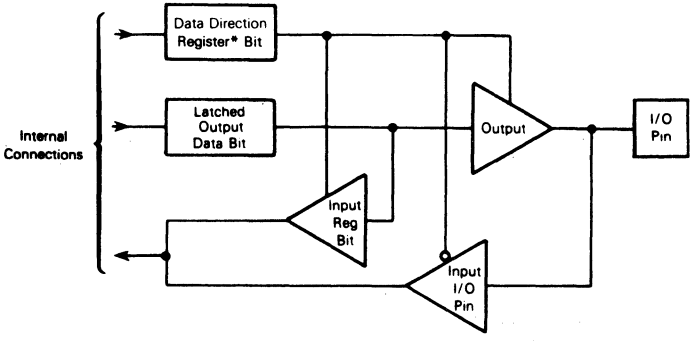
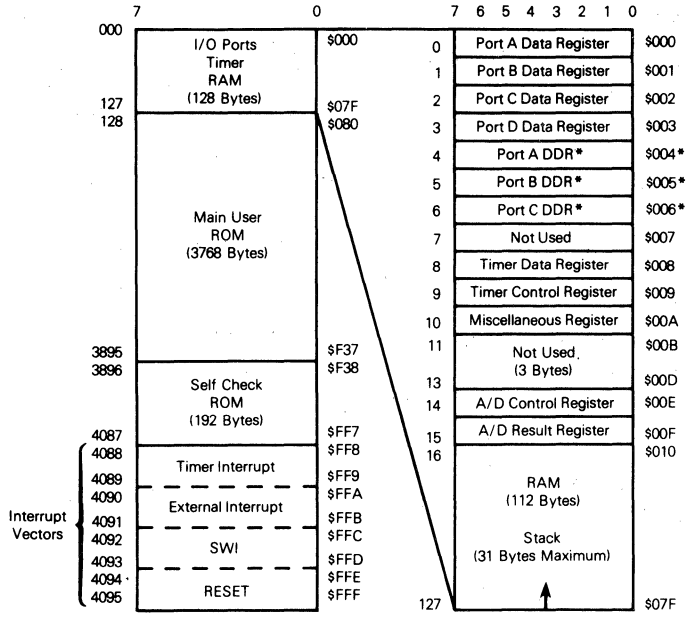


Figure 3. Typical Port I/O Circuitry and Register Configuration



*Caution: Data direction registers (DDRs) are write-only; they read as $FF.

Figure 4. Memory Map

decrements during pushes and increments during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

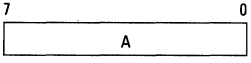
Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

REGISTERS

The MCU contains the registers described in the following paragraphs.

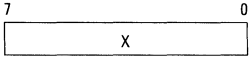
ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



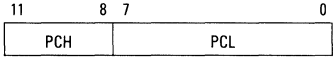
INDEX REGISTER (X)

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16-bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



PROGRAM COUNTER (PC)

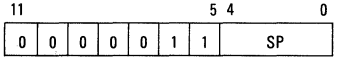
The program counter is a 12-bit register that contains the address of the next byte to be fetched.



STACK POINTER (SP)

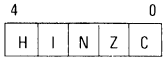
The stack pointer is a 12-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set at location $07F. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

The seven most-significant bits of the stack pointer are permanently set at 0000011. Subroutines and interrupts may be nested down to location $061 (31 bytes maximum), which allows the programmer to use up to 15 levels of subroutine calls (less if interrupts are allowed).



CONDITION CODE REGISTER (CC)

The condition code register is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program, and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.



Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, the timer and external interrupt is masked (disabled). If an external interrupt occurs while this bit is set, the interrupt is latched and is processed as soon as the interrupt bit is cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic 1).

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/Borrow (C)

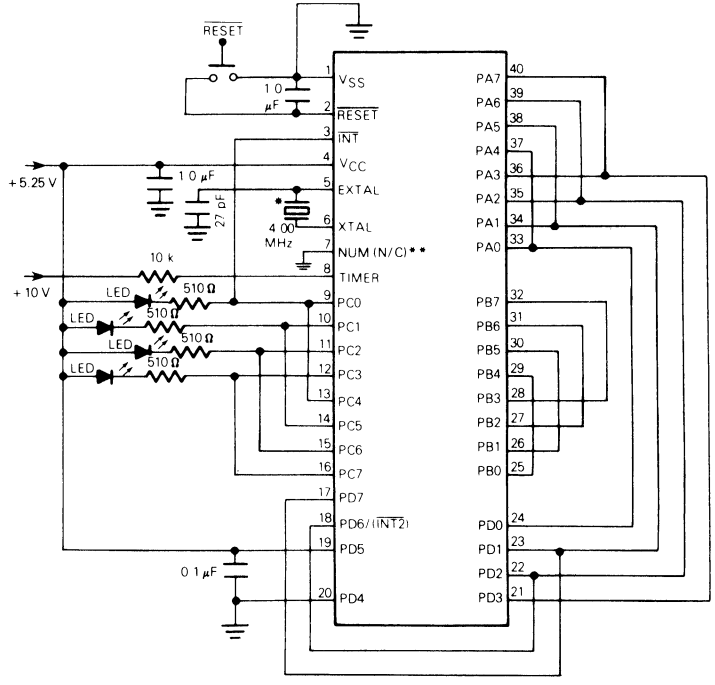
When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions, and during shifts and rotates.

SELF CHECK

The self check is initiated by connecting the MCU as shown in Figure 5 and then monitoring the output of port C (bit 3) for an oscillation of approximately 7 Hz. The following test are executed automatically:

I/O — Functionally exercise I/O ports,
RAM — Walking bit test,
ROM — Exclusive OR with ODD "1st" parity result,
Timer — Functionally exercise timer,
Interrupts — Functionally exercise external and timer interrupts, and
A/D Converter — Functionally test the Analog-to-Digital Converter.

The RAM, ROM, and the A/D test can be called by a user program. The Timer test may be called if the timer input is the internal clock. Table 2 shows the status of the LEDs as a result of a failure. Port C is tested only once (just after reset). If port C fails, no lights will appear.



*This connection depends on clock oscillator user selectable mask option. Use jumper if the RC mask option is selected.

Figure 5. Self-Check Connections

Table 2. Self-Check Error Patterns

LED Meanings				
PC0	**PC1**	**PC2**	**PC3**	**Remarks (1: LED ON; 0: LED OFF)**
1	0	1	0	Bad I/O
0	0	1	0	Bad Timer
1	1	0	0	Bad RAM
0	1	0	0	Bad ROM
1	0	0	0	Bad A/D
0	0	0	0	Bad Interrupts or Request Flag
All Flashing				Good Device

Anything else Bad Part, Bad Port C, etc.

RESETS

The MCU can be reset three ways: (1) by initial power-up (2) by the external reset input ($\overline{\text{RESET}}$) and (3) by an optional, internal, low-voltage detect circuit. The $\overline{\text{RESET}}$ input consists mainly of a Schmitt trigger that senses the line logic level.

3

POWER-ON-RESET (POR)

An internal reset is generated on power-up that allows the internal clock generator to stabilize. The power-on reset is used strictly for power turn-on conditions and should not be used to detect any drop in the power supply voltage. A delay of t_{RHL} milliseconds is required before allowing the $\overline{\text{RESET}}$ input to go high. Connecting a capacitor to the $\overline{\text{RESET}}$ input (Figure 6) typically provides sufficient delay.

EXTERNAL RESET INPUT

The MCU is reset when a logic zero is applied to the $\overline{\text{RESET}}$ input for a period longer than one machine cycle (t_{cyc}). Under this type of reset, the Schmitt trigger switches off at V_{IRES-} to provide an internal reset voltage.

LOW-VOLTAGE INHIBIT (LVI)

The optional low-voltage detection circuit causes a reset of the MCU if the power supply voltage falls below a certain level (V_{LVI}). The only requirement is that the V_{CC} must remain at or below the V_{LVI} threshold for one t_{cyc} minimum.

In typical applications, the V_{CC} bus filter capacitor will eliminate negative-going voltage glitches of less than one t_{cyc}. The output from the low-voltage detector is connected directly to the internal reset circuitry. It also forces the $\overline{\text{RESET}}$ pin low via a strong discharge device through a resistor. The internal reset is removed once the power supply voltage rises above a recovery level (V_{LVR}) at which time a normal power-on reset occurs.

INTERRUPTS

The MCU can be interrupted four different ways: (1) through the external interrupt $\overline{\text{IRQ}}$ input pin, (2) with the internal timer interrupt request, (3) using the software interrupt instruction (SWI), or (4) the external port D bit 6 ($\overline{\text{INT2}}$) input pin.

Interrupts cause the processor registers to be saved on the stack and the interrupt mask (I bit) set to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack and then normal processing resumes. The stacking order is shown in Figure 7.

Unlike $\overline{\text{RESET}}$, hardware interrupts do not cause the current instruction execution to be halted but are considered pending until the current instruction is complete.

NOTE

The current instruction is considered to be the one already fetched and being operated on.

When the current instruction is complete, the processor checks all pending hardware interrupts and, if unmasked (I bit clear), proceeds with interrupt processing; otherwise, the next instruction is fetched and executed. Masked interrupts are latched for later interrupt service. If the timer interrupt status bit is cleared before unmasking the interrupt, then the interrupt is not latched.

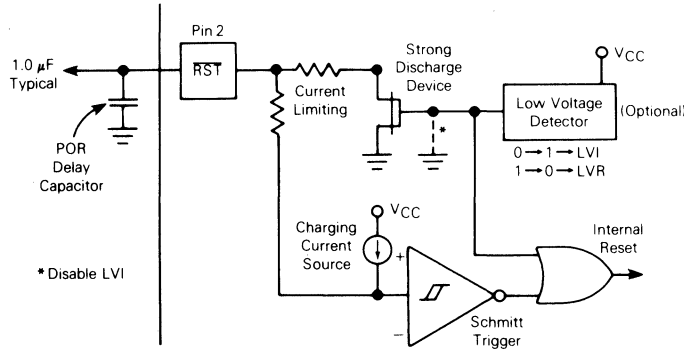
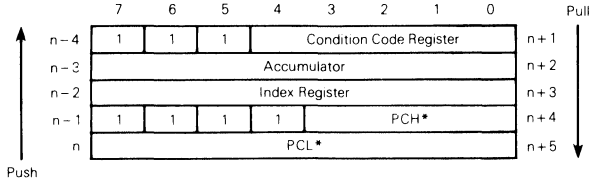


Figure 6. RESET Configuration



*For subroutine calls, only PCH and PCL are stacked.

Figure 7. Interrupt Stacking Order

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction regardless of the setting of the I bit. Refer to Figure 8 for the reset and interrupt instruction processing sequence.

TIMER INTERRUPT

If the timer mask bit (TCR6) is cleared, then, each time the timer decrements to zero (transitions from $01 to $00), an interrupt request is generated. The actual processor interrupt is generated only if the interrupt mask bit of the condition code register (CCR) is also cleared. When the interrupt is recognized, the current state of the machine is pushed onto the stack and the I bit in the CCR is set, masking further interrupts until the present one is serviced. The contents of the timer interrupt vector, containing the location of the timer interrupt service routine, is then loaded into the program counter. At the end of the timer interrupt service routine, the software normally executes an RTI instruction which restores the machine state and starts executing the interrupted program.

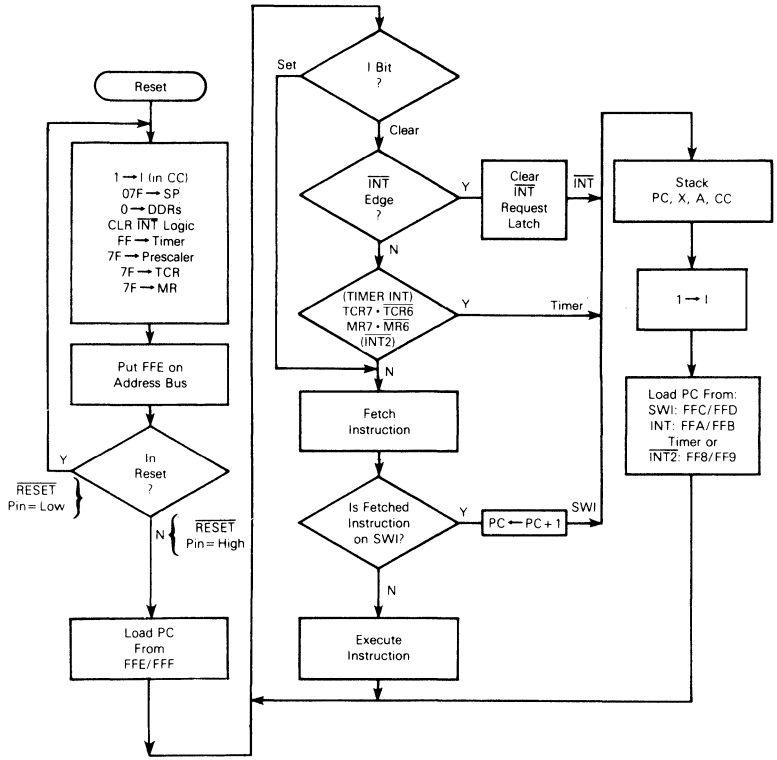


Figure 8. Reset and Interrupt Processing Flowchart

EXTERNAL INTERRUPT

The external interrupt is internally synchronized and then latched on the falling edge of $\overline{INT}$ and $\overline{INT2}$. Clearing the I bit enables the external interrupt. The $\overline{INT2}$ interrupt has an interrupt request bit (bit 7) and a mask bit (bit 6) in the miscellaneous register (MR). The $\overline{INT2}$ interrupt is inhibited when the mask bit is set. The $\overline{INT2}$ is always read as a digital input on port D. The $\overline{INT2}$ and timer interrupt request bits, if set, cause the MCU to process and interrupt when the condition code I bit is clear. The following paragraphs describe two typical external interrupt circuits.

Zero-Crossing Interrupt

A sinusoidal input signal (f_{INT} maximum) can be used to generate an external interrupt (see Figure 9a) for use as a zero-crossing detector (for negative transitions of the ac sinusoid). This type of circuit allows applications such as servicing time-of-day routines and engaging/disengaging ac power control devices. Off-chip, full-wave rectification provides an interrupt at every zero crossing of the ac signal and, thereby, provides a 2f clock.

Digital-Signal Interrupt

With this type of circuit (Figure 9b), the $\overline{INT}$ pin can be driven by a digital signal. The maximum frequency of a signal that can be recognized by the TIMER or $\overline{INT}$ pin logic is dependent on the parameter labeled t_{WL}, t_{WH}. Refer to **TIMER** for additional information.

SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. If the I bit is zero, SWI executes after the other interrupts. The SWI execution is similar to the hardware interrupts.

TIMER

The MCU consists of an 8-bit software programmable counter driven by a 7-bit software programmable prescaler. The various timer sources are made via the timer control register (TCR). The 8-bit counter may be loaded under program control and is decremented toward zero. When the timer reaches zero, the timer interrupt request bit (bit 7) in the timer control register (TCR) is set. Refer to Figure 10 for timer block diagram.

The timer interrupt can be masked (disabled) by setting the timer interrupt mask bit (bit 6) in the TCR. When the I bit in the condition code register is cleared and TCR bit 6 is cleared, the processor receives the interrupt. The MCU responds to this interrupt by 1) saving the present

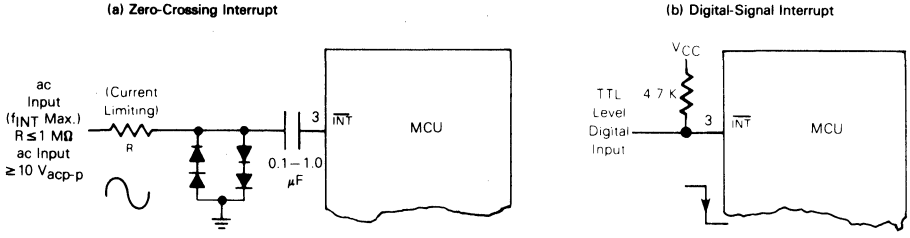
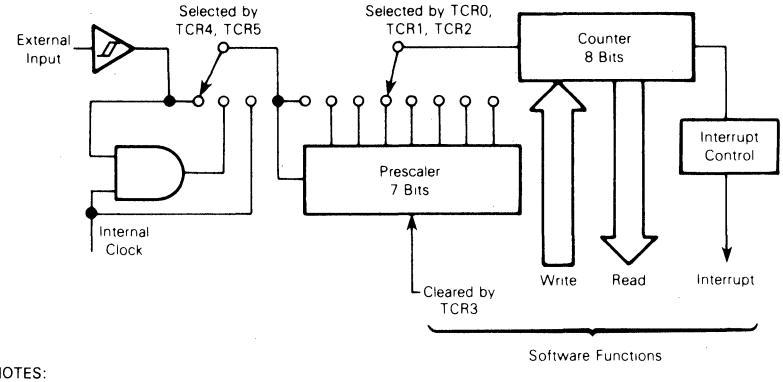


Figure 9. Typical Interrupt Circuits



NOTES:
1. The prescaler and 8-bit counter are clocked on the rising edge of the internal clock (phase two) or external input.
2. The counter is written to during data strobe (DS) and counts down continuously.

Figure 10. Timer Block Diagram

CPU state on the stack, 2) fetching the timer interrupt vector, and 3) executing the interrupt routine. The timer interrupt request bit must be cleared by software. Refer to **RESETS** and **INTERRUPTS** for additional information.

The prescaler is a 7-bit divider which is used to extend the maximum length of the timer. To avoid truncation errors, the prescaler is cleared when TCR bit 3 is set to a logic one; however, the TCR bit 3 always reads as a logic zero to ensure proper operation with read-modify-write instructions.

The timer continues to count past zero, falling from $00 through $FF, and continues the countdown. The counter can be read at any time by reading the timer data register (TDR). This allows a program to determine the length of time since a timer interrupt has occurred without disturbing the counting process. TDR is unaffected by reset.

SOFTWARE CONTROLLED MODE

The timer prescaler input can be configured for three different operating modes plus a disable mode, depending on the value written to TCR control bits 4 and 5 (TCR4 and TCR5). The following paragraphs describe the different modes.

Timer Input Mode 1

When TCR4 and TCR5 are both programmed to zero, the timer input is from the internal clock (phase two) and the timer input pin is disabled. The internal clock mode can be used for periodic interrupt generation as well as a reference for frequency and event measurement. During the WAIT instruction, the internal clock to the timer continues to run at its normal rate.

Timer Input Mode 2

When TCR4 = 1 and TCR5 = 0, the internal clock and the timer input signals are ANDed to form the timer input. This mode can be used to measure external pulse widths. The active high, external pulse gates in the internal clock for the duration of the external pulse. The accuracy of the count is ±1.

Timer Input Mode 3

When TCR4 = 0 and TCR5 = 1, no prescaler input frequency is applied to the prescaler and the timer is disabled.

Timer Input Mode 4

When TCR4 and TCR5 are both one, the timer input is from the external clock. The external clock can be used to count external events as well as to provide an external frequency for generating periodic interrupts.

TIMER CONTROL REGISTER (TCR) $009

This is an 8-bit register that controls various functions such as configuring operation mode, setting ratio of the prescaler, and generating timer interrupt request signal. Bit 3 is a write only bit.

7	6	5	4	3	2	1	0
TCR7	TCR6	TCR5	TCR4	TCR3	TCR2	TCR1	TCR0
RESET:							
0	1	U	U	U	U	U	U

TCR7 — Timer Interrupt Request
Used to indicate the timer interrupt when it is logic one
1 = Set when the timer data register changes to all zeros
0 = Cleared by external reset, power-on reset, or under program control

TCR6 — Timer Interrupt Mask
Used to inhibit the timer interrupt
1 = Interrupt inhibited
0 = Interrupt enabled

TCR5 — External or Internal
Selects input clock source
1 = External clock selected
0 = Internal clock selected ($f_{OSC}/4$)

TCR4 — TIMER External Enable
Used to enable external TIMER pin
1 = Enables external timer pin
0 = Disables external timer pin

TCR3 — Prescaler Clear
Write only bit. Writing a one to this bit resets the prescaler to zero. A read of this location always indicates a zero.

TCR2, TCR1, TCR0 — Prescaler Select Bits
Decoded to select one of eight outputs of the prescaler

Prescaler

TCR2	TCR1	TCR0	Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

ANALOG-TO-DIGITAL CONVERTER

The chip resident 8-bit analog-to-digital (A/D) converter uses a successive approximation technique as shown in Figure 11. Four external analog inputs can be connected to the A/D through a multiplexer via port D. Four internal analog channels ($V_{RH} - V_{RL}$, $V_{RH} - V_{RL}/2$, $V_{RH} - V_{RL}/4$, and V_{RL}) may be selected for calibration. The accuracy of these internal channels may not meet the accuracy specifications of the external channels.

Multiplexer selection is controlled by the A/D control register (ACR) bits 0, 1, and 2. Refer to Table 3 for multiplexer selection. The ACR is shown in Figure 11. The converter uses 30 machine cycles to complete a conversion of a sampled analog input. When the conversion is complete, the digital value is placed in the A/D result register (ARR); the conversion is flag set; selected input is sampled again; and a new conversion begins. When ACR7 is cleared, the conversion in progress is aborted and the selected input is sampled for five machine cycles and held internally.

Table 3. A/D Input MUX Selection

A/D Control Register			Input elected	A/D Output (Hex)		
ACR2	ACR1	ACR0		Min	Typ	Max
0	0	0	AN0			
0	0	1	AN1			
0	1	0	AN2			
0	1	1	AN3			
1	0	0	V_{RH}*	FE	FF	FF
1	0	1	V_{RL}*	00	00	01
1	1	0	$V_{RH}/4$*	3F	40	41
1	1	1	$V_{RH}/2$*	7F	80	81

*Internal (calibration) levels

The converter uses V_{RH} and V_{RL} as reference voltages. An input voltage equal to or greater than V_{RH} converts to $FF. An input voltage equal to or less than V_{RL}, but greater than V_{SS}, converts to $00. Maximum and minimum ratings must not be exceeded. Each analog input source should use V_{RH} as the supply voltage and be referenced to V_{RL} for the ratiometric conversion. To maintain full accuracy of the A/D, three requirements should be followed: (1) V_{RH} should be equal to or less than V_{DD}, (2) V_{RL} should be equal to or greater than V_{SS} but less than maximum specifications, and (3) $V_{RH}-V_{RL}$ should be equal to or greater than 4 volts.

The A/D has a built-in 1/2 LSB offset intended to reduce the magnitude of the quantizing error to ± 1/2 LSB, rather than +0, −1 LSB with no offset. This implies that, ignoring errors, the transition point from $00 to $01 occurs at 1/2 LSB above V_{RL}. Similarly, the transition from $FE to $FF occurs 1-1/2 LSB below V_{RH}, ideally.

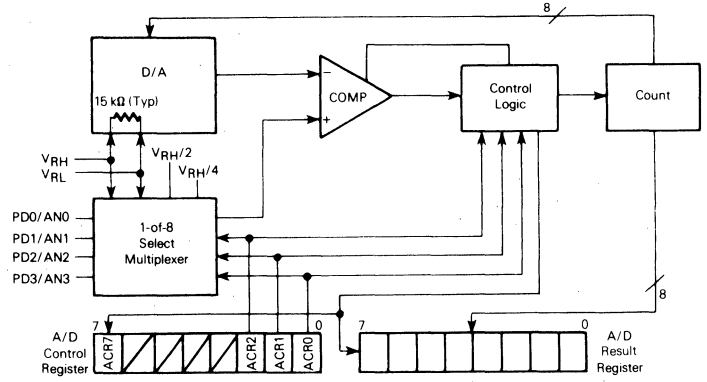


Figure 11. A/D Block Diagram

INSTRUCTION SET

The MCU has a set of 59 basic instructions which can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following list of instructions.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the

read-modify-write sequence since it does not modify the value. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (2's Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Higher	BHI
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
(Branch if Higher or Same)	(BHS)
Branch if Carry Set	BCS
(Branch if Lower)	(BLO)
Branch if Not Equal	BNE
Branch if Equal	BEQ
Branch if Half Carry Clear	BHCC
Branch if Half Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Bit is Clear	BMC
Branch if Interrupt Mask Bit is Set	BMS
Branch if Interrupt Line is Low	BIL
Branch if Interrupt Line is High	BIH
Branch to Subroutine	BSR

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the first 256 bytes of the memory space, where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear and bit test, and branch functions are all implemented with a single instruction. For test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch if Bit n is Set	BRSET n (n=0 . . . 7)
Branch if Bit n is Clear	BRCLR n (n=0 . . . 7)
Set Bit n	BSET n (n=0 . . . 7)
Clear Bit n	BCLR n (n=0 . . . 7)

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No-Operation	NOP

OPCODE MAP SUMMARY

Table 4 is an opcode map for the instructions used on the MCU.

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code coversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single byte instructions, while the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. Two-byte direct addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

Table 4. Opcode Map

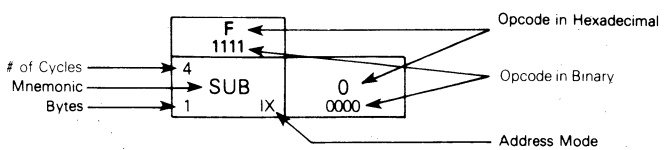
	Bit Manipulation		Branch	Read-Modify-Write					Control		Register/Memory						
	BTB 0 0000	BSC 1 0001	REL 2 0010	DIR 3 0011	INH 4 0100	INH 5 0101	IX1 6 0110	IX 7 0111	INH 8 1000	INH 9 1001	IMM A 1010	DIR B 1011	EXT C 1100	IX2 D 1101	IX1 E 1110	IX F 1111	
0 0000	10 BRSET0 3 BTB	7 BSET0 2 BSC	4 BRA 2 REL	6 NEG 2 DIR	4 NEG 1 INH	4 NEG 1 INH	7 NEG 2 IX1	6 NEG 1 IX	9 RTI 1 INH		2 SUB 2 IMM	4 SUB 2 DIR	5 SUB 3 EXT	6 SUB 3 IX2	5 SUB 2 IX1	4 SUB 1 IX	0 0000
1 0001	10 BRCLR0 3 BTB	7 BCLR0 2 BSC	4 BRN 2 REL						6 RTS 1 INH		2 CMP 2 IMM	4 CMP 2 DIR	5 CMP 3 EXT	6 CMP 3 IX2	5 CMP 2 IX1	4 CMP 1 IX	1 0001
2 0010	10 BRSET1 3 BTB	7 BSET1 2 BSC	4 BHI 2 REL								2 SBC 2 IMM	4 SBC 2 DIR	5 SBC 3 EXT	6 SBC 3 IX2	5 SBC 2 IX1	4 SBC 1 IX	2 0010
3 0011	10 BRCLR1 3 BTB	7 BCLR1 2 BSC	4 BLS 2 REL	6 COM 2 DIR	4 COMA 1 INH	4 COMX 1 INH	7 COM 2 IX1	6 COM 1 IX	11 SWI 1 INH		2 CPX 2 IMM	4 CPX 2 DIR	5 CPX 3 EXT	6 CPX 3 IX2	5 CPX 2 IX1	4 CPX 1 IX	3 0011
4 0100	10 BRSET2 3 BTB	7 BSET2 2 BSC	4 BCC 2 REL	6 LSR 2 DTR	4 LSRA 1 INH	4 LSRX 1 INH	7 LSR 2 IX1	6 LSR 1 IX			2 AND 2 IMM	4 AND 2 DIR	5 AND 3 EXT	6 AND 3 IX2	5 AND 2 IX1	4 AND 1 IX	4 0100
5 0101	10 BRCLR2 3 BTB	7 BCLR2 2 BSC	4 BCS 2 REL								2 BIT 2 IMM	4 BIT 2 DIR	5 BIT 3 EXT	6 BIT 3 IX2	5 BIT 2 IX1	4 BIT 1 IX	5 0101
6 0110	10 BRSET3 3 BTB	7 BSET3 2 BSC	4 BNE 2 REL	6 ROR 2 DIR	4 RORA 1 INH	4 RORX 1 INH	7 ROR 2 IX1	6 ROR 1 IX			2 LDA 2 IMM	4 LDA 2 DIR	5 LDA 3 EXT	6 LDA 3 IX2	5 LDA 2 IX1	4 LDA 1 IX	6 0110
7 0111	10 BRCLR3 3 BTB	7 BCLR3 2 BSC	4 BEQ 2 REL	6 ASR 2 DIR	4 ASRA 1 INH	4 ASRX 1 INH	7 ASR 2 IX1	6 ASR 1 IX		2 TAX 1 INH		5 STA 2 DIR	6 STA 3 EXT	7 STA 3 IX2	6 STA 2 IX1	5 STA 1 IX	7 0111
8 1000	10 BRSET4 3 BTB	7 BSET4 2 BSC	4 BHCC 2 REL	6 LSL 2 DIR	4 LSLA 1 INH	4 LSLX 1 INH	7 LSL 2 IX1	6 LSL 1 IX		2 CLC 1 INH	2 EOR 2 IMM	4 EOR 2 DIR	5 EOR 3 EXT	6 EOR 3 IX2	5 EOR 2 IX1	4 EOR 1 IX	8 1000
9 1001	10 BRCLR4 3 BTB	7 BCLR4 2 BSC	4 BHCS 2 REL	6 ROL 2 DIR	4 ROLA 1 INH	4 ROLX 1 INH	7 ROL 2 IX1	6 ROL 1 IX		2 SEC 1 INH	2 ADC 2 IMM	4 ADC 2 DIR	5 ADC 3 EXT	6 ADC 3 IX2	5 ADC 2 IX1	4 ADC 1 IX	9 1001
A 1010	10 BRSET5 3 BTB	7 BSET5 2 BSC	4 BPL 2 REL	6 DEC 2 DIR	4 DECA 1 INH	4 DECX 1 INH	7 DEC 2 IX1	6 DEC 1 IX		2 CLI 1 INH	2 ORA 2 IMM	4 ORA 2 DIR	5 ORA 3 EXT	6 ORA 3 IX2	5 ORA 2 IX1	4 ORA 1 IX	A 1010
B 1011	10 BRCLR5 3 BTB	7 BCLR5 2 BSC	4 BMI 2 REL							2 SEI 1 INH	2 ADD 2 IMM	4 ADD 2 DIR	5 ADD 3 EXT	6 ADD 3 IX2	5 ADD 2 IX1	4 ADD 1 IX	B 1011
C 1100	10 BRSET6 3 BTB	7 BSET6 2 BSC	4 BMC 2 REL	6 INC 2 DIR	4 INCA 1 INH	4 INCX 1 INH	7 INC 2 IX1	6 INC 1 IX		2 RSP 1 INH		3 JMP 2 DIR	4 JMP 3 EXT	5 JMP 3 IX2	4 JMP 2 IX1	3 JMP 1 IX	C 1100
D 1101	10 BRCLR6 3 BTB	7 BCLR6 2 BSC	4 BMS 2 REL	6 TST 2 DIR	4 TSTA 1 INH	4 TSTX 1 INH	7 TST 2 IX1	6 TST 1 IX		2 NOP 1 INH	8 BSR 2 REL	7 JSR 2 DIR	8 JSR 3 EXT	9 JSR 3 IX2	8 JSR 2 IX1	7 JSR 1 IX	D 1101
E 1110	10 BRSET7 3 BTB	7 BSET7 2 BSC	4 BIL 2 REL								2 LDX 2 IMM	4 LDX 2 DIR	5 LDX 3 EXT	6 LDX 3 IX2	5 LDX 2 IX1	4 LDX 1 IX	E 1110
F 1111	10 BRCLR7 3 BTB	7 BCLR7 2 BSC	4 BIH 2 REL	6 CLR 2 DIR	4 CLRA 1 INH	4 CLRX 1 INH	7 CLR 2 IX1	6 CLR 1 IX		2 TXA 1 INH		5 STX 2 DIR	6 STX 3 EXT	7 STX 3 IX2	6 STX 2 IX1	5 STX 1 IX	F 1111

Hi / Low

Abbreviations for Address Modes

INH	Inherent
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction.

RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −126 to +129 from the opcode address.

INDEX, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. This addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this two-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This addressing mode can be used in a manner similar to indexed, 8-bit offset except that this three-byte instruction allows tables to be anywhere in memory.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte to which the specified bit is to be set or cleared. Any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single two-byte instruction.

CAUTION

The corresponding DDRs for ports A, B, and C are write only registers (registers at $004, $005, and $006). A read operation on these registers is undefined. Since BSET and BCLR are read-modify-write functions, these instructions cannot be used to set or clear a DDR bit (all "unaffected" bits would be set). It is recommended that all DDR bits in a port be written using a single-store instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit to be tested, and its condition (set or clear), is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage Self-Check Mode (TIMER Pin Only)	V_{in}	−0.3 to +7.0 −0.3 to +15.0	V
Operating Temperature Range MC6805R3 MC6805R3C MC6805R3V	T_A	T_L to T_H 0 to 70 −40 to +85 −40 to +105	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature Plastic PLCC Cerdip	T_J	 150 150 175	°C/W

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation it is recommended the V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{CC}$. Reliability of operation is enhanced if unused inputs except EXTAL are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

3

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic (P Suffix) PLCC (FN Suffix) Cerdip (S Suffix)	θ_{JA}	 60 100 60	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient; °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

ELECTRICAL CHARACTERISTICS (V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage	V_{IH}				V
$\overline{RESET}$ (4.75 ≤ V_{CC} ≤ 5.75)		4.0	—	V_{CC}	
V_{CC} < 4.75)		V_{CC} − 0.5	—	V_{CC}	
$\overline{INT}$ (4.75 ≤ V_{CC} ≤ 5.75)		4.0	*	V_{CC}	
(V_{CC} < 4.75)		V_{CC} − 0.5	*	V_{CC}	
All Other		2.0	—	V_{CC}	
Input High Voltage Timer	V_{IH}				V
Timer Mode		2.0	—	V_{CC} + 1.0	
Self-Check Mode		9.0	10.0	15.0	
Input Low Voltage	V_{IL}				V
$\overline{RESET}$		V_{SS}	—	0.8	
$\overline{INT}$		V_{SS}	*	1.5	
All Other (Except A/D Inputs)		V_{SS}	—	0.8	
$\overline{RESET}$ Hysteresis Voltages					V
"Out of Reset"	V_{IRES+}	2.1	—	4.0	
"Into Reset"	V_{IRES-}	0.8	—	2.0	
$\overline{INT}$ Zero Crossing Input Voltage, Through a Capacitor	V_{INT}	2	—	4	V_{ac} p-p
Power Dissipation — (No Port Loading,	P_D				mW
V_{CC} 5.75 V T_A = 0°C		—	520	740	
for Steady-State Operation) T_A = −40°C		—	580	800	
Input Capacitance	C_{in}				pF
XTAL		—	25	—	
All Other Except Analog Inputs (See Note)		—	10	—	
Low Voltage Recover	V_{LVR}	—	—	4.75	V
Low Voltage Inhibit	V_{LVI}	2.75	3.75	4.70	V
Input Current					µA
$\overline{TIMER}$ (V_{in} = 0.4)	I_{in}	—	—	20	
$\overline{INT}$ (V_{in} = 2.4 V to V_{CC})		—	20	50	
EXTAL (V_{in} = 2.4 V to V_{CC} Crystal Option)		—	—	10	
(V_{in} = 0.4 V Crystal Option)		—	—	−1600	
$\overline{RESET}$ (V_{in} = 0.8 V)	I_{RES}	−4.0	—	−40	
(External Capacitor Charging Current)					

NOTE: Port D analog inputs, when selected C_{in} = 25 pF for the first 5 out of 30 cycles.
*Due to internal biasing this input (when unused) floats to approximately 2.0 V.

SWITCHING CHARACTERISTICS (V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Characteristic	Symbol	Min	Typ	Max	Unit
Oscillator Frequency	f_{osc}	0.4	—	4.2	MHz
Cycle time ($4/f_{osc}$)	t_{cyc}	0.95	—	10	µs
$\overline{INT}$, $\overline{INT2}$, and TIMER Pulse Width	t_{WL}, t_{WH}	t_{cyc} + 250	—	—	ns
$\overline{RESET}$ Pulse Width	t_{RWL}	t_{cyc} + 250	—	—	ns
$\overline{INT}$ Zero-Crossing Detection Input Frequency	f_{INT}	0.03	—	1	kHz
External Clock Input Duty Cycle (EXTAL)	—	40	50	60	%
Crystal Oscillator Start-Up Time	—	—	—	100	ms

A/D CONVERTER CHARACTERISTICS (V_{CC} = +5.25 Vdc ± 0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H), unless otherwise noted)

	Min	Typ	Max	Unit	Comments
Resolution	8	8	8	Bits	
Total Error	—	—	+/− 2.25*	LSB	Difference between ideal and actual transfer characteristics (includes non-linearity, zero offset and full scale errors)
Absolute Accuracy	—	—	+/− 2.75*	LSB	Difference between the actual input voltage and the full-scale weighted equivalent of the binary output code. All error sources included
Quantizing Error	—	—	+/− .5	LSB	Uncertainty due to converter resolution (inherent)
Conversion Range	V_{RL}	—	VRH	V	
V_{RH}	—	—	V_{CC}	V	A/D accuracy may decrease proportionately as VRH is reduced below 4.75 V. The sum of V_{RH} and V_{RL} must not exceed V_{CC}
V_{RL}	V_{SS}	—	0.2	V	
Conversion Time	30	30	30	t_{cyc}	Includes sample time
Monotonicity					Inherent with total error
Sample Time	5	5	5	t_{cyc}	
Sample/Hold Capacitance, Input	—	—	25	pF	
Analog Input Voltage	V_{RL}	—	V_{RH}	V	Negative transients on any analog lines (Pins 19-24) are not allowed at any time during conversion.

*Note: Accuracy may decrease at temperatures above T_A = 85°C or f_{osc} < 3.57 MHz.

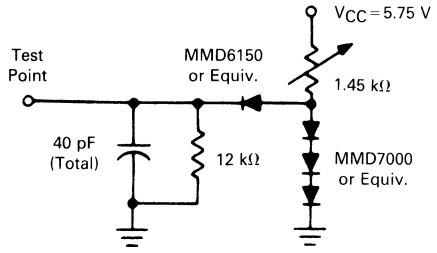


Figure 12. TTL Equivalent Test Load (Port B)

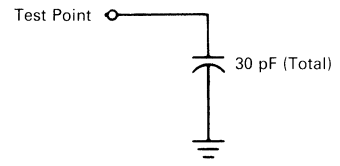


Figure 13. CMOS Equivalent Test Load (Port A)

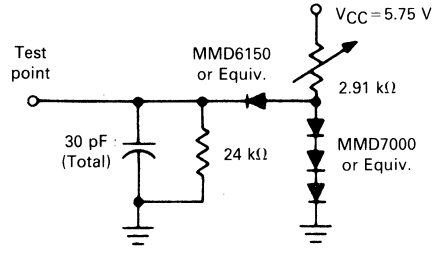


Figure 14. TTL Equivalent Test Load (Ports A and C)

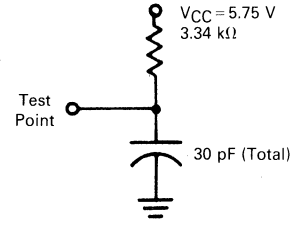


Figure 15. Open-Drain Equivalent Test Load (Port C)

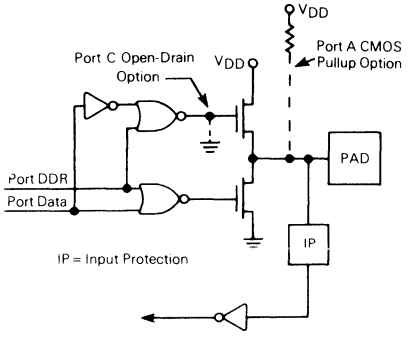


Figure 16. Ports A and C Logic Diagram

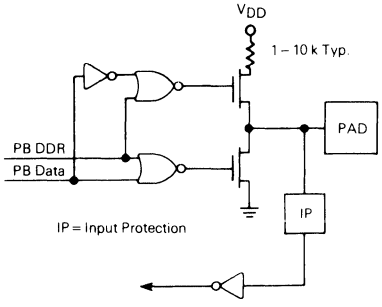


Figure 17. Port B Logic Diagram

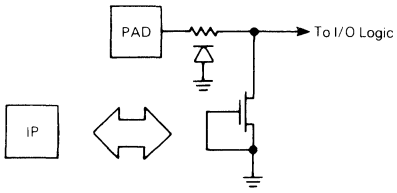


Figure 18. Typical Input Protection

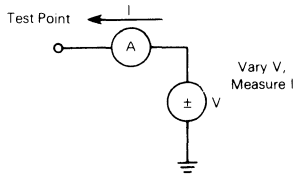


Figure 19. I/O Characteristic Measurement Circuit

3

PORT ELECTRICAL CHARACTERISTICS (V_{CC} = +5.25 Vdc ± 0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Port A with CMOS Drive Enabled					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Output High Voltage, I_{Load} = −10 μA	V_{OH}	V_{CC} − 1	—	—	V
Input High Voltage, I_{Load} = −300 μA (max.)	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage, I_{Load} = −500 μA (max.)	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current (V_{in} = 2.0 V to V_{CC})	I_{IH}	—	—	−300	μA
Hi-Z State Input Current (V_{in} = 0.4 V)	I_{IL}	—	—	−500	μA
Port B					
Output Low Voltage, I_{Load} = 3.2 mA	V_{OL}	—	—	0.4	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.0	V
Output High Voltage, I_{Load} = −200 μA	V_{OH}	2.4	—	—	V
Darlington Current Drive (Source), V_O = 1.5 V	I_{OH}	−1.0	—	−10	mA
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port C and Port A with TTL Drive					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port C (Open-Drain Option)					
Input High Voltage PC0-PC6	V_{IH}	2.0	—	13.0	V
Input High Voltage PC7	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Input Leakage Current (V_{in} = 13.0 V)	I_{LOD}	—	<3	15	μA
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Port D (Digital Inputs Only)					
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Input Current*	I_{in}	—	<1	5	μA

*PD4/V_{RL} — PD5/V_{RH}. The A/D conversion resistor (15 kΩ typical) is connected internally between these two lines, impacting their use as digital inputs in some applications.

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola in the following media:

MDOS, disk file
MS-DOS/PC-DOS disk file
EPROM(s) MC68705R3, 2532, 2732, or two 2516/2716

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, a sales person, or a Motorola representative.

FLEXIBLE DISKS

Several types of flexible disks (MDOS™ or MS™-DOS/PC-DOS disk file), programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. In either case, the diskette should be clearly labeled with the customer's name, date, project or product name, and the name of the file containing the pattern.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to expedite the process in case of any difficulty with the pattern file.

MDOS Disk File

MDOS is Motorola's Disk Operating System available on the EXORciser® development system. The disk media submitted must be a single-side, single-density, 8 inch MDOS compatible floppy diskette. The diskette must contain the minimum set of MDOS system files in addition to the pattern file.

The .LO output of the M6805 cross assembler should be furnished. In addition, the file must be produced (using the ROLLOUT command) containing the absolute image of the M6805 memory. Include the entire memory image of both data and program space. All unused bytes, including those in the user space, must be set to zero.

MS-DOS/PC-DOS Disk File

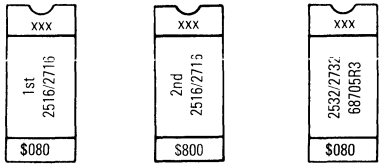
MS-DOS is Microsoft's Disk Operating System. PC-DOS is IBM® Personal Computer (PC) Disk Operating System. Disk media submitted must be a standard density (360K) double-sided 5 1/4 inch compatible floppy diskette. The diskette must contain object file code in Motorola's S-record format. The S-record format is a character-based object file format generated by M6805 cross assemblers and linkers on IBM PC style machines.

EPROMs

A MC68705R3, 2532, 2732, 2516 (2), or 2716 (2) type EPROM(s), programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. Since all program and data space information will fit on one MC68705R3/2532/2732 or two 2516/2716 type EPROM(s), the EPROM(s) must be programmed as described in the following paragraph.

For the 2532, 2732, or MC68705R3, the ROM code should be located from $080 to $F37 and the interrupt vectors from $FF8 to $FFF. For the 2516s or 2716s, the ROM code should be located from $080 to $7FF in the first EPROM and from $0 to $737 in the second EPROM. The interrupt vectors should be in the second EPROM from $7F8 to $7FF.

EPROM MARKING



xxx = Customer ID

VERIFICATION MEDIA

All original pattern media (EPROMs or floppy disk) are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked and the verification form should be completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for creation of the customer mask. To aid in the verification process, Motorola will program ***customer supplied*** blank EPROM(s) or DOS disk from the data file used to create the custom mask.

ROM

Ten MCUs containing the customer's ROM pattern will be sent for program verification. These units will have been made using the custom mask but are for the purpose of ROM verification only. For expediency, the MCUs are usually unmarked, packaged in ceramic, and tested at room temperature and at five volts. These RVUs are free with the minimum order quantity but are not production parts. These RVUs are not guaranteed by Motorola Quality Assurance.

ORDERING INFORMATION

The following table provides generic information pertaining to the package type, temperature, and MC order numbers for the MC6805R3.

Table 5. Generic Information

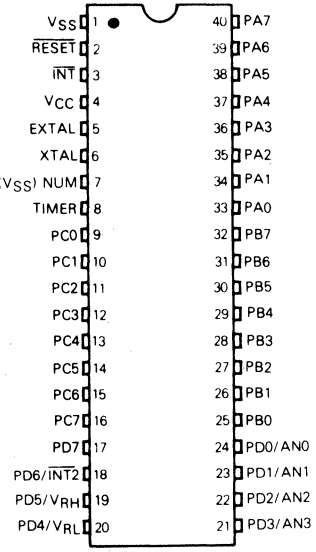
Package Type	Temperature	Order Number
Plastic (P Suffix)	0°C to 70°C −40°C to +85°C	MC6805R3P MC6805R3CP
Cerdip S Suffix	0°C to 70°C −40°C to +85°C	MC6805R3S MC6805R3CS
PLCC FN Suffix	0°C to 70°C −40°C to +85°C	MC6805R3FN MC6805R3CFN

MDOS is a trademark of Motorola Inc.
MS is a trademark of Microsoft, Inc.
EXORciser is a registered trademark of Motorola Inc.
IBM is a registered trademark of International Business Machines Corporation.

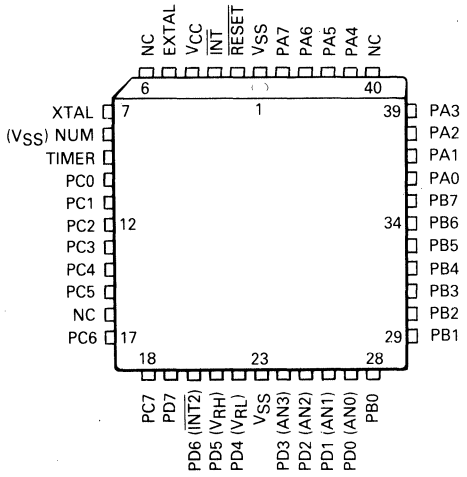
MECHANICAL DATA

PIN ASSIGNMENTS

Dual-in-Line Package



PLCC Package



MC6805S2

Technical Summary
8-Bit Microcontroller Unit

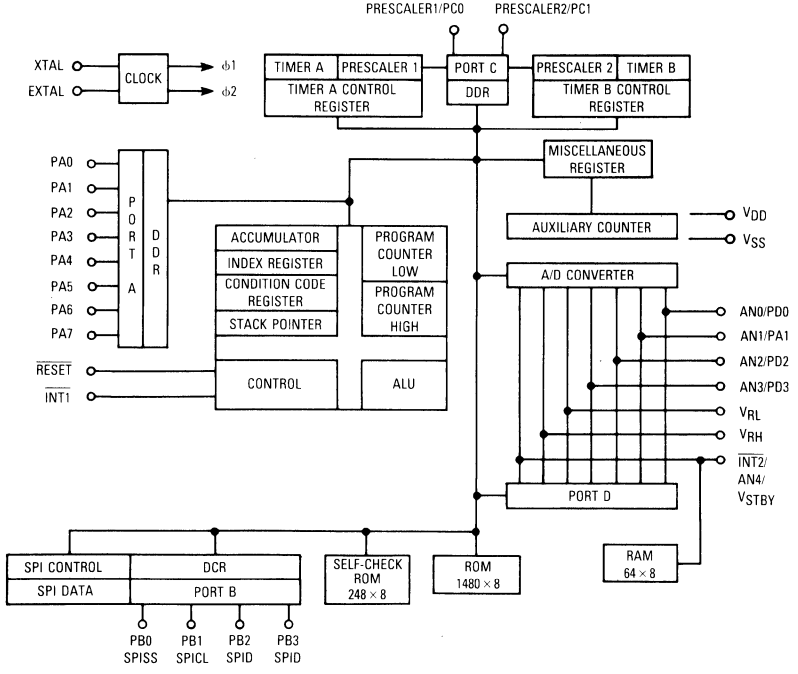
The MC6805S2 (HMOS) Microcontroller Unit (MCU) is a member of the MC6805 Family of microcontrollers. This low cost MCU has parallel I/O capability with pins programmable as input or output. This publication contains condensed information on the MCU; for detailed information, refer to *Advance Information 8-Bit Microcompters* (ADI997R1) or contact your local Motorola sales office.

Refer to the block diagram for the hardware features and to the list below for additional features available on the MCU.

- Internal 7-Bit Timer and 15-Bit Programmable Prescaler
- On-chip Oscillator
- Memory Mapped I/O
- Versatile Interrupt Handling
- Bit Manipulation
- Bit Test and Branch Instruction
- Vectored Interrupts
- Self-Check Mode
- 1480 Bytes of ROM
- 64 Bytes of RAM
- Serial Peripheral Interface (SPI)
- One 8-Bit and One 16-Bit Timer
- A/D Converter

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

V_{CC} and V_{SS}

Power is supplied to the microcontroller using these two pins. V_{CC} is +5.25 volts (±0.5Δ) power, and V_{SS} is ground.

NUM

This pin is for factory use only. It should be connected to V_{SS}.

$\overline{INT1}$, $\overline{INT2}$

These pins provide the capability for asynchronously applying an external interrupt to the MCU. Refer to **INTERRUPTS** for more detailed information.

XTAL, EXTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a resistor/capacitor combination, or an external signal (depending on user selected manufacturing mask option) is connected to these pins to provide a system clock.

RC Oscillator

With this option, a resistor/capacitor combination is connected to the oscillator pins as shown in Figure 1(c). Refer to Figure 2 for the relationship between R and f_{osc}.

Crystal

The circuit shown in Figure 1(b) is recommended when using a crystal. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time.

External Clock

An external clock should be applied to the EXTAL input with the XTAL input not connected, as shown in Figure

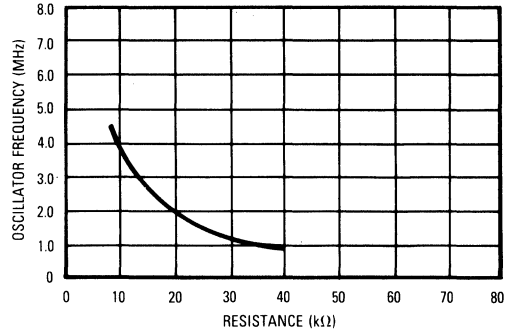
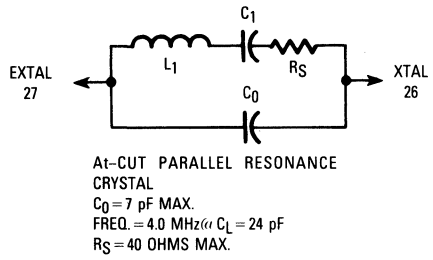
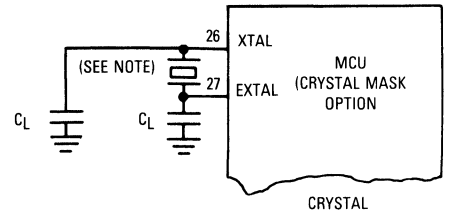
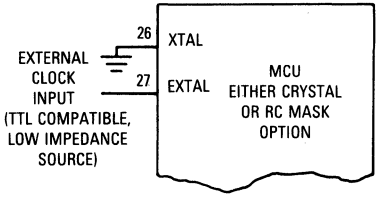
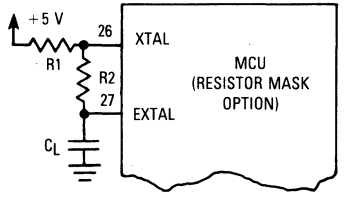


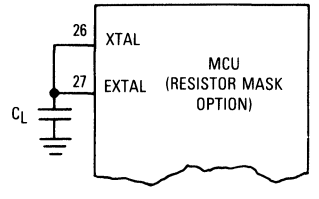
Figure 2. Typical Frequency vs Resistance for RC Oscillator Option Only











NOTE: The recommended C_L value with a 4.0 MHz crystal is 27 pF, maximum, including system distributed capacitance. For crystal frequencies other than 4 MHz, the total capacitance on each pin should be scaled as the inverse of the frequency ratio. For example, with 2 MHz crystal, use approximately 50 pF on EXTAL and approximately 50 pF on XTAL. The exact value depends on the motional-arm parametes of the crystal used.

Figure 1. Oscillator Connections

1(d). This option may only be used with the R_C or XTAL option selected.

PC0, PC1

These pins allow an external input to be used to decrement the internal timer circuit. Refer to **TIMERS** for additional information.

$\overline{\text{RESET}}$

This pin has a Schmitt trigger input and an on-chip pullup. The MCU can be reset by pulling $\overline{\text{RESET}}$ low.

INPUT/OUTPUT LINES (PA0-PA7, PB0-PB3, PC0-PC1, and PD0-PD6)

Port A, B, and C are programmable as either inputs or outputs under software control of the data direction registers. Port D is a fixed input port and not controlled by any data register. Port D has up to four analog inputs or five via the mask option, plus two voltage reference inputs when the analog-to-digital (A/D) converter is used (PD5/V_{RH}, PD4/V_{RL}) and an $\overline{\text{INT2}}$ input. If the analog input is used, then the voltage reference pins (PD5/V_{RH} and PD4/V_{RL}) must be used in the analog mode. Refer to **PROGRAMMING** for additional information.

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

Ports A, B, and C are programmable as either input or output under software control of the corresponding data direction register (DDR). The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic one for output and a logic zero for input.

On reset, all DDRs are initialized to a logic zero state to put the ports in the input mode. The port output registers are not initialized on reset and should be written to before setting the DDR bits.

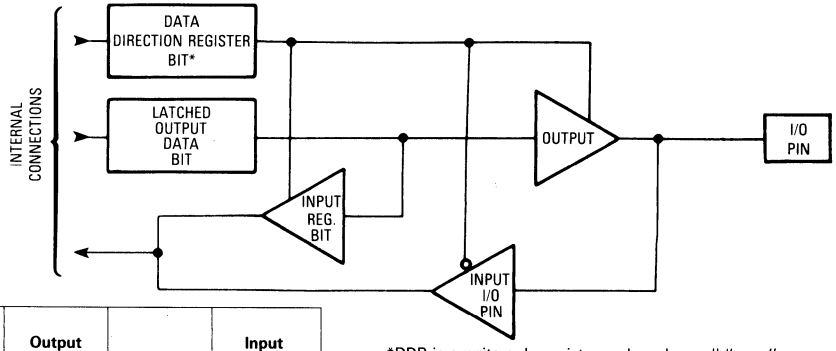
Port D provides the multiplexed analog inputs, reference voltages, and $\overline{\text{INT2}}$. These lines are shared with the port D digital inputs. PD0-PD3 may always be used as digital or analog inputs. The V_{RL} and V_{RH} lines are internally connected by the A/D resistor. Analog inputs may be prescaled to attain the V_{RL} and V_{RH} recommended input voltage range.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This port write may be used to initialize the data registers and avoid undefined outputs. Care must be exercised when using read-modify-write instructions since the data read corresponds to the pin level if the DDR is an input (zero) and, also, to the latched output when the DDR is an output (one). Refer to Figure 3 for typical port circuitry.

PORT B TOGGLE CAPABILITY

Port B0 and B1 registers have toggle capability at the timer underflow times. Under the control of the timer output cross-couple bit in the miscellaneous register (MR0), the overflow pulses from timer A and B are directed to port B0 and B1 data registers. See Figure 4 for port B configuration flow chart.

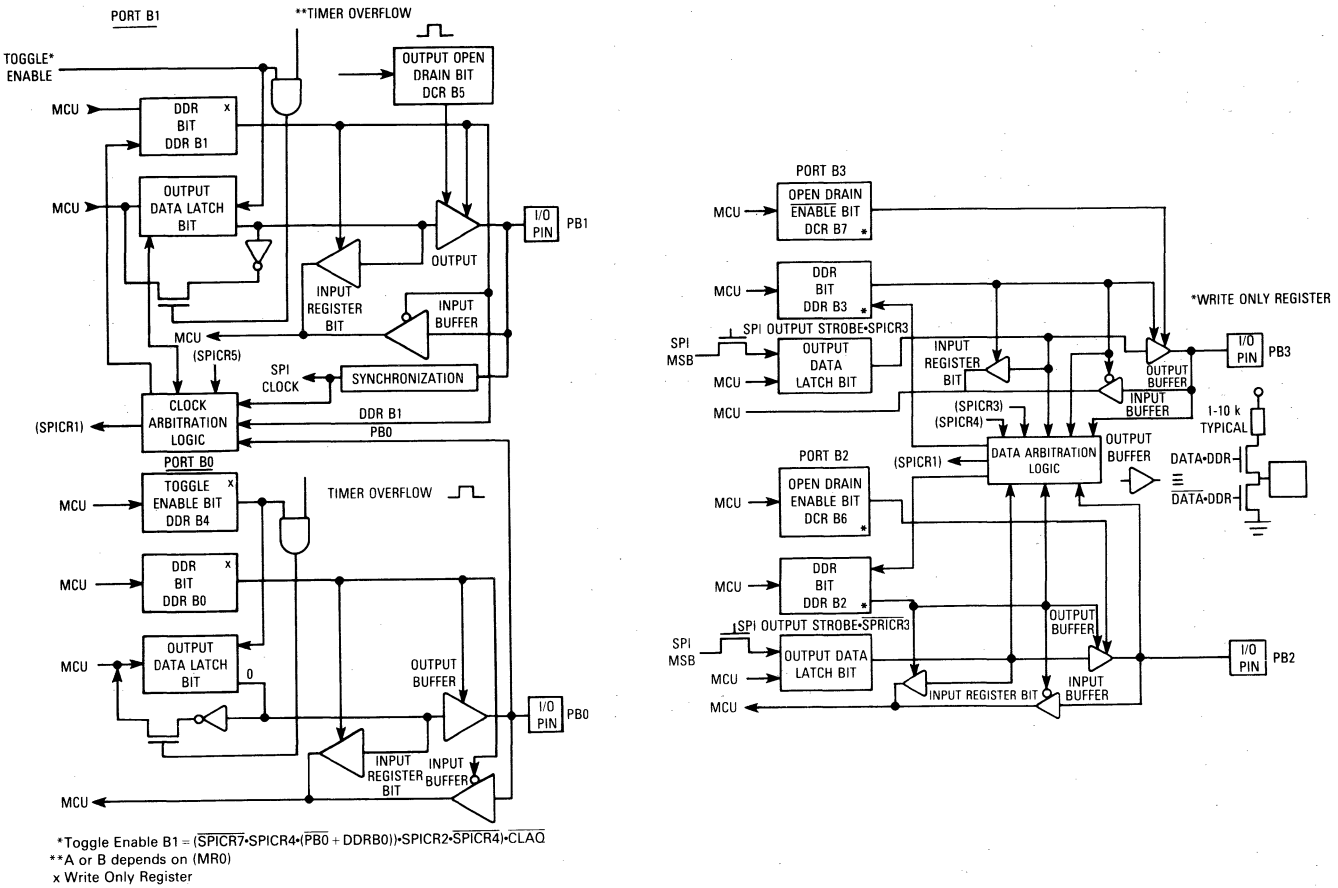
An incoming toggle pulse on port B0 is allowed to toggle the data register if port B DCR bit 4 (DCR4) is cleared. This bit is set on reset. An incoming toggle pulse on port B1 is allowed to toggle the port B1 data register



Data Direction Register Bit	Output Data Bit	Output State	Input To MCU
1	0	0	0
1	1	1	1
0	X	High-Z**	Pin

*DDR is a write-only register and reads as all "ones".
**Ports A (with CMOS drive disabled), B, and C are three-state ports. Port A has optional internal pullup devices to provide CMOS drive capability. See Electrical Characteristics tables for complete information.

Figure 3. Typical Port I/O Circuitry and Register Configuration



*Toggle Enable B1 = $(\overline{SPICR7} \cdot SPICR4 \cdot (\overline{PB0} + DDRB0)) \cdot SPICR2 \cdot \overline{SPICR4}) \cdot \overline{CLAQ}$
**A or B depends on (MR0)
x Write Only Register

Figure 4. Port B Configuration

under the following conditions governed by control bits in SPI control register and SPI clock arbitration flip-flop status.

PB1 toggle enable = $(\overline{SPICR7})$•SPICR4• $(\overline{PB0} + DDRB0)$ + SPICR2• $\overline{SPICR4 \cdot CLAQ}$

where:
SPICR7 = SPI interrupt request bit
SPICR4 = SPI operation enable bit
SPICR2 = port B1 toggle enable/start bit
CLAQ = clock arbitration flip-flop output

When PB1 toggle enable is asserted, the MCU write to PB1 data register is inhibited. When SPI is not used, SPICR4 and CLAQ are reset. Therefore, SPICR2 can directly control the port B1 toggle capability. Port toggle capability allows action on port B0 or B1 or both as a result of timer overflows. This method speeds up timer overflow to port service. A write to port B0 or B1 data registers is inhibited while the individual port toggle enable is asserted.

The port B DCR consists of four status bits (DCR7-DCR4) and four data direction bits (DCR3-DCR0). DCR4 is a toggle enable control bit for port B0. When cleared, the timer overflow pulse causes the data register on port B0 to toggle. Port A has an 8-bit and port C has a 2-bit wide data direction register.

MEMORY

The MCU is capable of addressing 4096 bytes of memory and I/O registers. The memory map is shown in Figure 5. The locations consist of user ROM, self-check ROM, user RAM, five timer registers, a miscellaneous register, two A/D registers, two SPI registers, and I/O. The interrupt vectors are located from $FF8 to $FFF.

The stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer decrements during pushes and increments during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

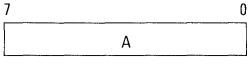
Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

REGISTERS

The MCU contains the registers described in the following paragraphs.

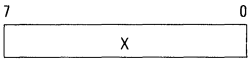
ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



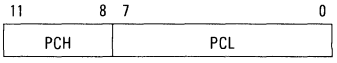
INDEX REGISTER (X)

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16-bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



PROGRAM COUNTER (PC)

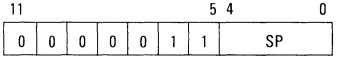
The program counter is a 12-bit register that contains the address of the next byte to be fetched.



STACK POINTER (SP)

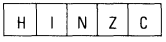
The stack pointer is a 12-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set at location $07F. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

The seven most-significant bits of the stack pointer are permanently set at 0000011. Subroutines and interrupts may be nested down to location $061 (31 bytes maximum), which allows the programmer to use up to 15 levels of subroutine calls (less if interrupts are allowed).



CONDITION CODE REGISTER (CC)

The condition code register is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.



Half Carry (H)

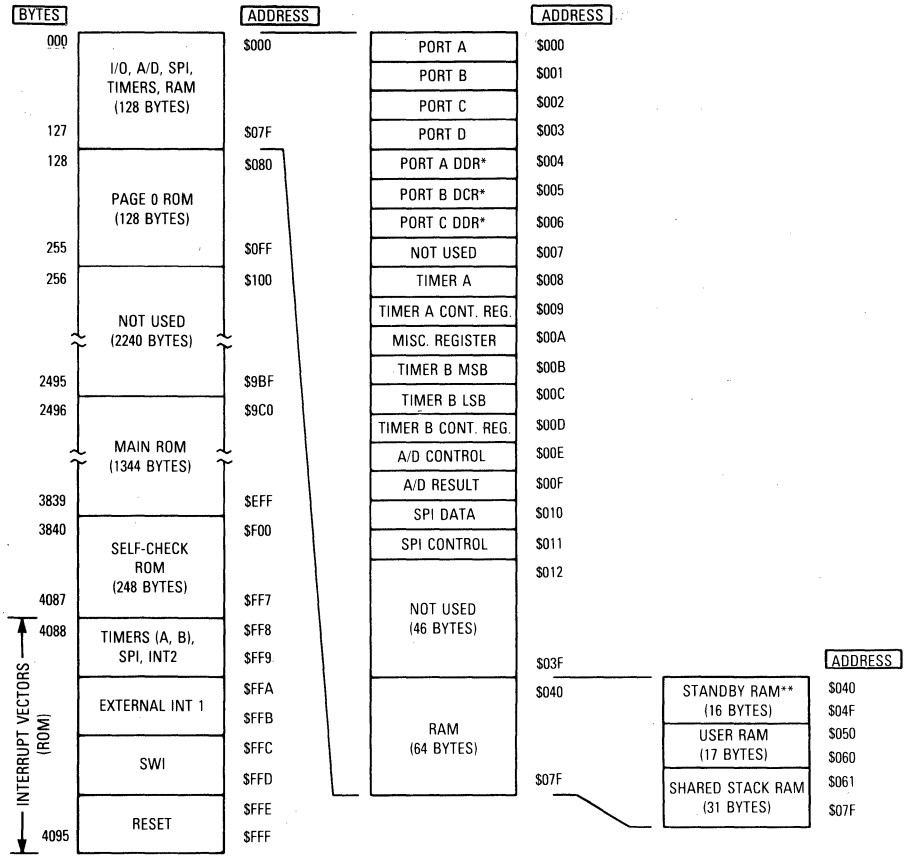
This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, the timers (both A and B), the external ($\overline{INT1}$ and $\overline{INT2}$) interrupts, and the SPI interrupt are masked (disabled). If an interrupt occurs while this bit is set, the interrupt is latched and is processed as soon as the interrupt bit is cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic one).



*Registers are write only and read as $FF.
**Mask option

Figure 5. Memory Map

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/Borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions, and during shifts and rotates.

MISCELLANEOUS REGISTERS (MR) $0A

This register contains control and status information related to $\overline{INT2}$, auxiliary counter, prescalers 1 and 2, and timer overflow.

MR7	MR6	MR5	MR4	MR3	MR2	MR1	MR0
RESET:							
0	1	0	1			0	0

MR7 — $\overline{INT2}$ Interrupt Request Bit
If not masked by MR6, it causes an interrupt to the MCU, and if the I bit in the CCR is clear, the MCU will acknowledge the interrupt.
1 = Interrupt requested
0 = Interrupt not requested

MR6 — $\overline{INT2}$ Interrupt Request Mask
1 = Inhibits $\overline{INT2}$ interrupt request
0 = Does not inhibit $\overline{INT2}$ interrupt request

MR5 — Auxiliary Counter Status/Preset Bit
If not masked by MR4, it will drive a switch to V_{SS} on the $\overline{RESET}$ pin causing the MCU to reset. This bit may

be used as an auxiliary counter preset bit. If MR5 is clear, a write of logic one will preset the auxiliary counter, and if set, a write of logic zero will preset the auxiliary counter.
1 = Auxiliary counter overflow
0 = Auxiliary counter clear

MR4 — Watchdog Control Bit
This bit cannot be set via software. The watchdog timer can only be disabled by reset.
1 = Watchdog timer disabled
0 = Watchdog timer enabled

MR3 — Prescaler 1 Clear Bit
Presets the contents of prescaler 1 to $7F
1 = Prescaler 1 preset
0 = Prescaler 1 not preset

MR2 — Prescaler 2 Clear Bit
Presets the contents of prescaler 2 to $7FFF
1 = Prescaler 2 preset
0 = Prescaler 2 not preset

MR1 — Prescaler Cross-Couple Bit
This bit controls the output of prescalers 1 and 2 and directs them to either timer A or B clock inputs.
1 = Prescaler 1 feeds timer B clock input, and prescaler 2 feeds timer A input
0 = Prescaler 1 output is used as clock input for timer A, and prescaler 2 output is used as clock input for timer B

MR0 — Port B Toggle Cross-Couple Bit
This bit controls the overflow pulses of timers A and B and directs them to either port B0 or B1.
1 = Timer A overflow output is directed to port B0, and timer B output is directed to port B1
0 = Overflow output pulse of timer A is used as a port B1 data register toggle clock source, and timer B overflow output pulse is directed to port B0 toggle clock input

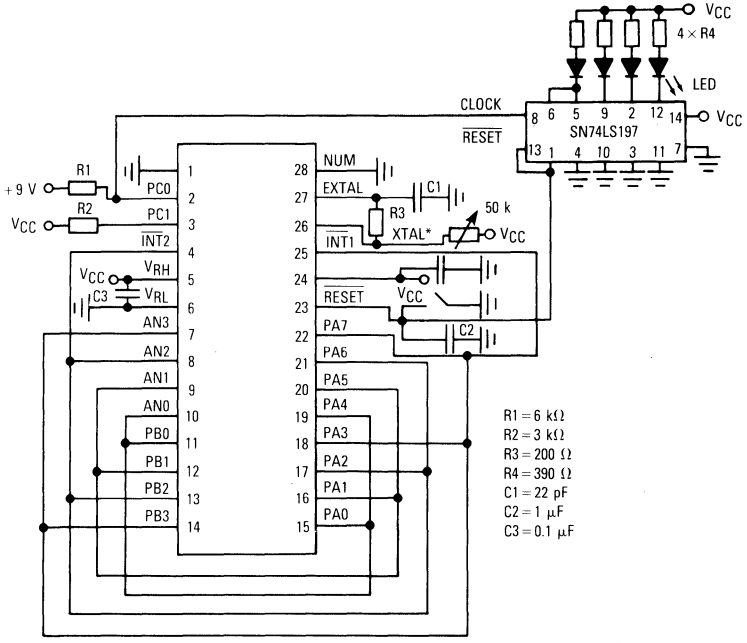
SELF CHECK

The self check is initiated by connecting the MCU as shown in Figure 6 and then monitoring the output of port C (bit 0) for an oscillation of approximately 7 Hz. The self-check program exercises the CPU, I/O, RAM, ROM, timers, interrupts, analog-to-digital (A/D) converter, and the auxiliary counter.

The RAM, ROM, and 4-channel A/D test can be called by a user program. The timer test may be called if the timer input is the internal clock.

RESETS

The MCU can be reset four ways: (1) by initial power-up; (2) by the external reset input ($\overline{\text{RESET}}$); (3) by a forced



*RC Oscillator Option Shown. If Q0-Q2 LEDs Blinking = Device Passes Test
Q3 Blinking = Watchdog Reset Problem

Figure 6. Self-Check Connections

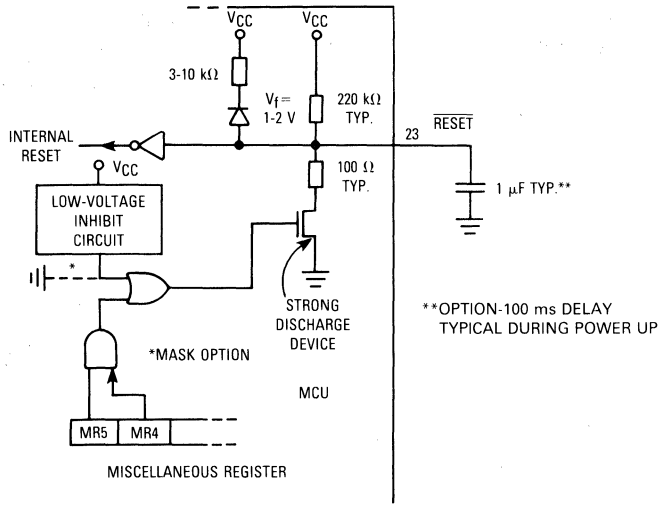


Figure 7. Reset Circuit

reset generated by the "watchdog" counter; and (4) by an optional internal low voltage detect circuit. The $\overline{\text{RESET}}$ input consists mainly of a Schmitt trigger that senses the line logic level. Figure 7 shows the MCU reset circuit.

POWER-ON-RESET (POR)

An internal reset is generated on power-up that allows the internal clock generator to stabilize. The power-on reset is used strictly for power turn-on conditions and should not be used to detect any drop in the power supply voltage. A delay of t_{RHL} milliseconds is required before allowing $\overline{\text{RESET}}$ input to go high. Connecting a capacitor to the $\overline{\text{RESET}}$ input (Figure 8) typically provides sufficient delay.

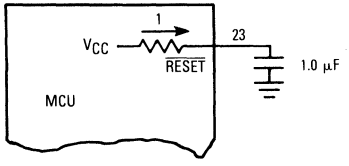


Figure 8. Power-Up Reset Delay Circuit

EXTERNAL RESET INPUT

The MCU is reset when a logic zero is applied to the $\overline{\text{RESET}}$ input for a period longer than one machine cycle (t_{cyc}). Under this type of reset, the Schmitt trigger switches off at V_{IRES-} to provide an internal reset voltage.

FORCED RESET

If the auxiliary counter reset mask bit in the miscellaneous counter (MR4) is cleared and the auxiliary counter status bit (MR5) is set as a result of counter overflow, a switch to V_{SS} is turned on pulling the $\overline{\text{RESET}}$ pin low. A consequent voltage drop below V_{IRES-} on $\overline{\text{RESET}}$ causes a reset, which in turn sets MR4. Switching to V_{SS} when the $\overline{\text{RESET}}$ pin is turned off allows voltage to rise above V_{IRES+}, after which the reset is released. $\overline{\text{RESET}}$ pin voltage variation occurring as a result of forced reset may be amplified externally in order to provide a reset to other peripheral circuits in the system. The reset output from the MCU is not TTL compatible.

LOW-VOLTAGE INHIBIT (LVI)

The optional low-voltage detection circuit causes a reset of the MCU if the power supply voltage falls below a certain level (V_{LVI}). The only requirement is that the V_{CC} must remain at or below the V_{LVI} threshold for one t_{cyc} minimum.

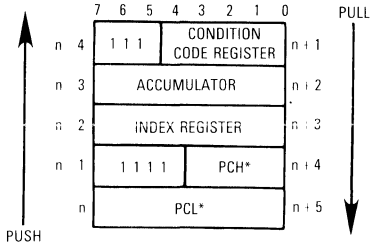
In typical applications, the V_{CC} bus filter capacitor will eliminate negative-going voltage glitches of less than one t_{cyc}. The output from the low-voltage detector is connected directly to the internal reset circuitry. It also forces the $\overline{\text{RESET}}$ pin low via a strong discharge device through a resistor. The internal reset is removed once the power supply voltage rises above a recovery level (V_{LVR}) at which time a normal power-on reset occurs.

INTERRUPTS

The MCU can be interrupted seven different ways: through the external interrupt $\overline{\text{INT1}}$ input pin, with the internal timer (either A or B) interrupt request, using the software interrupt instruction (SWI), SPI interrupt request, external port D bit 6 ($\overline{\text{INT2}}$) input pin, or at reset.

Interrupts cause the processor registers to be saved on the stack and the interrupt mask (I bit) set to prevent

additional interrupts. The RTI instruction causes the register contents to be recovered from the stack after which normal processing resumes. The stacking order is shown in Figure 9.



*For subroutine calls, only PCH and PCL are stacked.

Figure 9. Interrupt Stacking Order

Unlike $\overline{\text{RESET}}$, hardware interrupts do not cause the current instruction execution to be halted, but are considered pending until the current instruction is complete.

NOTE

The current instruction is considered to be the one already fetched and being operated on.

When the current instruction is complete, the processor checks all pending hardware interrupts and, if unmasked (I bit clear), proceeds with interrupt processing; otherwise, the next instruction is fetched and executed. Masked interrupts are latched for later interrupt service. If the timer interrupt status bit is cleared before unmasking the interrupt, then the interrupt is not latched.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction regardless of the setting of the I bit. Refer to Figure 10 for the reset and interrupt instruction processing sequence.

3

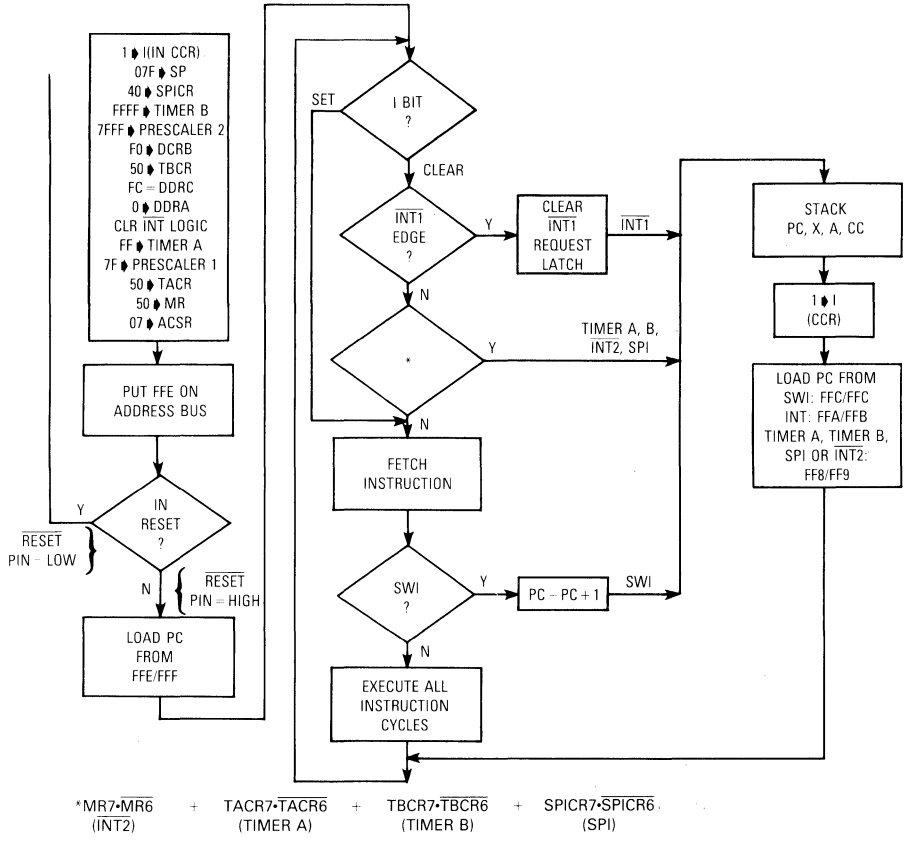


Figure 10. Reset and Interrupt Processing Flowchart

TIMER INTERRUPT

Each interrupt, except $\overline{INT1}$, has a separate mask bit which must also be cleared, in addition to the I bit, for the MCU to acknowledge the interrupt. The $\overline{INT2}$, timer A, timer B, and SPI interrupts each have their own independent mask bits contained in MR6, TACR6, TBCR6, and SPICR6. The interrupt routine must determine the source of the interrupt by examining the interrupt request bits, TACR7, TBCR7, MR7, and SPICR7. These bits must be cleared by software. The $\overline{INT1}$ interrupt has its own vector address. Therefore, the $\overline{INT1}$ interrupt request is cleared automatically, and then the $\overline{INT1}$ vector is serviced.

EXTERNAL INTERRUPT

The external interrupt is internally synchronized and then latched on the falling edge of $\overline{INT1}$ and $\overline{INT2}$. Clearing the I bit enables the external interrupt. The $\overline{INT2}$ interrupt has an interrupt request bit (bit 7) and a mask bit (bit 6) in the miscellaneous register (MR). The $\overline{INT2}$ interrupt is inhibited when the mask bit is set. The $\overline{INT2}$ is always read as a digital input on port D. The $\overline{INT2}$ and timer interrupt request bits, if set, cause the MCU to process an interrupt when the condition code I bit is clear. The following paragraphs describe two typical external interrupt circuits.

3

Zero-Crossing Interrupt

A sinusoidal input signal (f_{INT1} maximum) can be used to generate an external interrupt (see Figure 11a) for use as a zero-crossing detector (for negative transitions of the ac sinusoid). This type of circuit allows applications such as servicing time-of-day routines and engaging/disengaging ac power control devices. Off-chip, full-wave rectification provides an interrupt at every zero crossing of the ac signal and, thereby, provides a 2f clock.

Digital-Signal Interrupt

With this type of circuit (Figure 11b), the $\overline{INT1}$ pin can be driven by a digital signal. The maximum frequency of a signal that can be recognized by the TIMER or $\overline{INT1}$ pin logic is dependent on the parameter labeled t_{WL}, t_{WH}. Refer to **TIMER** for additional information.

SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. If the I bit is zero, SWI executes after the other interrupts. The SWI execution is similar to the hardware interrupts.

TIMERS

The MCU has three timers and two programmable prescalers. The timers are identified as timer A, B, and the auxiliary counter. Refer to Figure 12 for timers A and B block diagram. The following paragraphs described the different timers.

TIMER A

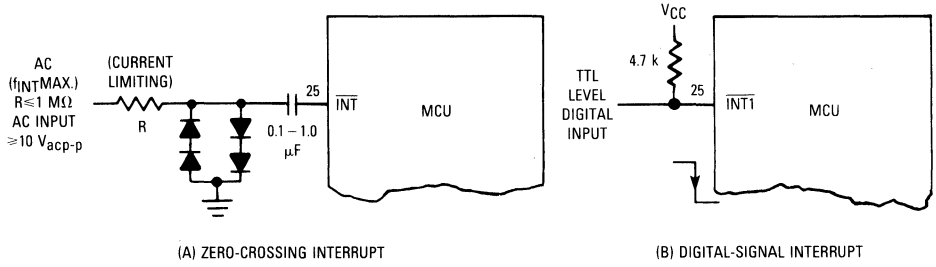
Timer A is an 8-bit programmable counter, which can be loaded under program control. Timer A also includes a modulus latch which allows the timer to be "auto-reloaded." As clock inputs are received, timer A decrements toward $00. When $00 is reached, bit 7 in the timer A control register is set and the timer is reloaded with the contents of the modulus latch. An overflow condition is also generated when value $00 is reached. This state can be used to toggle bit 0 or bit 1 of port B directly under the control of the miscellaneous register (MR0), the SPI control register, and the port B data direction register. Setting TACR6 or the I bit in the condition control register will prevent timer interrupts from being processed. The timer interrupt request bit *MUST* be cleared by software. There are three ways of loading data from the modulus latch into timer A as described in the following paragrahs.

Direct Loading

When the MCU writes to timer A data register, the data is latched by the modulus latch, and forced into the timer. This operation requires that TACR3 be cleared.

Asynchronous External Event Loading

When TACR3 is a logic one, the contents of the modulus latch are transferred to the timer at the rising edge of $\overline{INT2}$ interrupt request bit (MR7) gated with interrupt request mask bit (MR6). If this loading is used, care must



(A) ZERO-CROSSING INTERRUPT

(B) DIGITAL-SIGNAL INTERRUPT

Figure 11. Typical Interrupt Circuits ($\overline{INT1}$)

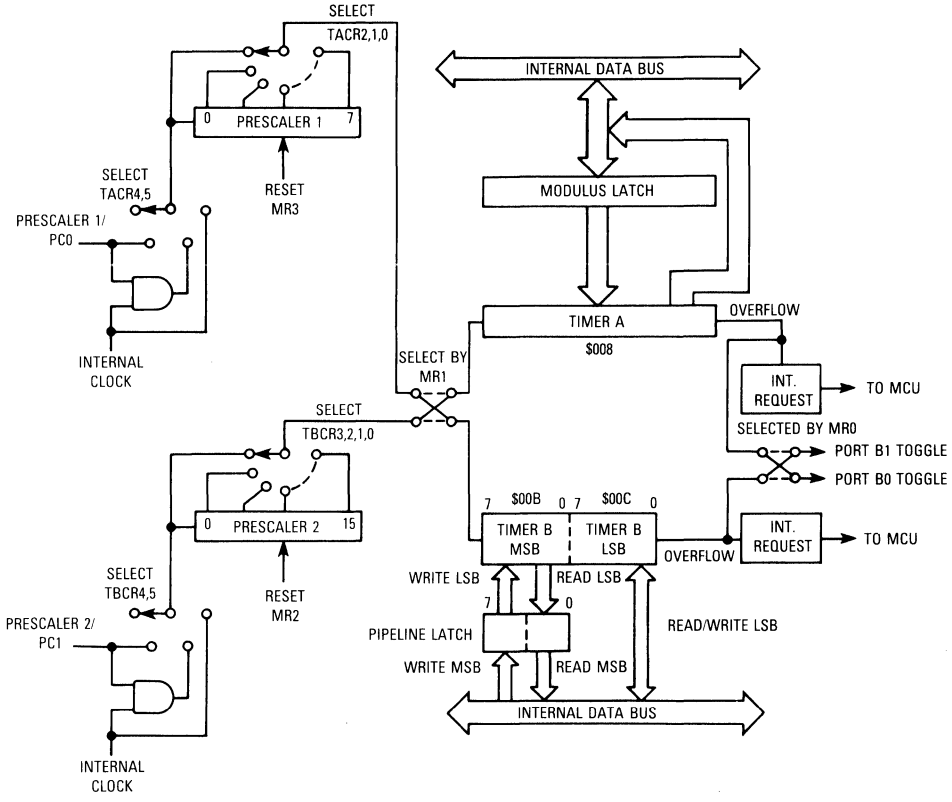


Figure 12. Timers A and B Block Diagram

be taken in programming as it will start an interrupt service routine if the I bit in the CCR is clear. Loading $00 to timer A allows a countdown of 256 clocks before the next $00 state is reached.

Auto-Loading

The modulus latch is automatically loaded when the timer reaches $00. This loading is dependent on the setting of TACR3. Auto-loading also occurs in both the previous loading modes. Timer A can be read at any time without affecting the countdown of the timer. The timer and modulus latch are set to $FF on reset.

NOTE

Loading $01 to timer A should be avoided when operating with a divide-by-one prescaler. Doing so will inhibit timer A auto-loading, interrupt generation, and port B toggle mechanisms.

TIMER A CONTROL REGISTER $09

7	6	5	4	3	2	1	0
TACR7	TACR6	TACR5	TACR4	TACR3	TACR2	TACR1	TACR0
RESET:							
0	1	0	0	0	0	0	0

TACR7 — Timer A Interrupt Request Flag
1 = Timer A has transition to $00
0 = Software or reset cleared
TACR6 — Timer A Interrupt Request Mask
1 = Interrupt request inhibited
0 = Interrupt request not inhibited
TACR5 — External or Internal Bit
1 = External clock source for prescaler 1
0 = Internal clock source for prescaler 1
TACR4 — External Enable Bit
Control bit used to enable the external timer pin (PRESCALER1/PC0).

TACR5	TACR4	Prescaler 1 Clock Source
0	0	Internal Clock
0	1	AND of Internal Clock and PRESCALER1/PC0*
1	0	Inputs Disabled
1	1	PRESCALER1/PC0* Low-to-High Transition

*The status of PRESCALER1/PC0 depends upon the data direction status of PRESCALER1/PC0. If PRESCALER1/PC0 is an output, then the clock source is equal to the port data register content, independent of the port electrical loading. If an input, then the clock source is the logic level of PRESCALER1/PC0.

TACR3 — Timer A Load Mode Control
1 = Asynchronous external event loading ($\overline{INT2}$ driven loading is enabled)
0 = Allows direct loading of timer A

TACR2, TACR1, TACR0 — Prescaler 1 Division Ratio Control Bits
When set, these bits select one of eight possible outputs on prescaler 1.

TACR2	TACR1	TACR0	Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

TIMER B

This is a 16-bit timer which is accessed via two registers ($0B for the most-significant byte (MSB) and $0C for the least-significant byte (LSB)). The MSB has a "pipeline" latch that allows a "snap shot" value of the entire 16 bits to be read. Read/write operations to the LSB are direct. The LSB can be read at anytime without disturbing the count. When the LSB is read, the contents of the MSB are loaded into the pipeline latch so a read of the MSB is actually the contents of the latch.

When writing to the LSB, the contents are immediately entered into the timer. At the same time the pipeline contents are forced into the MSB of the timer. This allows a 16-bit word to be placed into the timer data register during a LSB write operation. An underflow condition is also generated when value $00 is reached. This state can be used to toggle bit 0 or bit 1 of port B directly under the control of the miscellaneous register (MR0), the SPI control register, and the port B data direction register. Setting TBCR6 or the I bit in the condition control register will prevent timer interrupts from being processed. The timer interrupt request bit *MUST* be cleared by software.

TIMER B CONTROL AND STATUS REGISTER $0D

7	6	5	4	3	2	1	0
TBCR7	TBCR6	TBCR5	TBCR4	TBCR3	TBCR2	TBCR1	TBCR0
RESET:							
0	1	0	0	0	0	0	0

TBCR7 — Timer B Interrupt Request Flag
1 = Timer B has transition to $00
0 = Software or reset cleared

TBCR6 — Timer B Interrupt Request Mask
1 = Interrupt request inhibited
0 = Interrupt request not inhibited

TBCR5 — External or Internal Bit
1 = External clock source for prescaler 2
0 = Internal clock source for prescaler 2

TBCR4 — External Enable Bit
Control bit used to enable the external timer pin (PRESCALER2/PC1).

TBCR5	TBCR4	Prescaler 2 Clock Source
0	0	Internal Clock
0	1	AND of Internal Clock and PRESCALER2/PC1*
1	0	Inputs Disabled
1	1	PRESCALER2/PC1* Low-to-High Transition

*The status of PRESCALER2/PC1 depends upon the data direction status of PRESCALER2/PC1. If PRESCALER2/PC1 is an output, then the clock source is equal to the port data register content, independent of the port electrical loading. If an input, then the clock source is the logic level of PRESCALER2/PC1.

TBCR3, TBCR2, TBCR1, TBCR0 — Prescaler 2 Division Ratio Control Bits
When set, these bits select one of eight possible output on prescaler 2.

TBCR3	TBCR2	TBCR1	TBCR0	Divide By
0	0	0	0	1
0	0	0	1	2
0	0	1	0	4
0	0	1	1	8
0	1	0	0	16
0	1	0	1	32
0	1	1	0	64
0	1	1	1	128
1	0	0	0	256
1	0	0	1	512
1	0	1	0	1024
1	0	1	1	2048
1	1	0	0	4096
1	1	0	1	8192
1	1	1	0	16384
1	1	1	1	32768

PRESCALER 1

Prescaler 1 is a 7-bit binary down counter; its value is selected by TACR2, TACR1, and TACR0. The selected output is used as the clock input to either timer A or B, depending upon the status of the prescaler cross-couple bit (MR1). The type of clock source to prescaler 1 may be selected by TACR5 and TACR4. Prescaler 1 is set to $7F at reset or under program control when a one is written to prescaler 1 clear bit (MR3).

PRESCALER 2

Prescaler 2 is a 15-bit down counter; its value is selected by TBCR3, TBCR2, TBCR1, and TBCR0. The selected output is used as the clock input to either timer A or B, depending upon the status of the prescaler cross-couple bit (MR1). The type of clock source to prescaler 2 may be selected by TBCR5 and TBCR4. Prescaler 2 is set to $7FFF at reset or under program control when a one is written to prescaler 2 clear bit (MR2).

AUXILIARY COUNTER

This register is a fixed counter which is clocked by the internal clock (f_{OSC} divided by four). Total count period is 4095 cycles. The MCU communicates with this counter via the miscellaneous register (MR5 and MR4). Countdown may be aborted at any time under program control, which also resets the counter to 4095 and clears MR5. When MR4 is clear and MR5 is set as a result of counter time out, the reset pin is internally pulled to ground. If the MCU loses control of the program, the "watchdog" timer will bring the MCU back to reset. Refer to Figure 13 for counter operation diagram.

SERIAL PERIPHERAL INTERFACE

The serial peripheral interface (SPI) has arbitration on the data and clock lines. The SPI communicates with the MCU via data and control registers. The SPI data and clock inputs are always taken from their respective I/O ports, regardless of the status of the data direction registers relative to that port. The SPI can operate in modes from auto clocked (NRZ), half duplex, and full duplex with from a one wire to a four wire combination. Refer to Figure 14 for the SPI block diagram.

SPI CONTROL AND STATUS REGISTER

This 8-bit register contains the status and control bits relative to SPI operations. The SPI control register operation is shown in Figure 15. The SPI control and status register bits can be set or cleared under program control.

7	6	5	4	3	2	1	0
SPICR7	SPICR6	SPICR5	SPICR4	SPICR3	SPICR2	SPICR1	SPICR0
RESET:							
0	1	0	0	0	0	0	0

SPICR7 — SPI Interrupt Request Bit
Set on eighth data input strobe. MCU services this interrupt if I bit is clear in CCR.
1 = Interrupt request (if SPICR6 not masked)
0 = No interrupt pending

SPICR6 — SPI Interrupt Request Mask Bit
1 = Disables interrupt request from SPICR7
0 = Enables interrupt request from SPICR7

SPICR5 — SPI Clock Sense Bit/Bus-Busy Flag
Dual-function bit controlled by the status of SPICR4
1 = Start SPI operation when SPICR4 = 1. Input data latched on positive edge and output data changed on negative edge of SPI clock when SPICR4 = 0.
0 = Stop SPI operation when SPICR4 = 1. Input data latched on negative edge and output data changed on positive edge of SPI clock when SPICR4 = 0.

SPICR4 — SPI Operation Enable Bit
This bit determines the functions of SPICR5 and SPICR2.
1 = Enables SPI data register shifting, data and clock arbitration logic, and slave select input logic
0 = Disables SPI data register shifting, data and clock arbitration logic, and slave select input logic

SPICR3 — SPI Data Output Select Bit
1 = Output of the SPI data register is loaded to port B3 data register at the appropriate SPI clock edge selected by SPICR5, during the active transaction mode
0 = Output of the SPI data register is loaded to port B2 data register at the appropriate SPI clock edge selected by SPICR5, during the active transaction mode

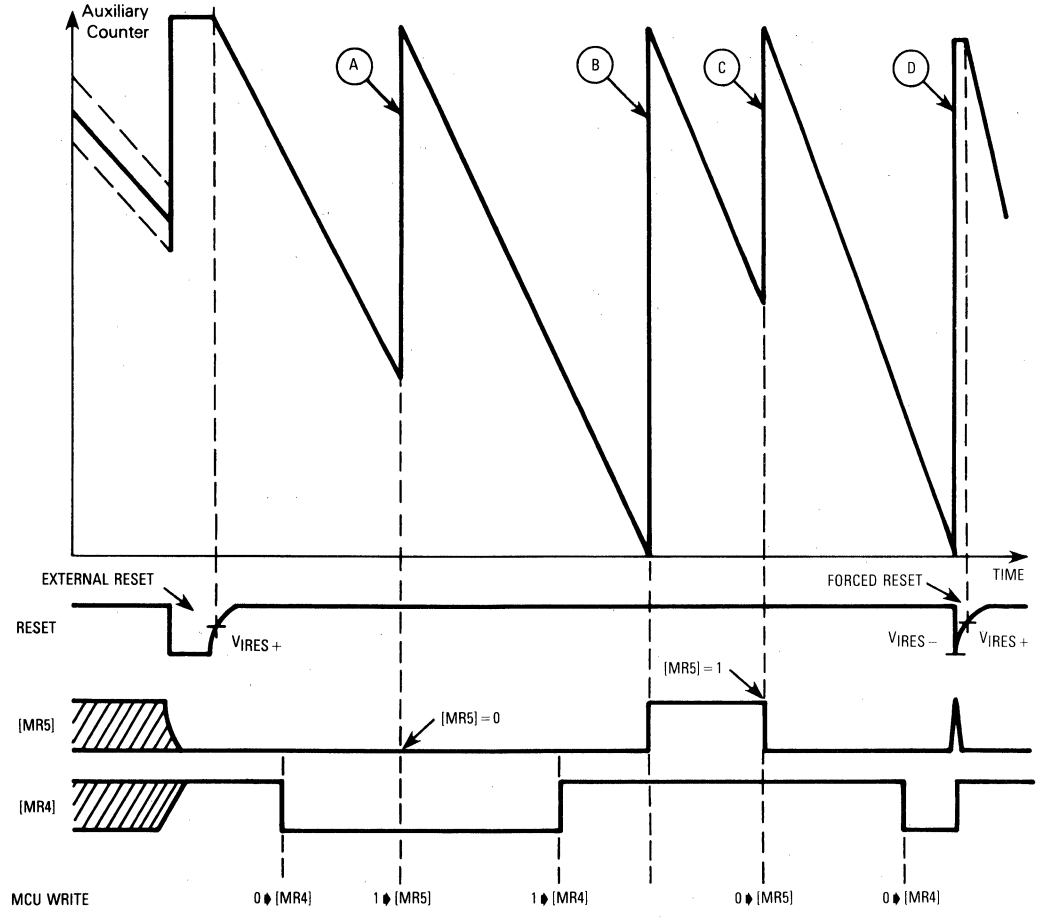
SPICR2 — Port B1 Toggle Enable/Start Bit
Dual-function bit controlled by the status of SPICR4
1 = Start bit is set by negative transition of the data input of the SPI data shift register while the clock is at the idle level when SPICR4 = 1. Start bit set under program control to enable port B1 data register toggle facility when SPICR4 = 0.
0 = Stop SPI operation when SPICR4 = 1. Cleared under program control when SPICR4 = 0.

SPICR1 — Mode Fault Flag
1 = (a) Mode flag is set when SPI data output arbitration occurs on the SPI data output port (PB3 or PB2) selected by SPICR3. The MCU loses data mastership, and the SPI data output port DDR is cleared.
(b) Mode flag is set if a low level is detected on slave input PB0. Then, the MCU loses clock mastership switching to the clock slave mode, and port B1 DDR is cleared.
(c) Mode flag is set during the idle mode when a negative clock edge is detected on the SPI clock input, and the port B1 data register is cleared.
0 = Cleared under program control

SPICR0 — SPI Input Data Select Bit
1 = SPI data from port B3 is latched into the SPI data register
0 = SPI data from port B2 is routed to the input of the SPI data register

SPI DATA REGISTER

This register can be written to any time and can also be read, regardless of serial operations, without disturbing the data. A one bit shift to the left occurs each time there is a data input strobe while the LSB is loaded with data from port B2 or B3. The MSB is loaded every time there is data output strobe. Data input and output strobes



(A) Counter Preset by Writing "1"

(B) Underflow: MR5 ↓ 1; No Forced Reset

(C) Counter Reset by Writing "0"

(D) Underflow MR5 ↓ 1 Forced Reset

Figure 13. Auxiliary Counter Operation

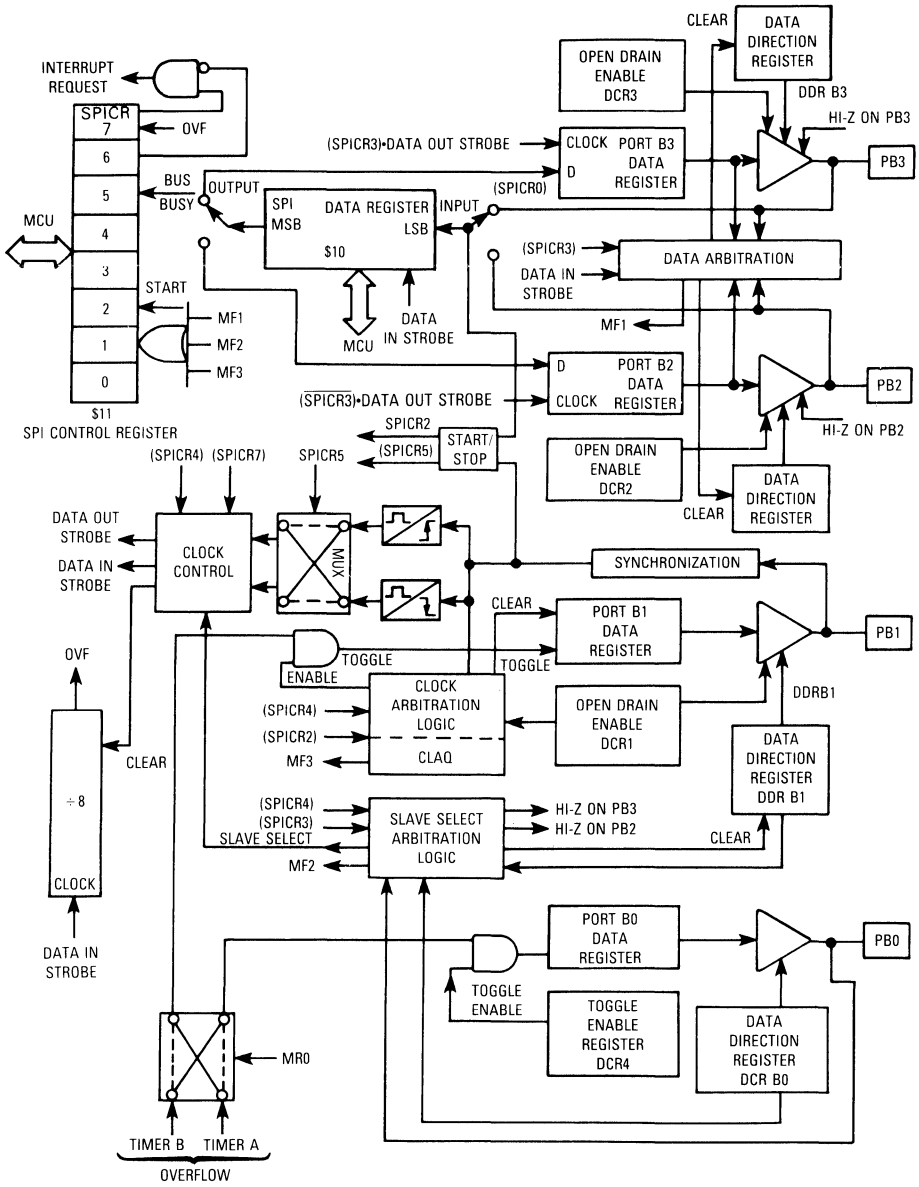


Figure 14. Serial Peripheral Interface Block Diagram

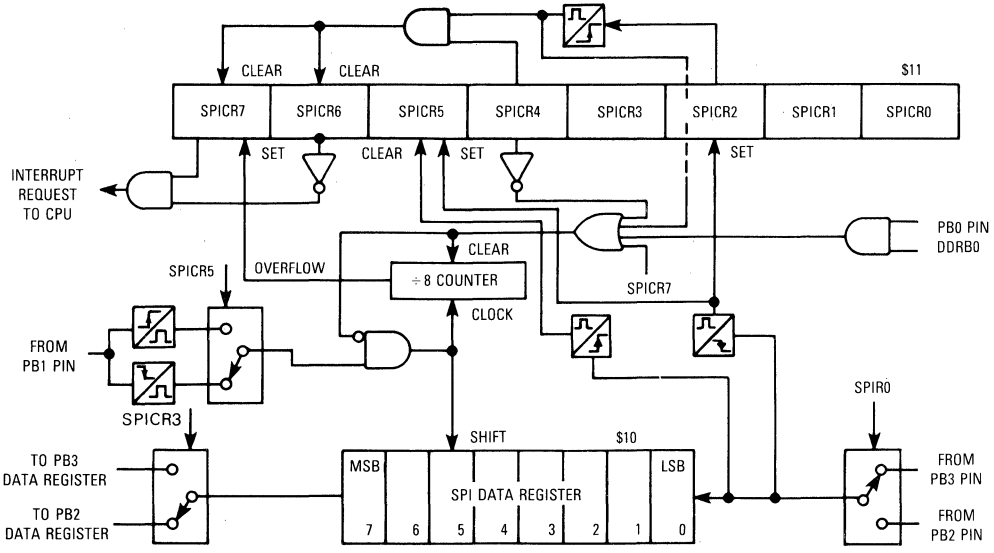


Figure 15. SPI Control Register Operation

are generated internally only during the active transaction time.

SPI DIVIDE-BY-EIGHT COUNTER

The counter is cleared during SPI deselect or idle modes. A count occurs at every data input strobe during the active transaction mode. At overflow, SPICR7 is set which puts the SPI in idle mode and blocks all data input and output strobes. The counter is cleared when PB0 is high if the SPI is in the slave mode or when a "start" condition is detected.

SPI OPERATION

The SPI can operate in a variety of modes. Software assisted protocols may be defined to upgrade the hardware versatility and/or system performance of the MCU. Some features common to all operating modes are summarized in Table 1 and in the following paragraphs.

1) SPI data input and output may be individually routed to or from PB2 or PB3 (Table 2). These four routings provide half and full duplex operations, as well as allowing bidirectional information to flow in daisy-chained systems.
2) When data input and output is done on PB2, PB3 is available for any other use and vice versa.
3) Data input is always relative to the port pin logic level regardless of the data direction register status on that pin.
4) In full duplex operation, 16 bits of information may be transferred with eight clock pulses between at least two devices with transmit capability. Both PB2 and PB3 are used for data transfer. The same shift register is used for data in and data out. The byte transmitted replaces the byte received. SPICR7 is used to signify that the I/O operation is complete.
5) SPI clock is always provided on port B1. In the clock slave mode, port B1 DDR is in the input mode (cleared). In the clock master mode, port B1 DDR is set; therefore, the MCU imposes the clock level on PB1 until there is clock arbitration on the clock line or until the MCU loses clock mastership when PB0 goes low.
6) No fixed baud rate generation exists. The clock frequency is dependent on the prescaler clock source option, prescaler divide ratio, and timer divide ratio as well as the port C status in case of external clocking for the timer. Toggling of the port B1 data register is automatically allowed during the active transmission mode.
7) All devices connected to the SPI must have their output and input data strobe on the same clock edge for correct transfer of data.
8) During the active transmission mode, the first clock edge must be the output data strobe. When this occurs, the MSBs of the data registers of all transmitters are copied onto the data output pins, and the MCU copies the MSB of its SPI onto the port B2 or B3 data register.
9) Port B data direction registers and port B data control registers are accessible during SPI operation. During active transaction mode, the PB1 data register, PB2 data register (if SPICR3 = 0), and PB3 (if

SPICR3 = 1) are not write accessible under program control.

10) Port B lines not used for SPI can be used for other digital functions.

Table 1. Summary of SPI Operations

DEFINITIONS
- Transmitter — Data Master: DDRB2 or 3 = 1
- Receiver — Data Slave: DDRB2 or 3 = 0
- Clock Master: DDRB1 = 1
- Clock Slave: DDRB1 = 0
- Transaction Mode: SPICR4 = 1
  1) Active: $\overline{SPICR7}$•($\overline{DDRB0}$•$\overline{PB0}$ + DDRB0) if DDRB1 = 0 (clock slave mode) or
     $\overline{SPICR7}$•($\overline{DDRB0}$•$\overline{PB0}$ + DDRB0) if DDRB1 = 1 (clock master mode)
     Clock Pulses allowed, data shifted
  2) Idle: SPICR7 + $\overline{DDRB0}$•PB0 if DDRB1 = 0 (clock slave mode)
     Clock pulses blocked, data output line in high-impedance state
- Deselect Mode: SPICR4 = 0 – No SPI Operations

SLAVE SELECT INPUT
- Slave Select Input: SPISS – PB0
  - If DDRB0 = 0 then so SPISS action on MCU
    1) Master Mode: SPISS = 1 DDRB1 = 1
       SPISS 1 – 0: Switch to Slave Mode (DDRB1 1 – 0)
       Set SPICR1 (Mode Fault Flag)
    2) Slave Mode: SPISS = 0 DDRB1 = 0
       External clock is allowed to shift data in/out. If SPISS is pulled high, the external clock input pulses are inhibited; no data shift; divide-by-eight counter cleared; SPID (PB2 or PB3) switched to high-impedance state.
  - Used as Chip-Select Input

DATA ARBITRATION

Data master loses data mastership when data collision occurs during internal data strobe time.

If SPID output port (PB2 or PB3) = 1 while actual pin level is pulled low externally — conflict detected at internal data strobe time.

Then SPICR1 (mode fault flag) is set; SPID output port DDR (B2 or B3) 1 ➧ 0 (high-impedance state).

CLOCK ARBITRATION

MCU has clock mastership (DDRB1 = 1)
1) Via SPISS line (DDRB0 = 0). If SPISS is pulled low, then clock mastership lost; DDRB1 1 ➧ 0 (high-impedance state); SPICR1 is set (mode fault flag).
2) Via clock line SPICL (DDRB1 = 1 and DCRB5 = 0)
   Condition: SPICL must have open-drain output (DCRB5 = 0)

   If clock line is held low externally then clock mastership is not lost; minimum t_{CLH} and t_{CLK} times are guaranteed.

   If SPICL goes low during idle mode then SPICR1 = 1 and clock line is switched low to inhibit the system clock.

MODE FAULT FLAGE OPERATION (SPICR1)

Flag set when any of the following conditions occur:
- Data arbitration occurs on SPID output.
- Clock arbitration with SPISS during master to slave switching.
- Clock arbitration via clock line if SPICL 1 ➧ 0 during idle.

START, STOP, AND CLOCK IDLE CONDITIONS

Clock Idle: The clock level just prior to the transition that causes data on the serial output data line to be changed is defined as the SPI clock idle state.
- SPICR5 = 0: SPICL Idle = Low State
- SPICR5 = 1: SPICL Idle = High State

These definitions are necessary for determining start and stop conditions.

NOTE

Clock idle state can only be defined if SPICR4 = 0 (Deselect Mode)

Start Condition: Any negative transition of the data input line (PB2 or PB3) during an SPICL idle state.
Stop Condition: Any positive transition of the data input line during an SPICL idle state.

3

Table 2. Port B Status During SPI Operation

Port Name	Use	Input	Output	Comments
PB0	SPISS	Yes	No	Used as slave select input
PB0	Data	No	Yes	Used as "busy" signal or any digital output
PB1	SPICL	Yes	No	Clock slave
PB1	SPICL	No	Yes	Clock master
PB2	SPID	Yes	No	SPI data input SPICR0 = 0
PB2	SPID	No	Yes	SPI data output SPICR3 = 0
PB2	Data	Yes	Yes	Any digital signal SPICR3 = 1
PB3	SPID	Yes	No	SPI data input SPICR0 = 1
PB3	SPID	No	Yes	SPI data output SPICR3 = 1
PB3	Data	Yes	Yes	Any digital signal SPICR3 = 0

3

SELECT INPUT OPERATION

An external device supplies slave select information via port B0. If slave select is not used, set port B0 to output mode to inhibit slave select function.

The following paragraphs describe clock master and clock slave operating modes of the SPI.

Master Mode Slave Select Actions

The MCU monitors slave select input in master mode to assure that it stays false. If slave select goes true, the MCU exits master mode and becomes a slave. This implies that a write collision has occurred which means two devices attempted to become masters. Write collisions normally result from a software error, and the default master must clean up the system. The mode fault flag is set to signal that clock mastership is lost. Slave select actions can take place during either active or idle transaction modes.

Slave Select Input Actions During Slave Mode

The current clock master generates slave select to enable one of several slaves to accept or return data. The $\overline{SS}$ signal must go low before serial clock pulses occur and must remain low until after the eighth serial clock cycle. Individual lines or a daisy chain can be used for multiple slaves. When $\overline{SS}$ is high, the following occur:

- Serial data output is forced to a high-impedance state without affecting the DDR status.
- Serial clock input pulses are inhibited from generating internal data output and input strobe pulses.
- The eight-bit counter is cleared.

SPI OPERATING MODES

Six methods of operating the SPI are discussed in the following paragraphs.

One-Wire Autoclocked Mode

Various SPI devices can be connected on a single wire, with data transmission using an implicit clock, and each device being its own clock master.

Two-Wire Half-Duplex Mode

In this mode, separate data and clock lines connect the elements in the system. Data and clock mastership should be monitored via protocol included in the data patterns. A transmitter can send all zeros to take all other transmitters off the bus.

Three-Wire Half-Duplex Mode with Slave Select Input

This mode is the same as the half-duplex mode except that the slave select input allows using the MCU as a peripheral in a system where clock mastership is passed through the slave select line. Typically, the slave select lines can be wired together. The current master sets its slave select line in the output mode prior to a serial transmission and pulls it low to indicate that the system is busy. This allows the clock master to retain mastership until the end of transmission. Software protocol can be arranged so that slaves do not request mastership until their slave select lines go high. At the end of a transmission, the current master pulls $\overline{SPISS}$ high and puts the $\overline{SPISS}$ port (PB0) in the input mode. A slave requesting clock mastership pulls the $\overline{SPISS}$ line low, removing the current master from the line. Time multiplexed protocols may be required to avoid simultaneous mastership requests.

Three-Wire Full-Duplex Mode

This mode allows the MCU to operate simultaneously as transmitter and receiver. Bus or daisy-chain networks are feasible. Protocols in the data stream are required to change:

- Clock masters
- The number of transmitters in the system
- The direction of data flow in daisy-chained systems with collision

It is possible for the MCU to shift out one byte of data while receiving another, as illustrated in Figure 16. This eliminates the need for XMIT EMPTY or REC FULL status bits.

Three-Wire Full-Duplex Mode with Clock Arbitration

This mode is a mix of the three-wire full-duplex mode and two-wire half-duplex mode with clock arbitration, where the SPI clock line operates as a wire-or. Simultaneous masters are allowed, and clock arbitration is via the clock line.

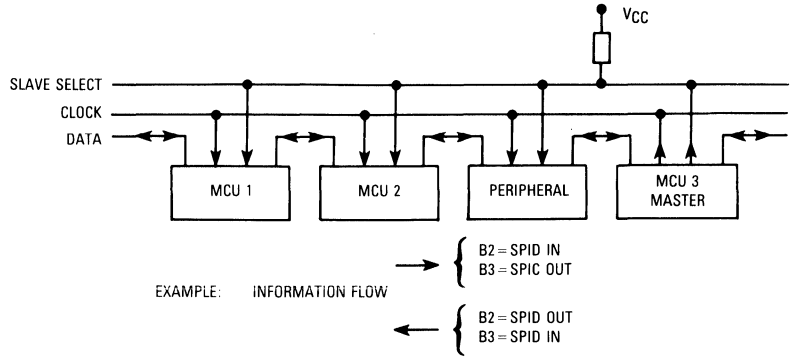


Figure 16. Daisy Chain/Cascade Organization

Four-Wire Full-Duplex Mode with Slave-Select Input

This mode is similar to the three-wire full-duplex mode in network construction and to the three-wire half-duplex mode with slave-select input in clock arbitration and slave selection. Refer to Figure 17.

ANALOG-TO-DIGITAL CONVERTER

The chip resident 8-bit analog-to-digital (A/D) converter uses a successive approximation technique as shown in Figure 18. Four external analog inputs can be connected to the A/D through a multiplexer via port D. Four internal analog channels ($V_{RH} - V_{RL}$, $V_{RH} - V_{RL}/2$, $V_{RH} - V_{RL}/4$, and V_{RL}) may be selected for calibration. The accuracy of these internal channels may not meet the accuracy specifications of the external channels.

A fifth external analog input (AN4) is available via the mask option. When selected, it replaces the V_{RH} internal channel. Due to signal routing, the accuracy of this fifth channel may be slightly less than AN0-AN3.

Multiplexer selection is controlled by the A/D control register (ACR) bits 0, 1, and 2. Refer to Table 3 for multiplexer selection. The ACR is shown in Figure 18. The converter uses 30 machine cycles to complete a conversion of a sampled analog input. When the conversion is complete, the digital value is placed in the A/D result

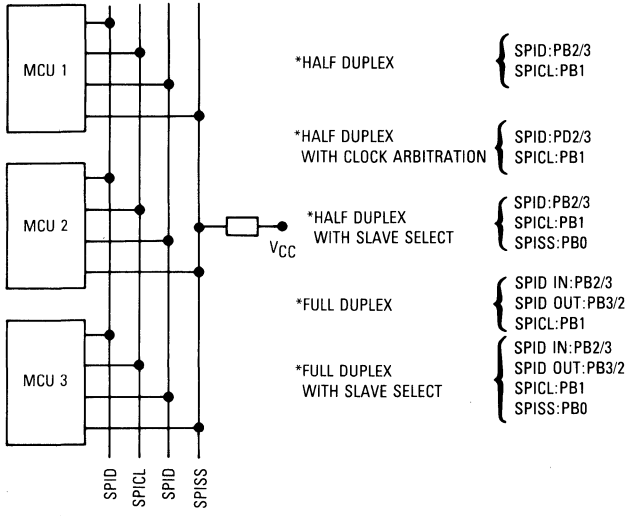


Figure 17. SPI Operation Bus Organization

Table 3. A/D Input MUX Selection

A/D Control Register			Input Selected	A/D Output (Hex)		
ACR2	ACR1	ACR0		Min	Typ	Max
0	0	0	AN0			
0	0	1	AN1			
0	1	0	AN2			
0	1	1	AN3			
1	0	0	V_{RH}**	FE**	FF**	FF**
1	0	1	V_{RL}*	00	00	01
1	1	0	$V_{RH}/4$*	3F	40	41
1	1	1	$V_{RH}/2$*	7F	80	81

*Internal (calibration) levels
**AN4 may replace the V_{RH} calibration channel if selected via mask option.

register (ARR); the conversion flag is set; selected input is sampled again; and a new conversion begins. When ACR7 is cleared, the conversion in progress is aborted and the selected input, which is held internally, is sampled for five machine cycles.

The converter uses V_{RH} and V_{RL} as reference voltages. An input voltage equal to or greater than V_{RH} converts to $FF. An input voltage equal to or less than V_{RL}, but greater than V_{SS}, converts to $00. Maximum and minimum ratings must not be exceeded. Each analog input source should use V_{RH} as the supply voltage and be referenced to V_{RL} for the ratiometric conversion. To maintain full accuracy of the A/D, three requirements should be followed: (1) V_{RH} should be equal to or less than V_{CC}, (2) V_{RL} should be equal to or greater than V_{SS} but less than maximum specifications, and (3) $V_{RH} - V_{RL}$ should be equal to or greater than 4 volts.

The A/D has a built-in 1/2 LSB offset intended to reduce the magnitude of the quantizing error to ±1/2 LSB, rather than +0, −1 LSB with no offset. This implies that, ignoring errors, the transition point from $00 to $01 occurs at 1/2 LSB above V_{RL}. Similarly, the transition from $FE to $FF occurs 1-1/2 LSB below V_{RH}, ideally.

INSTRUCTION SET

The MCU has a set of 61 basic instructions which can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and

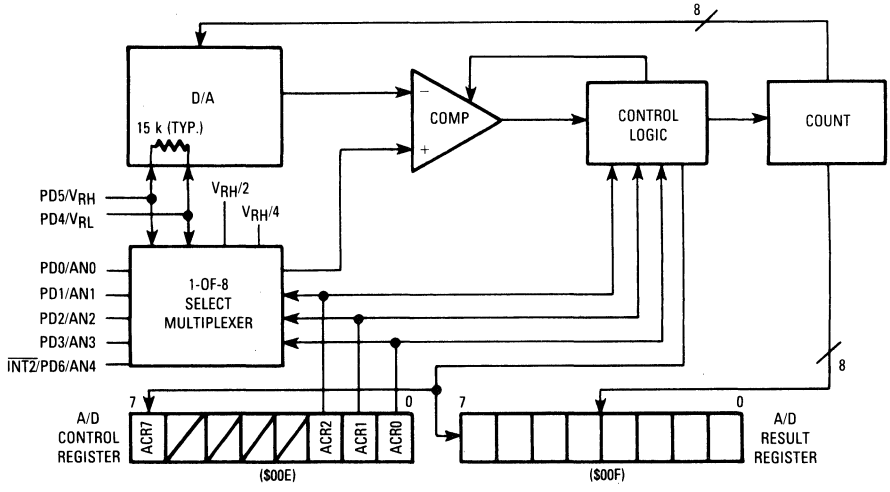


Figure 18. A/D Block Diagram

jump to subroutine (JSR) instructions have no register operand. Refer to the following instruction list.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch IFF Higher	BHI
Branch IFF Lower or Same	BLS
Branch IFF Carry Clear	BCC
(Branch IFF Higher or Same)	(BHS)
Branch IFF Carry Set	BCS
(Branch IFF Lower)	(BLO)
Branch IFF Not Equal	BNE
Branch IFF Equal	BEQ
Branch IFF Half Carry Clear	BHCC
Branch IFF Half Carry Set	BHCS
Branch IFF Plus	BPL
Branch IFF Minus	BMI
Branch IFF Interrupt Mask Bit is Clear	BMC
Branch IFF Interrupt Mask Bit is Set	BMS
Branch IFF Interrupt Line is Low	BIL
Branch IFF Interrupt Line is High	BIH
Branch to Subroutine	BSR

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (2's Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the first 256 bytes of the memory space where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear and bit test, and branch functions are all implemented with a single instruction. For test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch if Bit n is Set	BRSET n (n=0 . . . 7)
Branch if Bit n is Clear	BRCLR n (n=0 . . . 7)
Set Bit n	BSET n (n=0 . . . 7)
Clear Bit n	BCLR n (n=0 . . . 7)

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No-Operation	NOP

OPCODE MAP SUMMARY

Table 4 is an opcode map for the instructions used on the MCU.

Low \ Hi	Bit Manipulation BTB 0 0000	Bit Manipulation BSC 1 0001	Branch REL 2 0010	Read-Modify-Write DIR 3 0011	Read-Modify-Write INH 4 0100	Read-Modify-Write INH 5 0101	Read-Modify-Write IX1 6 0110	Read-Modify-Write IX 7 0111	Control INH 8 1000	Control INH 9 1001	Register/Memory IMM A 1010	Register/Memory DIR B 1011	Register/Memory EXT C 1100	Register/Memory IX2 D 1101	Register/Memory IX1 E 1110	Register/Memory IX F 1111	Hi \ Low
0 0000	10 BRSET0 3 BTB	7 BSET0 2 BSC	4 BRA 2 REL	6 NEG 2 DIR	4 NEG 1 INH	4 NEG 1 INH	7 NEG 2 IX1	6 NEG 1 IX	9 RTI 1 INH		2 SUB 2 IMM	4 SUB 2 DIR	5 SUB 3 EXT	6 SUB 3 IX2	5 SUB 2 IX1	4 SUB 1 IX	0 0000
1 0001	10 BRCLR0 3 BTB	7 BCLR0 2 BSC	4 BRN 2 REL						6 RTS 1 INH		2 CMP 2 IMM	4 CMP 2 DIR	5 CMP 3 EXT	6 CMP 3 IX2	5 CMP 2 IX1	4 CMP 1 IX	1 0001
2 0010	10 BRSET1 3 BTB	7 BSET1 2 BSC	4 BHI 2 REL								2 SBC 2 IMM	4 SBC 2 DIR	5 SBC 3 EXT	6 SBC 3 IX2	5 SBC 2 IX1	4 SBC 1 IX	2 0010
3 0011	10 BRCLR1 3 BTB	7 BCLR1 2 BSC	4 BLS 2 REL	6 COM 2 DIR	4 COMA 1 INH	4 COMX 1 INH	7 COM 2 IX1	6 COM 1 IX	11 SWI 1 INH		2 CPX 2 IMM	4 CPX 2 DIR	5 CPX 3 EXT	6 CPX 3 IX2	5 CPX 2 IX1	4 CPX 1 IX	3 0011
4 0100	10 BRSET2 3 BTB	7 BSET2 2 BSC	4 BCC 2 REL	6 LSR 2 DTR	4 LSRA 1 INH	4 LSRX 1 INH	7 LSR 2 IX1	6 LSR 1 IX			2 AND 2 IMM	4 AND 2 DIR	5 AND 3 EXT	6 AND 3 IX2	5 AND 2 IX1	4 AND 1 IX	4 0100
5 0101	10 BRCLR2 3 BTB	7 BCLR2 2 BSC	4 BCS 2 REL								2 BIT 2 IMM	4 BIT 2 DIR	5 BIT 3 EXT	6 BIT 3 IX2	5 BIT 2 IX1	4 BIT 1 IX	5 0101
6 0110	10 BRSET3 3 BTB	7 BSET3 2 BSC	4 BNE 2 REL	6 ROR 2 DIR	4 RORA 1 INH	4 RORX 1 INH	7 ROR 2 IX1	6 ROR 1 IX			2 LDA 2 IMM	4 LDA 2 DIR	5 LDA 3 EXT	6 LDA 3 IX2	5 LDA 2 IX1	4 LDA 1 IX	6 0110
7 0111	10 BRCLR3 3 BTB	7 BCLR3 2 BSC	4 BEQ 2 REL	6 ASR 2 DIR	4 ASRA 1 INH	4 ASRX 1 INH	7 ASR 2 IX1	6 ASR 1 IX		2 TAX 1 INH		5 STA 2 DIR	6 STA 3 EXT	7 STA 3 IX2	6 STA 2 IX1	5 STA 1 IX	7 0111
8 1000	10 BRSET4 3 BTB	7 BSET4 2 BSC	4 BHCC 2 REL	6 LSL 2 DIR	4 LSLA 1 INH	4 LSLX 1 INH	7 LSL 2 IX1	6 LSL 1 IX		2 CLC 1 INH	2 EOR 2 IMM	4 EOR 2 DIR	5 EOR 3 EXT	6 EOR 3 IX2	5 EOR 2 IX1	4 EOR 1 IX	8 1000
9 1001	10 BRCLR4 3 BTB	7 BCLR4 2 BSC	4 BHCS 2 REL	6 ROL 2 DIR	4 ROLA 1 INH	4 ROLX 1 INH	7 ROL 2 IX1	6 ROL 1 IX		2 SEC 1 INH	2 ADC 2 IMM	4 ADC 2 DIR	5 ADC 3 EXT	6 ADC 3 IX2	5 ADC 2 IX1	4 ADC 1 IX	9 1001
A 1010	10 BRSET5 3 BTB	7 BSET5 2 BSC	4 BPL 2 REL	6 DEC 2 DIR	4 DECA 1 INH	4 DECX 1 INH	7 DEC 2 IX1	6 DEC 1 IX		2 CLI 1 INH	2 ORA 2 IMM	4 ORA 2 DIR	5 ORA 3 EXT	6 ORA 3 IX2	5 ORA 2 IX1	4 ORA 1 IX	A 1010
B 1011	10 BRCLR5 3 BTB	7 BCLR5 2 BSC	4 BMI 2 REL							2 SEI 1 INH	2 ADD 2 IMM	4 ADD 2 DIR	5 ADD 3 EXT	6 ADD 3 IX2	5 ADD 2 IX1	4 ADD 1 IX	B 1011
C 1100	10 BRSET6 3 BTB	7 BSET6 2 BSC	4 BMC 2 REL	6 INC 2 DIR	4 INCA 1 INH	4 INCX 1 INH	7 INC 2 IX1	6 INC 1 IX		2 RSP 1 INH		3 JMP 2 DIR	4 JMP 3 EXT	5 JMP 3 IX2	4 JMP 2 IX1	3 JMP 1 IX	C 1100
D 1101	10 BRCLR6 3 BTB	7 BCLR6 2 BSC	4 BMS 2 REL	6 TST 2 DIR	4 TSTA 1 INH	4 TSTX 1 INH	7 TST 2 IX1	6 TST 1 IX		2 NOP 1 INH	8 BSR 2 REL	7 JSR 2 DIR	8 JSR 3 EXT	9 JSR 3 IX2	8 JSR 2 IX1	7 JSR 1 IX	D 1101
E 1110	10 BRSET7 3 BTB	7 BSET7 2 BSC	4 BIL 2 REL								2 LDX 2 IMM	4 LDX 2 DIR	5 LDX 3 EXT	6 LDX 3 IX2	5 LDX 2 IX1	4 LDX 1 IX	E 1110
F 1111	10 BRCLR7 3 BTB	7 BCLR7 2 BSC	4 BIH 2 REL	6 CLR 2 DIR	4 CLRA 1 INH	4 CLRX 1 INH	7 CLR 2 IX1	6 CLR 1 IX		2 TXA 1 INH		5 STX 2 DIR	6 STX 3 EXT	7 STX 3 IX2	6 STX 2 IX1	5 STX 1 IX	F 1111

Abbreviations for Address Modes

INH	Inherent
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND

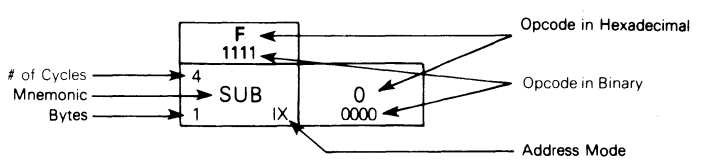


Table 4. Opcode Map

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single-byte instructions, while the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. Two-byte direct-addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction.

RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −126 to +129 from the opcode address.

INDEXED, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. Thus, this addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this two-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This addressing mode can be used in a manner similar to indexed 8-bit offset except that this three-byte instruction allows tables to be anywhere in memory.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte to which the specified bit is to be set or cleared. Thus, any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single two-byte instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit to be tested, and its condition (set or clear), is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage PC0 in Self-Check Mode All Other	V_{in}	 −0.3 to +15.0 −0.3 to +7.0	V
Port A and C Source Current per Pin (One at a Time)	I_{out}	10	mA
Operating Temperature Range MC6805S2P MC6805S2CP	T_A	 0 to 70 −40 to +85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature Plastic Package	T_J	 150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation, it is recommended the V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{CC}$. Reliability of operation is enhanced if unused inputs except EXTAL are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic (P Suffix)	θ_{JA}	 70	°C/W

3

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

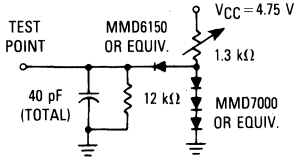


Figure 19. TTL Equivalent Test Load (Port B)

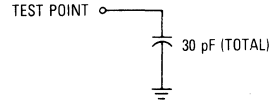


Figure 20. CMOS Equivalent Test Low (Port A)

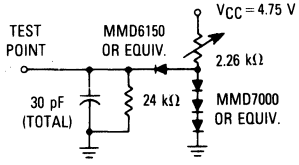


Figure 21. TTL Equivalent Test Load (Ports A and C)

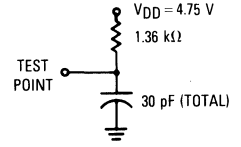


Figure 22. Open-Drain Equivalent Test Load (PB1, PB2, and PB3)

ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage	V_{IH}				V
$\overline{RESET}$ (4.75 ≤ V_{CC} ≤ 5.75)		4.0	—	V_{CC} + 0.7	
V_{CC} ≤ 4.75)		V_{CC} − 0.5	—	V_{CC} + 0.7	
$\overline{INT}$ (4.75 ≤ V_{CC} ≤ 5.75)		4.0	*	V_{CC} + 0.7	
(V_{CC} ≤ 4.75)		V_{CC} − 0.5	*	V_{CC} + 0.7	
All Other		2.0	—	V_{CC} + 0.7	
Input High Voltage Timer PC0	V_{IH}				V
Port/Timer Mode		2.0	—	V_{CC} + 1.0	
Self-Check Mode		9.0	10.0	15.0	
Input Low Voltage	V_{IL}				V
$\overline{RESET}$		V_{SS}	—	0.8	
$\overline{INT}$		V_{SS}	*	1.5	
All Other (Except A/D Inputs)		V_{SS}	—	0.8	
$\overline{RESET}$ Hysteresis Voltages (See RESETS)					V
"Out of Reset"	V_{IRES+}	2.1	—	4.0	
"Into Reset"	V_{IRES-}	0.8	—	2.0	
Standby Supply Voltage ($\overline{INT2}$ Input Option)	V_{STBY}	3.0	—	5.75	V
Standby Current ($\overline{INT2}$ Input Option) (V_{STBY} = 3.0 V)	I_{STBY}	—	1.0	5.0	mA
Power Dissipation — No Port Loading					mW
(V_{CC} = 5.75 V, T_A = 0°C)	P_D	—	600	830	
(V_{CC} = 5.75 V, T_A = −40°C)	P_D	—	670	890	
Input Capacitance (Except Analog Inputs — See Note)	C_{in}	—	10	—	pF
Low Voltage Recover	V_{LVR}	—	—	4.75	V
Low Voltage Inhibit	V_{LVI}	2.75	3.75	4.70	V
Input Current	I_{in}				μA
$\overline{INT}$		—			
(V_{in} = 2.4 V to V_{CC})			20	50	
EXTAL		—			
(V_{in} = 2.4 V to V_{CC} Crystal Option)		—	—	10	
(V_{in} = 0.4 V Crystal Option)			—	−1600	
$\overline{RESET}$		−4.0			
(V_{in} = 0.8 V)			—	−50	
(External Charging Current)					

TBD = To Be Determined

NOTE: Port D analog inputs, when selected, C_{in} = 25 pF for the first 5 out of 30 cycles.

*This input (when unused) floats to approximately 2.0 V due to internal biasing.

3

SWITCHING CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Oscillator Frequency	f_{osc}	0.4	—	4.2	MHz
Cycle time ($4/f_{osc}$)	t_{cyc}	0.95	—	10	µs
$\overline{INT}$, $\overline{INT2}$, and TIMER Pulse Width $\overline{RESET}$ Pulse Width	t_{WL}, t_{WH} t_{RWL}	$t_{cyc}+250$ $t_{cyc}+250$	— —	— —	ns
$\overline{RESET}$ Delay Time (External Capacitance = 1 µF)	t_{RHL}	—	100	—	ns
$\overline{INT}$ Zero-Crossing Detection Input Frequency (for ±5° Accuracy)	f_{INT}	0.03	—	1	kHz
External Clock Input Duty Cycle (EXTAL)	—	40	50	60	%
Oscillator Startup Time Crystal	t_{su}	—	—	100	ms
SPICL High Time	t_{SPICLH}	4	—	—	t_{cyc}
SPICL Low Time	t_{SPICHL}	4	—	—	t_{cyc}
SPICL Rise and Fall Time	t_{Sr}, t_{Sf}	—	—	1	µs
SPID Input Data Setup Time	t_{SDS}	2	—	—	t_{cyc}
SPID Input Data Hold Time	t_{SDH}	2	—	—	t_{cyc}
SPICL to SPISS Lag Time	t_{SStG}	4	—	—	t_{cyc}
SPISS to SPICL Lead Time	t_{SSLD}	4	—	—	t_{cyc}
Start Bit to First Clock Lead Time	t_{STL}	1	—	—	t_{cyc}
External Timer Input to Timer Change Time	t_{PCT}	3	—	—	t_{cyc}
Timer Change to Port B Toggle Time	t_{TPB}	2	—	—	t_{cyc}
$\overline{INT2}$ to Timer A Load Time	t_{INTL}	3	—	—	t_{cyc}

A/D CONVERTER CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H), unless otherwise noted

Characteristic	Min	Typ	Max	Unit	Comments
Resolution	8	8	8	Bits	
Non-Linearity*	—	—	±1/2	LSB	After removing zero-offset and full-scale errors
Quantizing Error	—	—	±1/2	LSB	
Conversion Range V_{RH} V_{RL}	 — V_{SS}	 — —	 V_{CC} 0.2	V V	A/D accuracy may decrease proportionately as $V_{RH} - V_{RL}$ is reduced below 4.0 V. The sum of V_{RH} and V_{RL} must not exceed V_{CC}
Conversion Time	30	30	30	t_{cyc}	Includes sampling time
Monotonicity	(Inherent within total error)				
Sample Time	5	5	5	t_{cyc}	
Sample/Hold Capacitance, Input	—	—	25	pF	
Analog Input Voltage	V_{RL}	—	V_{RH}	V	Transients on any analog lines are not allowed at any time during sampling or accuracy may be degraded

PORT ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Port A with CMOS Drive Enable					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Output High Voltage, I_{Load} = −10 μA	V_{OH}	V_{CC} − 1.0	—	—	V
Input High Voltage, I_{Load} = −300 μA (max.)	V_{IH}	2.0	—	V_{CC} + 0.7	V
Input Low Voltage, I_{Load} = −500 μA (max.)	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current (V_{in} = 2.0 V to V_{CC})	I_{IH}	—	—	−300	μA
Hi-Z State Input Current (V_{in} = 0.4 V)	I_{IL}	—	—	−500	μA
Port B					
Output Low Voltage, I_{Load} = 3.2 mA	V_{OL}	—	—	0.4	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.0	V
Output High Voltage, I_{Load} = −200 μA	V_{OH}	2.4	8	—	V
Darlington Current Drive (Source)*, V_O = 1.5 V	I_{OH}	−1.0	—	−10	mA
Input High Voltage	V_{IH}	2.0	—	V_{CC} + 0.7	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port C and Port A with CMOS Drive Disabled					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC} + 0.7	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port D (Digital Inputs Only)					
Input High Voltage	V_{IH}	2.0	—	V_{CC} + 0.7	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Input Current**	I_{in}	—	<1	10	μA

*Not applicable if programmed to open-drain state.

**PD4/V_{RL} — PD5/V_{RH}:
The A/D conversion resistor (15 kΩ typical) is connected internally between these two lines, impacting their use as digital inputs in some applications.

3

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola in the following media:

MDOS, disk file
MS-DOS/PC-DOS disk file
EPROM(s) 2532, 2732, or two each: 2516/2716

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, a sales person, or a Motorola representative.

FLEXIBLE DISKS

Several types of flexible disks (MDOS™ or MS™-DOS/PC-DOS disk file), programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. In either case, the diskette should be clearly labeled with the customer's name, date, project or product name, and the name of the file containing the pattern.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to expedite the process in case of any difficulty with the pattern file.

MDOS Disk File

MDOS is Motorola's Disk Operating System available on the EXORciser® development system. The disk media submitted must be a single-side, single-density, 8-inch MDOS compatible floppy diskette. The diskette must contain the minimum set of MDOS system files in addition to the pattern file.

The .LO output of the M6805 cross assembler should be furnished. In addition, the file must be produced (using the ROLLOUT command) containing the absolute image of the M6805 memory. Include the entire memory image of both data and program space. All unused bytes, including those in the user space, must be set to zero.

MS-DOS/PC-DOS Disk File

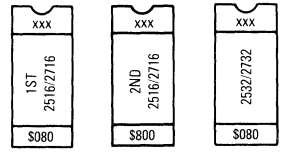
MS-DOS is Microsoft's Disk Operating System. PC-DOS is IBM® Personal Computer (PC) Disk Operating System. Disk media submitted must be a standard density (360K) double-sided 5 1/4 inch compatible floppy diskette. The diskette must contain object file code in Motorola's S-record format. The S-record format is a character-based object file format generated by M6805 cross assemblers and linkers on IBM PC style machines.

EPROMs

An MC68705S3, 2532, 2732, 2516 (2), or 2716 (2) type EPROM(s), programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. Since all program and data space information will fit on one MC68705S3/2532/2732 or two 2516/2716 type EPROM(s), the EPROM(s) must be programmed as described in the following paragraph.

For the 2532 or 2732, the ROM code should be located from $080 to $FF and $9C0 to $EFF, and the interrupt vectors from $FF8 to $FFF. For the 2516s or 2716s, the ROM code should be located from $080 to $FF in the first EPROM and from $1C0 to $6EF in the second EPROM. The interrupt vectors should be in the second EPROM from $7F8 to $7FF.

EPROM MARKING



xxx = CUSTOMER ID

VERIFICATION MEDIA

All original pattern media (EPROMs or floppy disk) are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked and the verification form should be completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for creation of the customer's mask. To aid in the verification process, Motorola will program ***customer supplied*** blank EPROM(s) or DOS disk from the data file used to create the custom mask.

ROM

Ten MCUs containing the customer's ROM pattern will be sent for program verification. These units will have been made using the custom mask but are for the purpose of ROM verification only. For expediency, the MCUs are usually unmarked, packaged in ceramic, and tested at five volts and at room temperature. These RVUs are free with the minimum order quantity but are not production parts. These RVUs are not guaranteed by Motorola Quality Assurance.

ORDERING INFORMATION

The following table provides generic information pertaining to the package type, temperature, and MC order numbers for the MC6805S2.

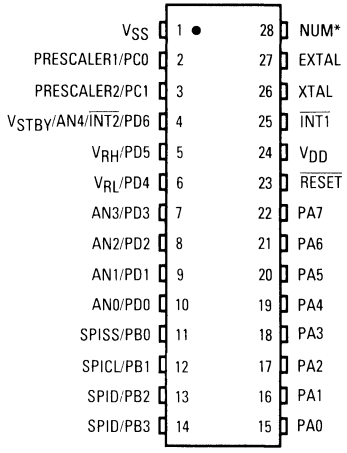
Table 5. Generic Information

Package Type	Temperature	Order Number
Plastic (P Suffix)	0°C to 70°C −40°C to +85°C	MC6805S2P MC6805S2CP

MDOS is a trademark of Motorola Inc.
MS is a trademark of Microsoft, Inc.
EXORciser is a registered trademark of Motorola Inc.
IBM is a registered trademark of International Business Machines Corporation.

MECHANICAL DATA

PIN ASSIGNMENT



NOTE: *Denotes Non User Mode (NUM) pin reserved for factory use only. This pin should be tied to V_{SS} (GND/ground).

3

MC6805S3

Technical Summary

8-Bit Microcontroller Unit

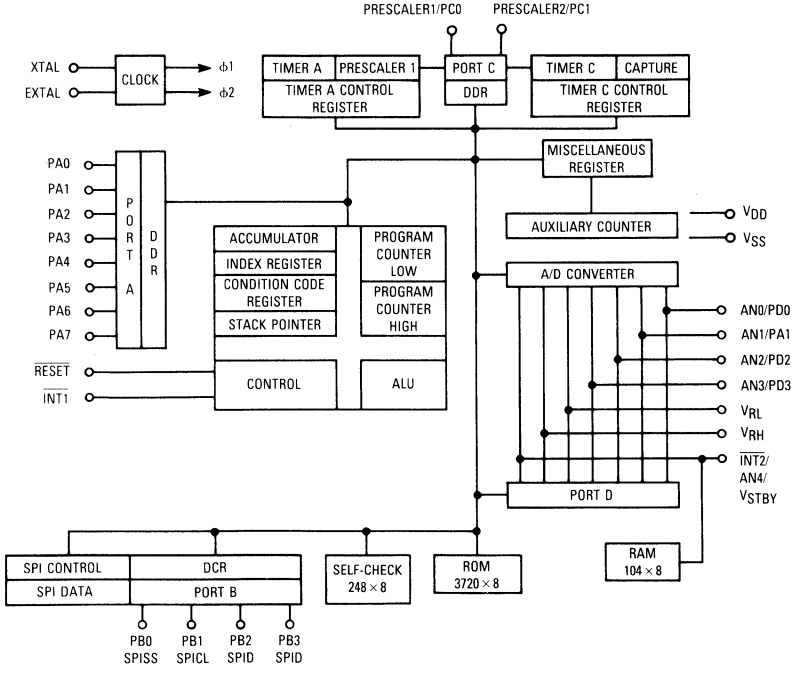
The MC6805S3 (HMOS) Microcontroller Unit (MCU) is a member of the MC6805 Family of microcontrollers. This low cost MCU has parallel I/O capability with pins programmable as either input or output. This publication contains condensed information on the MCU; for detailed information, refer to *Advance Information 8-Bit Microcontrollers* (ADI997R1) or contact your local Motorola sales office.

Refer to the block diagram for the hardware features and to the list below for additional features available on the MCU.

- Internal 7-Bit Timer and 15-Bit Programmable Prescaler
- On-chip Oscillator
- Memory Mapped I/O
- Versatile Interrupt Handling
- Bit Manipulation
- Bit Test and Branch Instruction
- Vectored Interrupts
- Self-Check Mode
- 3720 Bytes of ROM
- 104 Bytes of RAM
- Serial Peripheral Interface (SPI)
- Two 8-Bit and One 16-Bit Timers
- A/D Converter

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

V_{CC} and V_{SS}

Power is supplied to the microcontroller using these two pins. V_{CC} is +5.25 volts (±0.5Δ) power, and V_{SS} is ground.

NUM

This pin is for factory use only. It should be connected to V_{SS}.

$\overline{INT1}$, $\overline{INT2}$

These pins provide the capability for asynchronously applying an external interrupt to the MCU. Refer to **INTERRUPTS** for more detailed information.

XTAL, EXTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a resistor/capacitor combination, or an external signal (depending on user selected manufacturing mask option) is connected to these pins to provide a system clock.

RC Oscillator

With this option, a resistor/capacitor combination is connected to the oscillator pins as shown in Figure 1(c). The relationship between R and f_{osc} is shown in Figure 2.

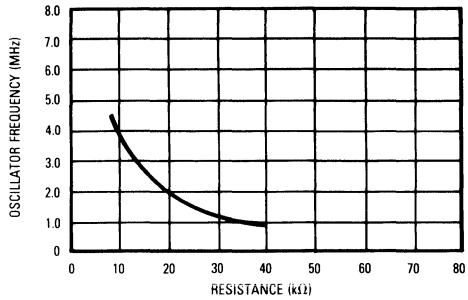


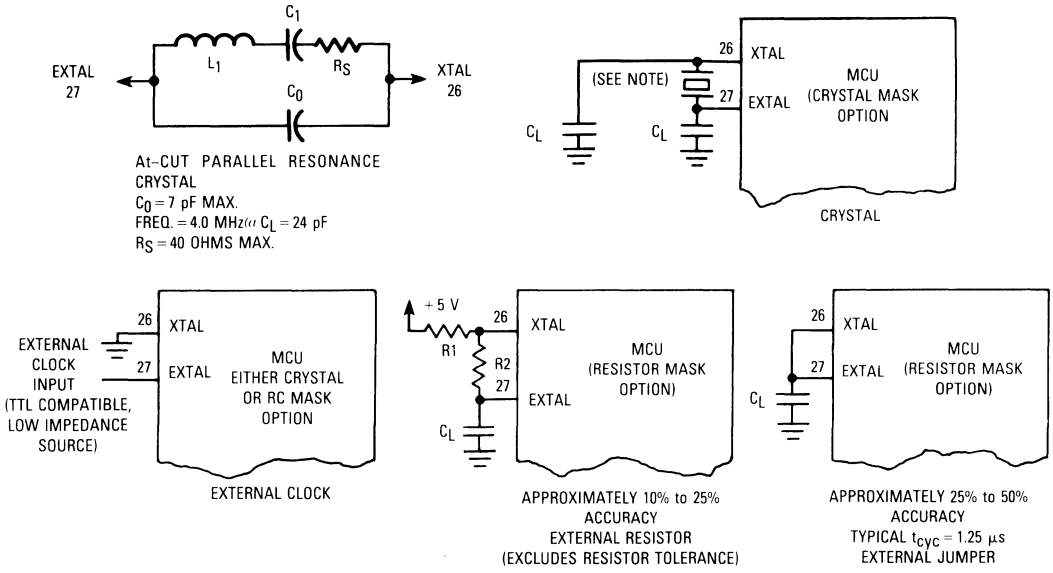
Figure 2. Typical Frequency vs Resistance for RC Oscillator Option Only

Crystal

The circuit shown in Figure 1(b) is recommended when using a crystal. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time.

External Clock

An external clock should be applied to the EXTAL input with the XTAL input grounded, as shown in Figure 1(d).



NOTE: The recommended C_L value with a 4.0 MHz crystal is 27 pF, maximum, including system distributed capacitance. For crystal frequencies other than 4 MHz, the total capacitance on each pin should be scaled as the inverse of the frequency ratio. For example, with 2 MHz crystal, use approximately 50 pF on EXTAL and approximately 50 pF on XTAL. The exact value depends on the motional-arm parametes of the crystal used.

Figure 1. Oscillator Connections

This option may be used with either RC or XTAL option selected.

PC0, PC1

These pins allow an external input to be used to decrement the internal timer circuit. Refer to **TIMERS** for additional information.

$\overline{\text{RESET}}$

This pin has a Schmitt trigger input and an on-chip pullup. The MCU can be reset by pulling $\overline{\text{RESET}}$ low.

INPUT/OUTPUT LINES (PA0-PA7, PB0-PB3, PC0-PC1, and PD0-PD6)

Ports A, B, and C are programmable as either inputs or outputs under software control of the data direction registers. Port D is a fixed input port and not controlled by any data register. Port D has up to four analog inputs or five via the mask option, plus two voltage reference inputs when the analog-to-digital (A/D) converter is used (PD5/V_{RH}, PD4/V_{RL}) and an $\overline{\text{INT2}}$ input. If the analog input is used, then the voltage reference pins (PD5/V_{RH} and PD4/V_{RL}) must be used in the analog mode. Refer to **PROGRAMMING** for additional information.

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

Ports A, B, and C are programmable as either input or output under software control of the corresponding data direction register (DDR). The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic one for output and a logic zero for input.

On reset, all DDRs are initialized to a logic zero state to put the ports in the input mode. The port output registers are not initialized on reset and should be written to before setting the DDR bits.

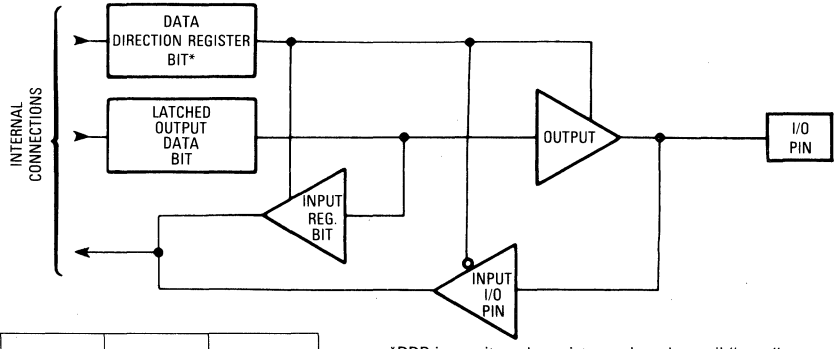
Port D provides the multiplexed analog inputs, reference voltages, and $\overline{\text{INT2}}$. These lines are shared with the port D digital inputs. PD0-PD3 may always be used as digital or analog inputs. The V_{RL} and V_{RH} lines are internally connected by the A/D resistor. Analog inputs may be prescaled to attain the V_{RL} and V_{RH} recommended input voltage range.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This port write may be used to initialize the data registers and avoid undefined outputs. Care must be exercised when using read-modify-write instructions since the data read corresponds to the pin level if the DDR is an input (zero) and, also, to the latched output when the DDR is an output (one). Refer to Figure 3 for typical port circuitry.

PORT B TOGGLE CAPABILITY

Port B0 and B1 registers have toggle capability at the timer underflow times. Under the control of the timer output cross-couple bit in the miscellaneous register (MR0), the overflow pulses from timer A, B, and C are directed to port B0 and B1 data registers. See Figure 4 for port B configuration flow chart.

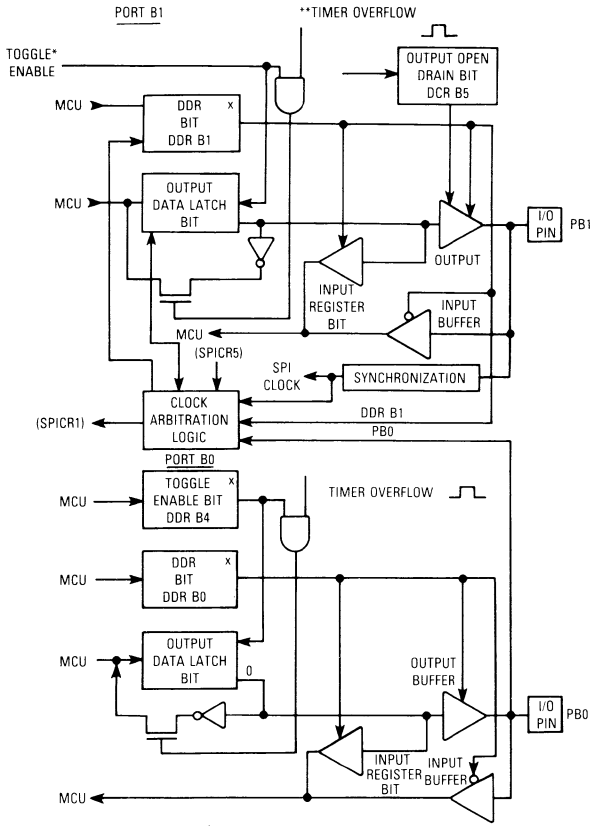
An incoming toggle pulse on port B0 is allowed to toggle the data register if port B DCR bit 4 (DCR4) is cleared. This bit is set on reset. An incoming toggle pulse on port B1 is allowed to toggle the port B1 data register



Data Direction Register Bit	Output Data Bit	Output State	Input To MCU
1	0	0	0
1	1	1	1
0	X	High-Z**	Pin

*DDR is a write-only register and reads as all "ones".
**Ports A (with CMOS drive disabled), B, and C are three-state ports. Port A has optional internal pullup devices to provide CMOS drive capability. See Electrical Characteristics tables for complete information.

Figure 3. Typical Port I/O Circuitry and Register Configuration



*Toggle Enable B1 = (SPICR7•SPICR4•(PB0 - DDRB0))•SPICR2•SPICR4)•CLAQ
**A or B depends on (MR0)
x Write Only Register

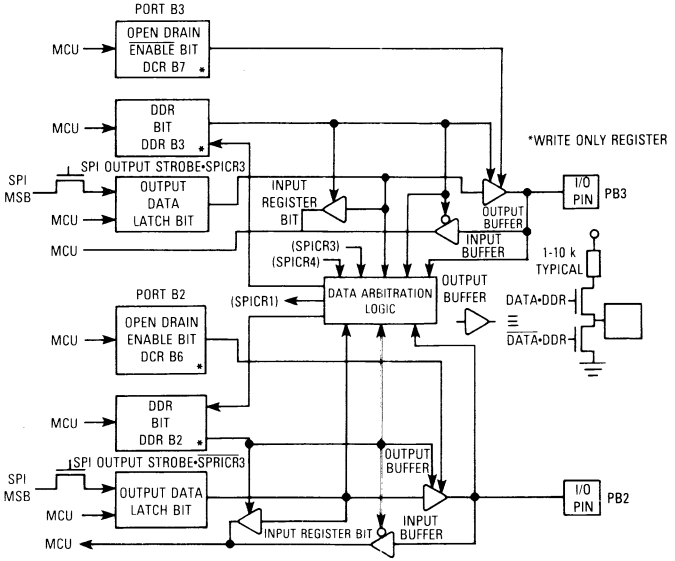


Figure 4. Port B Configuration

under the following conditions governed by control bits in SPI control register and SPI clock arbitration flip-flop status.

PB1 toggle enable = $(\overline{SPICR7})$•SPICR4• $(\overline{PB0} + DDRB0)$ + SPICR2• $\overline{SPICR4 \cdot CLAQ}$

where: SPICR7 = SPI interrupt request bit
SPICR4 = SPI operation enable bit
SPICR2 = port B1 toggle enable/start bit
CLAQ = clock arbitration flip-flop output

When PB1 toggle enable is asserted, the MCU write to PB1 data register is inhibited. When SPI is not used, SPICR4 and CLAQ are reset. Therefore, SPICR2 can directly control the port B1 toggle capability. Port toggle capability allows action on port B0 or B1 or both as a result of timer overflows. This method speeds up timer overflow to port service. A write to port B0 or B1 data registers is inhibited while the individual port toggle enable is asserted.

The port B DCR consists of four status bits (DCR4-DCR7) and four data direction bits (DCR0-DCR3). DCR4 is a toggle enable control bit for port B0. When cleared, the timer overflow pulse causes the data register on port B0 to toggle. Port A has an 8-bit and port C has a 2-bit wide data direction register.

3

MEMORY

The MCU is capable of addressing 4096 bytes of memory and I/O registers. The memory map is shown in Figure 5. The locations consist of user ROM, self-check ROM, user RAM, eight timer registers, a miscellaneous register, two A/D registers, two SPI registers, and I/O. The interrupt vectors are located from $FF8 to $FFF.

The stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer decrements during pushes and increments during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

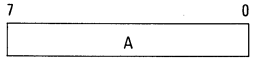
Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

REGISTERS

The MCU contains the registers described in the following paragraphs.

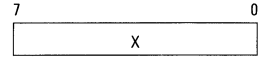
ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



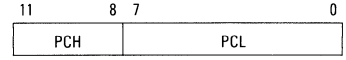
INDEX REGISTER (X)

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16-bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



PROGRAM COUNTER (PC)

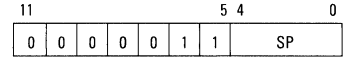
The program counter is a 12-bit register that contains the address of the next byte to be fetched.



STACK POINTER (SP)

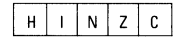
The stack pointer is a 12-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set at location $07F. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

The seven most-significant bits of the stack pointer are permanently set at 0000011. Subroutines and interrupts may be nested down to location $061 (31 bytes maximum), which allows the programmer to use up to 15 levels of subroutine calls (less if interrupts are allowed).



CONDITION CODE REGISTER (CC)

The condition code register is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.



Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, the timers (A, B, and C), the external ($\overline{INT1}$ and $\overline{INT2}$) interrupts, and the SPI interrupt are masked (disabled). If an interrupt occurs while this bit is set, the interrupt is latched and is processed as soon as the interrupt bit is cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic one).

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

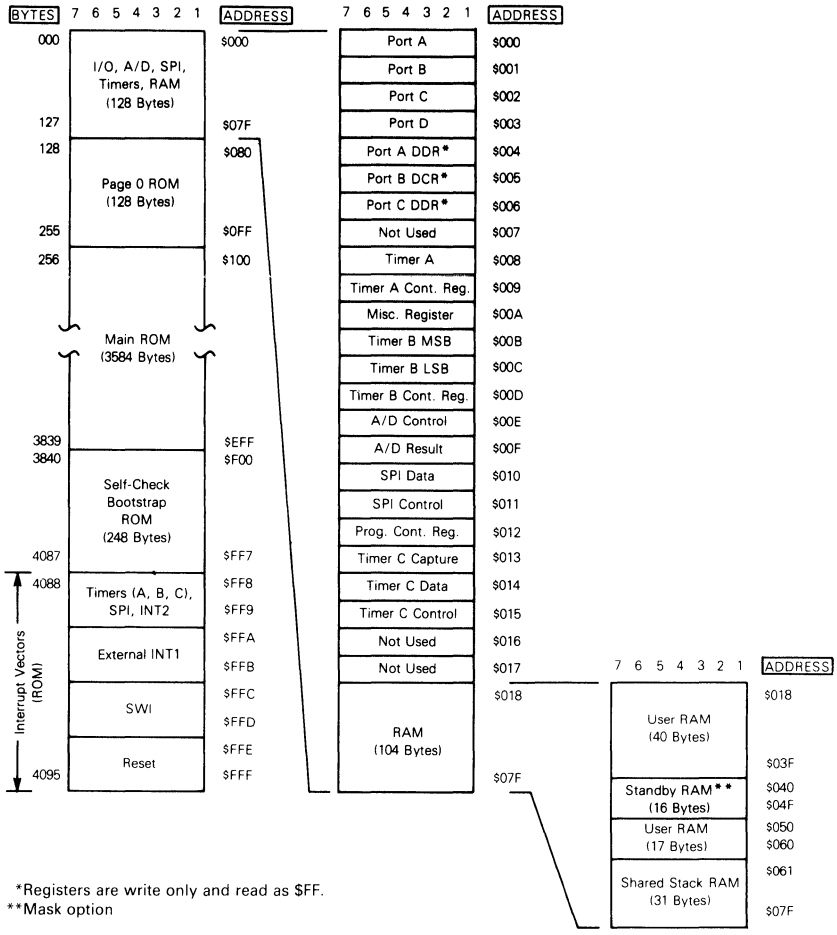


Figure 5. Memory Map

Carry/Borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions, and during shifts and rotates.

MISCELLANEOUS REGISTERS (MR) $0A

This register contains control and status information related to $\overline{\text{INT2}}$, auxiliary counter, prescalers 1 and 2, and timer overflow.

MR7	MR6	MR5	MR4	MR3	MR2	MR1	MR0
RESET:							
0	1	0	1			0	0

MR7 — $\overline{\text{INT2}}$ Interrupt Request Bit

If not masked by MR6, it causes an interrupt to the MCU, and if the I bit in the CCR is clear, the MCU will acknowledge the interrupt.

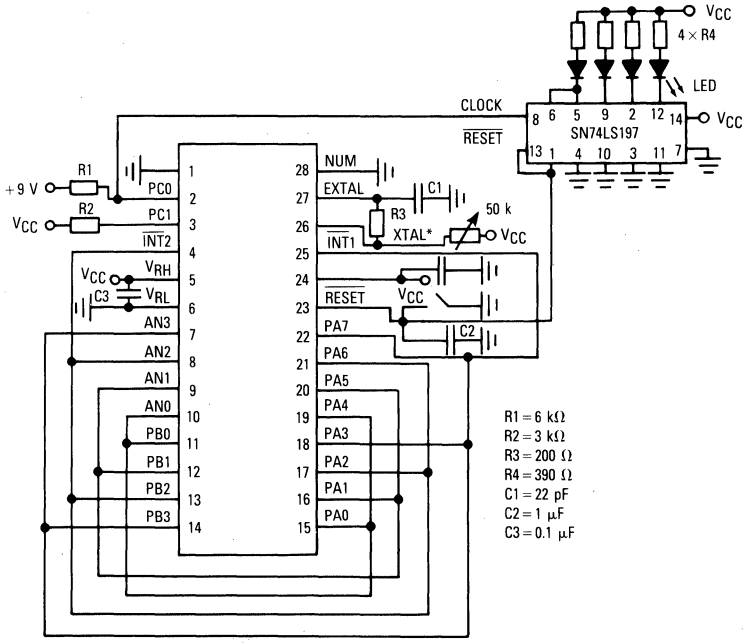
1 = Interrupt requested
0 = Interrupt not requested

MR6 — $\overline{INT2}$ Interrupt Request Mask
1 = Inhibits $\overline{INT2}$ interrupt request
0 = Does not inhibit $\overline{INT2}$ interrupt request
MR5 — Auxiliary Counter Status/Preset Bit
If not masked by MR4, it will drive a switch to V_{SS} on the $\overline{RESET}$ pin causing the MCU to reset. This bit may be used as an auxiliary counter preset bit. If MR5 is clear, a write of logic one will preset the auxiliary counter, and if set, a write of logic zero will preset the auxiliary counter.
1 = Auxiliary counter overflow
0 = Auxiliary counter clear
MR4 — Watchdog Control Bit
This bit cannot be set via software. The watchdog timer can only be disabled by reset.
1 = Watchdog timer disabled
0 = Watchdog timer enabled
MR3 — Prescaler 1 Clear Bit
Presets the contents of prescaler 1 to $7F.
1 = Prescaler 1 preset
0 = Prescaler 1 not preset
MR2 — Prescaler 2 Clear Bit
Presets the contents of prescaler 2 to $7FFF.
1 = Prescaler 2 preset
0 = Prescaler 2 not preset
MR1 — Prescaler Cross-Couple Bit
This bit controls the output of prescalers 1 and 2 and directs them to either timer A or B clock inputs.
1 = Prescaler 1 feeds timer B clock input, and prescaler 2 feeds timer A input
0 = Prescaler 1 output is used as clock input for timer A, and prescaler 2 output is used as clock input for timer B
MR0 — Port B Toggle Cross-Couple Bit
This bit controls the overflow pulses of timers A and B and directs them to either port B0 or B1.
1 = Timer A overflow output is directed to port B0, and timer B output is directed to port B1
0 = Overflow output pulse of timer A is used as a port B1 data register toggle clock source, and timer B overflow output pulse is directed to port B0 toggle clock input

SELF CHECK

The self check is initiated by connecting the MCU as shown in Figure 6 and then monitoring the output of port C (bit 0) for an oscillation of approximately 7 Hz. The self-check program exercises the CPU, I/O, RAM, ROM, timers, interrupts, analog-to-digital (A/D) converter, and the auxiliary counter.

The RAM, ROM, and 4-channel A/D test can be called by a user program. The timer test may be called if the timer input is the internal clock.



*RC Oscillator Option Shown. If Q0-Q2 LEDs Blinking = Device Passes Test
Q3 Blinking = Watchdog Reset Problem

Figure 6. Self-Check Connections

RESETS

The MCU can be reset four ways: (1) by initial power-up; (2) by the external reset input ($\overline{RESET}$); (3) by a forced reset generated by the "watchdog" counter; and (4) by an optional internal low voltage detect circuit. The $\overline{RESET}$ input consists mainly of a Schmitt trigger that senses the line logic level. Figure 7 shows the MCU reset circuit.

POWER-ON-RESET (POR)

An internal reset is generated on power-up that allows the internal clock generator to stabilize. The power-on reset is used strictly for power turn-on conditions and should not be used to detect any drop in the power supply voltage. A delay of t_{RHL} milliseconds is required before allowing $\overline{RESET}$ input to go high. Connecting a capacitor to the $\overline{RESET}$ input (Figure 8) typically provides sufficient delay.

EXTERNAL RESET INPUT

The MCU is reset when a logic zero is applied to the $\overline{RESET}$ input for a period longer than one machine cycle (t_{cyc}). Under this type of reset, the Schmitt trigger switches off at V_{IRES-} to provide an internal reset voltage.

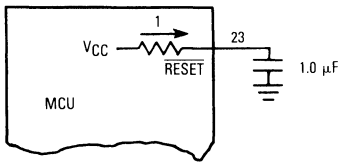


Figure 8. Power-Up Reset Delay Circuit

FORCED RESET

If the auxiliary counter reset mask bit in the miscellaneous counter (MR4) is cleared and the auxiliary counter status bit (MR5) is set, as a result of counter overflow, a switch to V_{SS} is turned on pulling the $\overline{RESET}$ pin low. A consequent voltage drop below V_{IRES-} on $\overline{RESET}$ causes a reset, which in turn sets MR4. Switching to V_{SS} when the $\overline{RESET}$ pin is turned off allows voltage to rise above V_{IRES+}, after which the reset is released. $\overline{RESET}$ pin voltage variation occurring as a result of forced reset may be amplified externally in order to provide a reset to other peripheral circuits in the system. The reset output from the MCU is not TTL compatible.

LOW-VOLTAGE INHIBIT (LVI)

The optional low-voltage detection circuit causes a reset of the MCU if the power supply voltage falls below a certain level (V_{LVI}). The only requirement is that the V_{CC} must remain at or below the V_{LVI} threshold for one t_{cyc} minimum.

In typical applications, the V_{CC} bus filter capacitor will eliminate negative-going voltage glitches of less than one t_{cyc}. The output from the low-voltage detector is connected directly to the internal reset circuitry. It also forces the $\overline{RESET}$ pin low via a strong discharge device through a resistor. The internal reset is removed once the power supply voltage rises above a recovery level (V_{LVR}) at which time a normal power-on reset occurs.

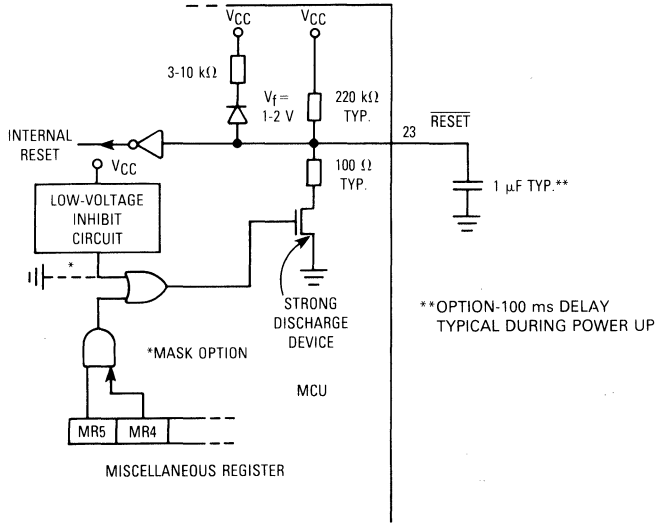
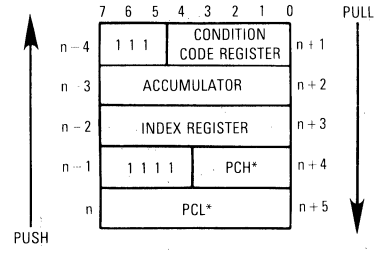


Figure 7. Reset Circuit

INTERRUPTS

The MCU can be interrupted seven different ways: through the external interrupt $\overline{INT1}$ input pin, with the internal timer (either A or B) interrupt request, using the software interrupt instruction (SWI), SPI interrupt request, external port D bit 6 ($\overline{INT2}$) input pin, or at reset.

Interrupts cause the processor registers to be saved on the stack and the interrupt mask (I bit) set to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack after which normal processing resumes. The stacking order is shown in Figure 9.



*For subroutine calls, only PCH and PCL are stacked.

Figure 9. Interrupt Stacking Order

Unlike $\overline{RESET}$, hardware interrupts do not cause the current instruction execution to be halted, but are considered pending until the current instruction is complete.

NOTE

The current instruction is considered to be the one already fetched and being operated on.

When the current instruction is complete, the processor checks all pending hardware interrupts and, if unmasked (I bit clear), proceeds with interrupt processing; otherwise, the next instruction is fetched and executed. Masked interrupts are latched for later interrupt service. If the timer interrupt status bit is cleared before unmasking the interrupt, then the interrupt is not latched.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction regardless of the setting of the I bit. Refer to Figure 10 for the reset and interrupt instruction processing sequence.

TIMER INTERRUPT

Each interrupt, except $\overline{INT1}$, has a separate mask bit which must also be cleared, in addition to the I bit, for the MCU to acknowledge the interrupt. The $\overline{INT2}$, timer A, timer B, timer C, and SPI interrupts each have their own independent mask bits contained in MR6, TACR6, TBCR6, TCOM, TCCM, and SPICR6. The interrupt routine must determine the source of the interrupt by examining the interrupt request bits, TACR7, TBCR7, MR7, TCOF, TCCF, and SPICR7. These bits must be cleared by software. The $\overline{INT1}$ interrupt has its own vector address. Therefore, the $\overline{INT1}$ interrupt request is cleared automatically, and then the $\overline{INT1}$ vector is serviced.

EXTERNAL INTERRUPT

The external interrupt is internally synchronized and then latched on the falling edge of $\overline{INT1}$ and $\overline{INT2}$. Clearing the I bit enables the external interrupt. The $\overline{INT2}$ interrupt has an interrupt request bit (bit 7) and a mask bit (bit 6) in the miscellaneous register (MR). The $\overline{INT2}$ interrupt is inhibited when the mask bit is set. The $\overline{INT2}$ is always read as a digital input on port D. The $\overline{INT2}$ and timer interrupt request bits, if set, cause the MCU to process an interrupt when the condition code I bit is clear. The following paragraphs describe two typical external interrupt circuits.

Zero-Crossing Interrupt

A sinusoidal input signal ($f_{\overline{INT1}}$ maximum) can be used to generate an external interrupt (see Figure 11a) for use as a zero-crossing detector (for negative transitions of the ac sinusoid). This type of circuit allows applications such as servicing time-of-day routines and engaging/disengaging ac power control devices. Off-chip, full-wave rectification provides an interrupt at every zero crossing of the ac signal and, thereby, provides a 2f clock.

Digital-Signal Interrupt

With this type of circuit (Figure 11b), the $\overline{INT1}$ pin can be driven by a digital signal. The maximum frequency of a signal that can be recognized by the TIMER or $\overline{INT1}$ pin logic is dependent on the parameter labeled t_{WL}, t_{WH}. Refer to **TIMER** for additional information.

SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. If the I bit is zero, SWI executes after the other interrupts. The SWI execution is similar to the hardware interrupts.

TIMERS

The MCU has four timers and two programmable prescalers. The timers are identified as timer A, B, C, and the auxiliary counter. Refer to Figure 12 for timers A, B, and C block diagram. The following paragraphs described the different timers.

TIMER A

Timer A is an 8-bit programmable counter, which can be loaded under program control. Timer A also includes a modulus latch which allows the timer to be "auto-reloaded." As clock inputs are received, timer A decrements toward $00. When $00 is reached, bit 7 in the timer A control register is set and the timer is reloaded with the contents of the modulus latch. An overflow condition is

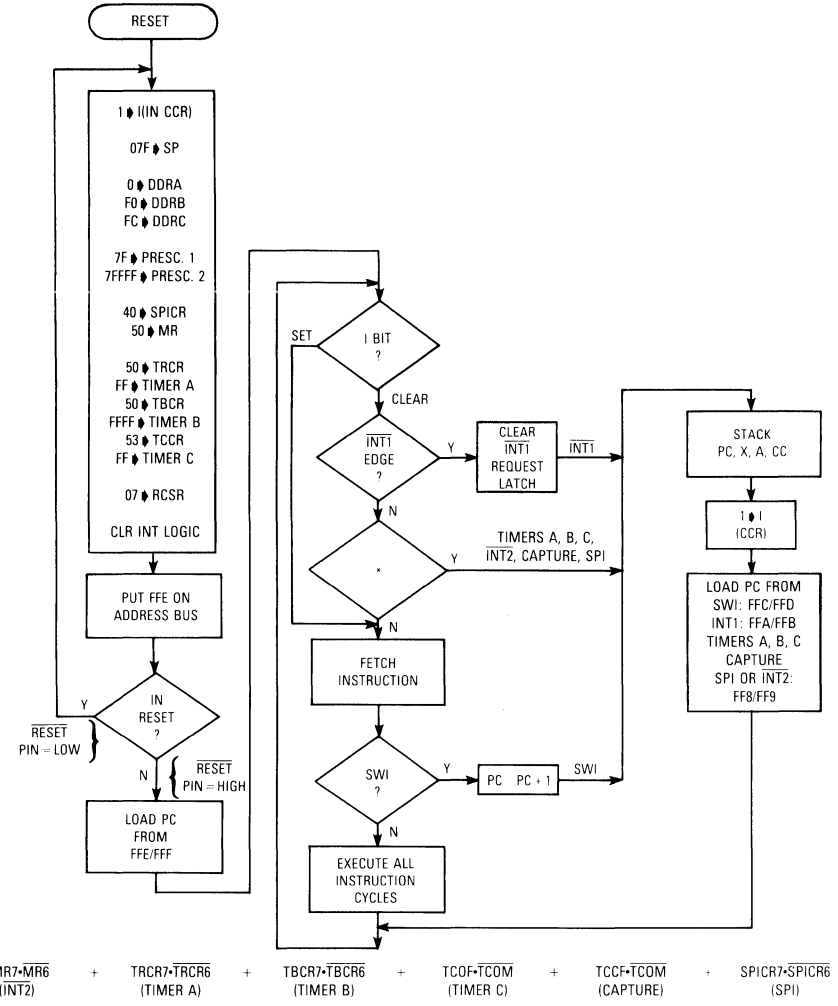


Figure 10. Reset and Interrupt Processing Flowchart

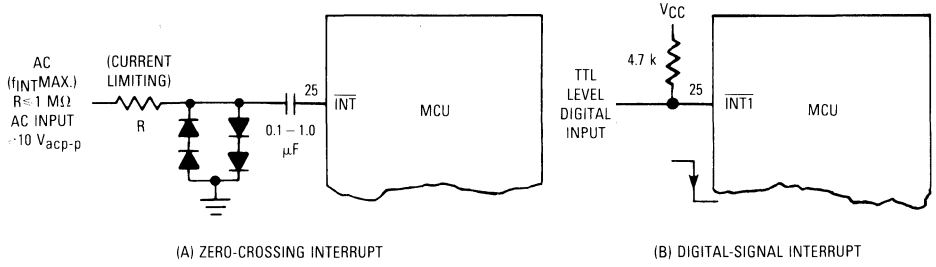


Figure 11. Typical Interrupt Circuits (INT1)

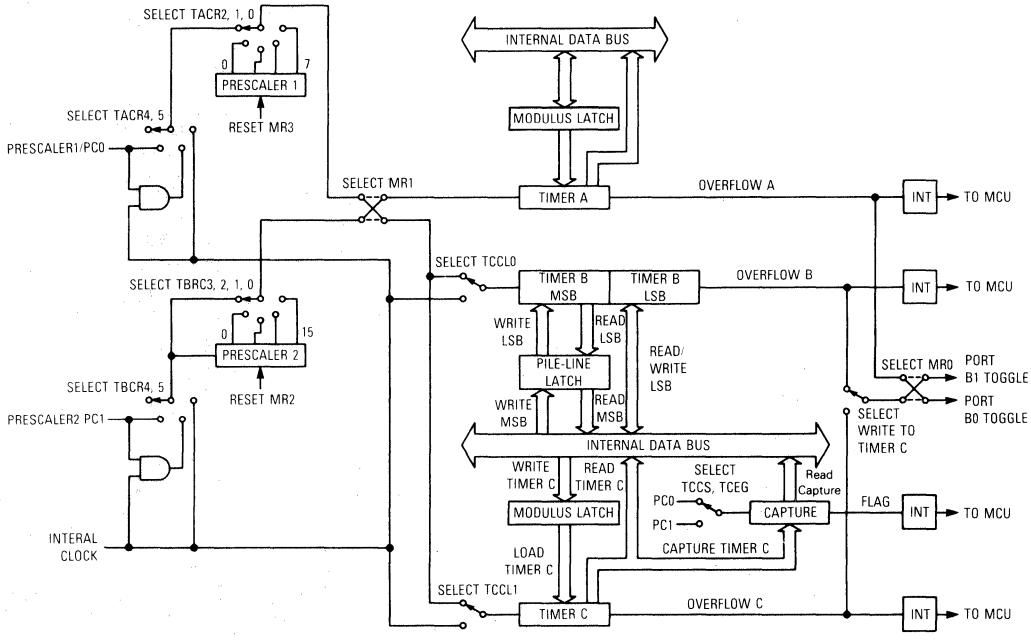


Figure 12. Timers A, B, and C Block Diagram

also generated when value $00 is reached. This state can be used to toggle bit 0 or bit 1 of port B directly under the control of the miscellaneous register (MR0), the SPI control register, and the port B data direction register. Setting TACR6 or the I bit in the condition control register will prevent timer interrupts from being processed. The timer interrupt request bit *MUST* be cleared by software. There are three ways of loading data from the modulus latch into timer A as described in the following paragrahs.

Direct Loading

When the MCU writes to timer A data register, the data is latched by the modulus latch, and forced into the timer. This operation requires that TACR3 be cleared.

Asynchronous External Event Loading

When TACR3 is a logic one, the contents of the modulus latch are transferred to the timer at the rising edge of $\overline{\text{INT2}}$ interrupt request bit (MR7) gated with interrupt request mask bit (MR6). If this loading is used, care must be taken in programming as it will start an interrupt service routine if the I bit in the CCR is clear. Loading $00 to timer A allows a countdown of 256 clocks before the next $00 state is reached.

Auto-Loading

The modulus latch is automatically loaded when the timer reaches $00. This loading is dependent on the setting of TACR3. Auto-loading also occurs in both the previous loading modes. Timer A can be read at any time without affecting the countdown of the timer. The timer and modulus latch are set to $FF on reset.

NOTE

Loading $01 to timer A should be avoided when operating with a divide-by-one prescaler. Doing so

will inhibit timer A auto-loading, interrupt generation, and port B toggle mechanisms.

TIMER A CONTROL REGISTER $09

7	6	5	4	3	2	1	0
TACR7	TACR6	TACR5	TACR4	TACR3	TACR2	TACR1	TACR0
RESET:							
0	1	0	0	0	0	0	0

TACR7 — Timer A Interrupt Request Flag
1 = Timer A has transition to $00
0 = Software or reset cleared
TACR6 — Timer A Interrupt Request Mask
1 = Interrupt request inhibited
0 = Interrupt request not inhibited
TACR5 — External or Internal Bit
1 = External clock source for prescaler 1
0 = Internal clock source for prescaler 1
TACR4 — External Enable Bit
Control bit used to enable the external timer pin (PRESCALER1/PC0).

TACR5	TACR4	Prescaler 1 Clock Source
0	0	Internal Clock
0	1	AND of Internal Clock and PRESCALER1/PC0*
1	0	Inputs Disabled
1	1	PRESCALER1/PC0* Low-to-High Transition

*The status of PRESCALER1/PC0 depends upon the data direction status of PRESCALER1/PC0. If PRESCALER1/PC0 is an output, then the clock source is equal to the port data register content, independent of the port electrical loading. If an input, then the clock source is the logic level of PRESCALER1/PC0.

TACR3 — Timer A Load Mode Control
1 = Asynchronous external event loading ($\overline{INT2}$ driven loading is enabled)
0 = Allows direct loading of timer A
TACR2, TACR1, TACR0 — Prescaler 1 Division Ratio Control Bits
When set, these bits select one of eight possible outputs on prescaler 1.

TACR2	TACR1	TACR0	Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	0	32
1	1	0	64
1	1	1	128

TIMER B

This is a 16-bit timer which is accessed via two registers ($0B for the most-significant byte (MSB) and $0C for the least-significant byte (LSB)). The MSB has a "pipeline" latch that allows a "snap shot" value of the entire 16 bits to be read. Read/write operations to the LSB are direct. The LSB can be read at anytime without disturbing the count. When the LSB is read, the contents of the MSB are loaded into the pipeline latch so a read of the MSB is actually the contents of the latch.

When writing to the LSB, the contents are immediately entered into the timer. At the same time the pipeline contents are forced into the MSB of the timer. This allows a 16-bit word to be placed into the timer data register during a LSB write operation. An underflow condition is also generated when value $00 is reached. This state can be used to toggle bit 0 or bit 1 of port B directly under the control of the miscellaneous register (MR0), the SPI control register, and the port B data direction register. Setting TBCR6 or the I bit in the condition control register will prevent timer interrupts from being processed. The timer interrupt request bit *MUST* be cleared by software.

TIMER B CONTROL AND STATUS REGISTER $0D

7	6	5	4	3	2	1	0
TBCR7	TBCR6	TBCR5	TBCR4	TBCR3	TBCR2	TBCR1	TBCR0
RESET:							
0	1	0	0	0	0	0	0

TBCR7 — Timer B Interrupt Request Flag
1 = Timer B has transition to $00
0 = Software or reset cleared
TBCR6 — Timer B Interrupt Request Mask
1 = Interrupt request inhibited
0 = Interrupt request not inhibited
TBCR5 — External or Internal Bit
1 = External clock source for prescaler 2
0 = Internal clock source for prescaler 2
TBCR4 — External Enable Bit
Control bit used to enable the external timer pin (PRESCALER2/PC1).

TBCR5	TBCR4	Prescaler 2 Clock Source
0	0	Internal Clock
0	1	AND of Internal Clock and PRESCALER2/PC1*
1	0	Inputs Disabled
1	1	PRESCALER2/PC1* Low-to-High Transition

*The status of PRESCALER2/PC1 depends upon the data direction status of PRESCALER2/PC1. If PRESCALER2/PC1 is an output, then the clock source is equal to the port data register content, independent of the port electrical loading. If an input, then the clock source is the logic level of PRESCALER2/PC1.

TBCR3, TBCR2, TBCR1, TBCR0 — Prescaler 2 Division Ratio Control Bits
When set, these bits select one of eight possible output on prescaler 2.

TBCR3	TBCR2	TBCR1	TBCR0	Divide By
0	0	0	0	1
0	0	0	1	2
0	0	1	0	4

— Continued —

TBCR3	TBCR2	TBCR1	TBCR0	Divide By
0	0	1	1	8
0	1	0	0	16
0	1	0	1	32
0	1	1	0	64
0	1	1	1	128
1	0	0	0	256
1	0	0	1	512
1	0	1	0	1024
1	0	1	1	2048
1	1	0	0	4096
1	1	0	1	8192
1	1	1	0	16384
1	1	1	1	32768

3

TIMER C

Timer C is an 8-bit programmable down counter. The timer contains a modulus latch which allows the timer to be auto reloaded. The timer auto reloads with the contents of the modulus latch upon every $01 to $00 transition. Timer C contains a capture register. This read-only register and the contents are refreshed by the contents of the data register during the capture instance. The timer can be written to at any time, and the contents of both the data register and modulus latch are updated immediately. The timer is set to $FF on reset, but the contents of the capture register are not valid until the first capture after reset.

TIMER C CONTROL REGISTER $015

7	6	5	4	3	2	1	0
TCOF	TCOM	TCCF	TCCM	TCEG	TCCS	TCCL1	TCCL0
RESET: 0	1	0	0	0	0	0	0

TCOF — Timer C Overflow Flag
- 1 = Timer C has transition to $00
- 0 = Software or reset cleared

TCOM — Timer C Interrupt Mask
- 1 = Interrupt request inhibited
- 0 = Interrupt request not inhibited

TCCF — Timer C Capture Flag
- 1 = Proper capture occurred on PRESCALER1 or PRESCALER2. No new capture occurs when set
- 0 = Software or reset cleared

TCCM — Timer C Capture Interrupt Request Mask
- 1 = Inhibits interrupt request generated from TCCF
- 0 = Does not inhibit interrupt request generated from TCCF

TCEG — Timer C Capture Edge Select
- 1 = Selects rising edge of PC0or PC1 to be capture instance
- 0 = Selects falling edge of PC0 or PC1 to be capture instance

TCCS — Timer C Capture Source Select
- 1 = Select PRESCALER2/PC1 as capture source
- 0 = Select PRESCALER1/PC0 as capture source

TCCL1 and TCCL0 — Timer C Clock Source Select
Clock source selection is defined below.

TCCL1	Timer C Source
0	Internal Clock
0	Internal Clock
1	MR1 Status*
1	MR1 Status*

TCCL0	Timer B Source
0	Internal Clock
1	MR1 Status*
0	Internal Clock
1	MR1 Status*

NOTES:
1. *Denotes prescaler 1 or 2 clock source depending on miscellaneous register bit 1 (MR1) status.
2. MR1 bit cleared (logic zero) at reset:
   Prescaler 1 clock selected to timer A
   Prescaler 2 clock selected to timer B and C
3. MR1 bit set (logic one):
   Prescaler 1 clock selected to timer B and C
   Prescaler 2 clock selected to timer A
4. Prescaler 1 output determined by the status of Timer A control register bits 2, 1, and 0 (TACR2, TACR1, and TACR0)
5. Prescaler 2 output determined by the status of Timer B control register bits 3, 2, 1, and 0 (TBCR3, TBCR2, TBCR1, and TBCR0)

PRESCALER 1

Prescaler 1 is a 7-bit binary down counter whose value is selected by TACR2, TACR1, and TACR0. The selected output is used as the clock input to either timer A or B, depending upon the status of the prescaler cross-couple bit (MR1). The type of clock source to prescaler 1 may be selected by TACR5 and TACR4. Prescaler 1 is set to $7F at reset or under program control when a one is written to prescaler 1 clear bit (MR3).

PRESCALER 2

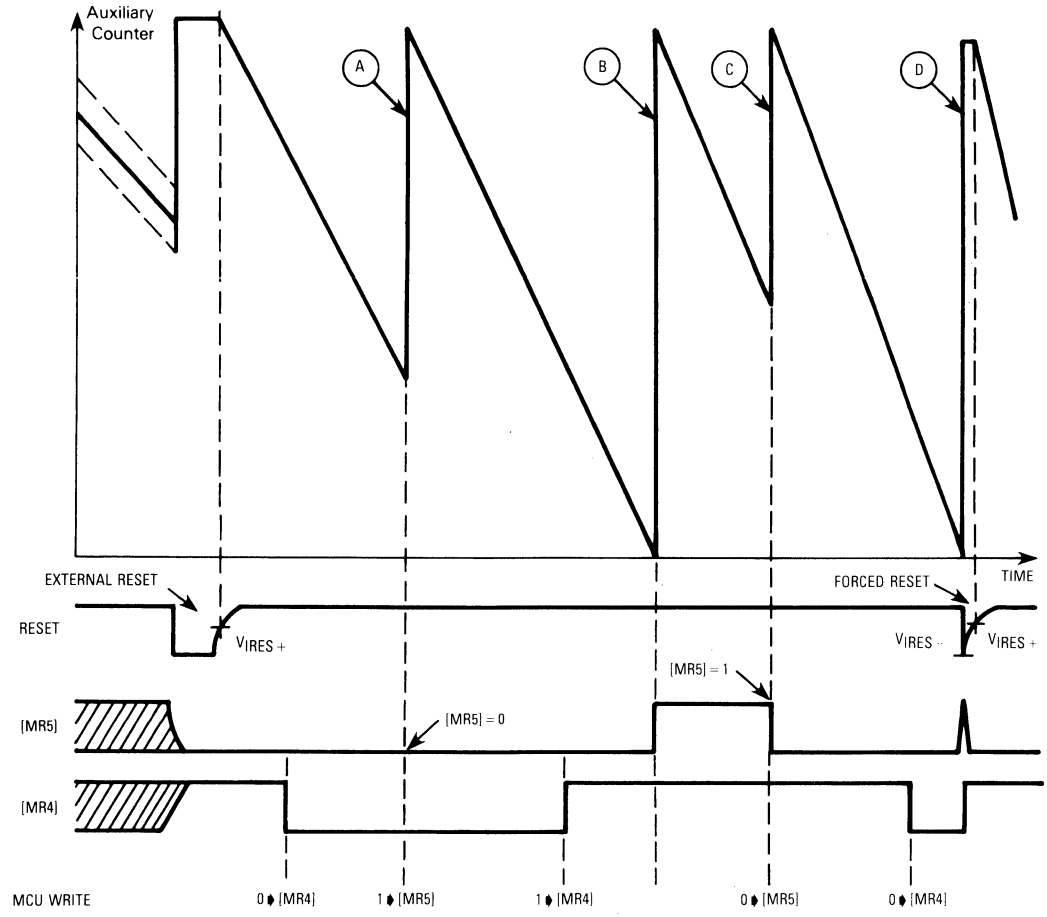
Prescaler 2 is a 15-bit down counter; its value is selected by TBCR3, TBCR2, TBCR1, and TBCR0. The selected output is used as the clock input to either timer A or B, depending upon the status of the prescaler cross-couple bit (MR1). The type of clock source to prescaler 2 may be selected by TBCR5 and TBCR4. Prescaler 2 is set to $7FFF at reset or under program control when a one is written to prescaler 2 clear bit (MR2).

AUXILIARY COUNTER

This register is a fixed counter which is clocked by the internal clock (f_{OSC} divided by four). Total count period is 4095 cycles. The MCU communicates with this counter via the miscellaneous register (MR5 and MR4). Countdown may be aborted at any time under program control, which also resets the counter to 4095 and clears MR5. When MR4 is clear and MR5 is set as a result of counter time out, the reset pin is internally pulled to ground. If the MCU loses control of the program, the "watchdog" timer will bring the MCU back to reset. Refer to Figure 13 for counter operation diagram.

SERIAL PERIPHERAL INTERFACE

The serial peripheral interface (SPI) has arbitration on the data and clock lines. The SPI communicates with the MCU via data and control registers. The SPI data and



(A) Counter Preset by Writing "1"

(B) Underflow: MR5 ◗ 1; No Forced Reset

(C) Counter Reset by Writing "0"

(D) Underflow MR5 ◗ 1 Forced Reset

Figure 13. Auxiliary Counter Operation

clock inputs are always taken from their respective I/O ports, regardless of the status of the data direction registers relative to that port. The SPI can operate in modes from auto clocked (NRZ), half duplex, and full duplex with from a one to a four wire combination. Refer to Figure 14 for the SPI block diagram.

SPI CONTROL AND STATUS REGISTER

This 8-bit register contains the status and control bits relative to SPI operations. The SPI control register operation is shown in Figure 15. The SPI control and status register bits can be set or cleared under program control.

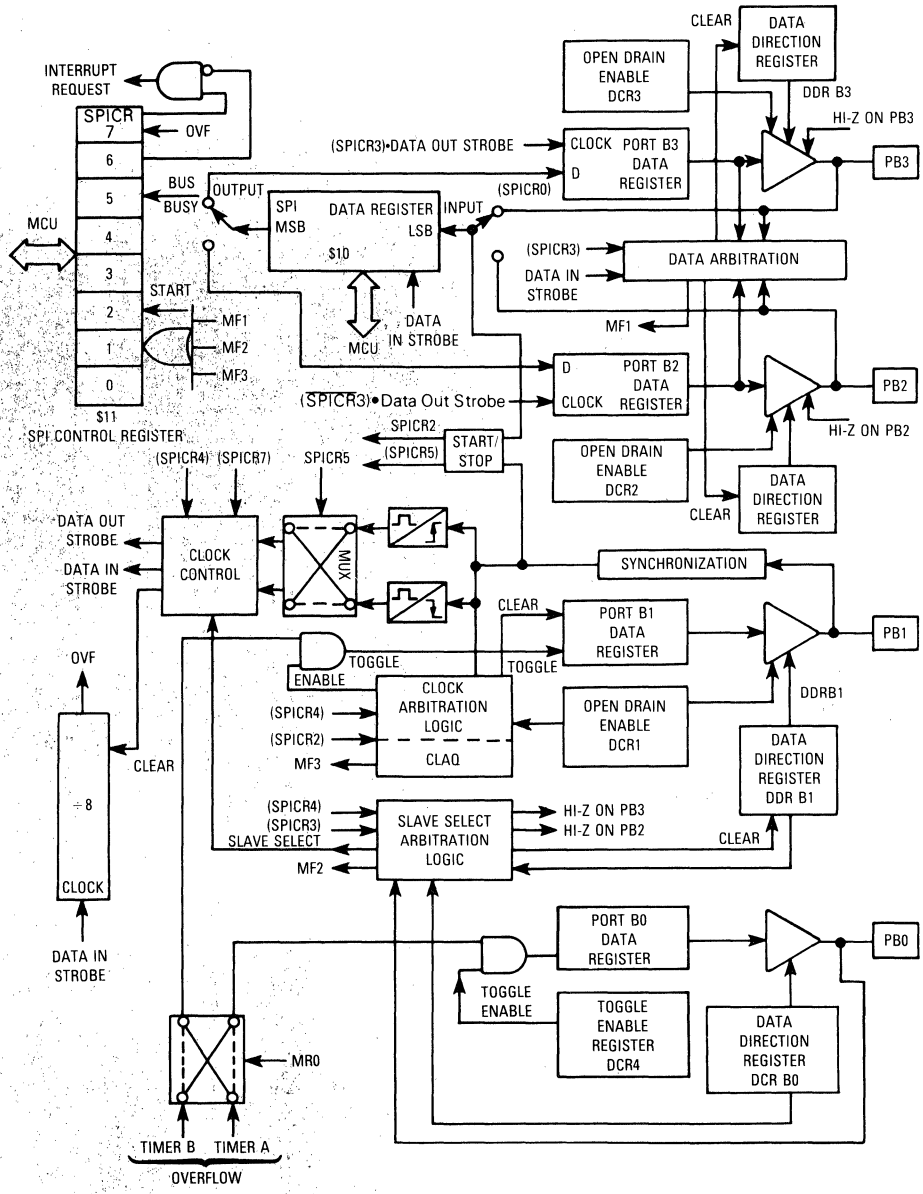


Figure 14. Serial Peripheral Interface Block Diagram

7	6	5	4	3	2	1	0
SPICR7	SPICR6	SPICR5	SPICR4	SPICR3	SPICR2	SPICR1	SPICR0
RESET:							
0	1	0	0	0	0	0	0

SPICR7 — SPI Interrupt Request Bit
Set on eighth data input strobe. MCU services this interrupt if I bit is clear in CCR.
1 = Interrupt request (if SPICR6 not masked)
0 = No interrupt pending

SPICR6 — SPI Interrupt Request Mask Bit
1 = Disables interrupt request from SPICR7
0 = Enables interrupt request from SPICR7

SPICR5 — SPI Clock Sense Bit/Bus-Busy Flag
Dual-function bit controlled by the status of SPICR4.
1 = Start SPI operation when SPICR4 = 1. Input data latched on positive edge and output data changed on negative edge of SPI clock when SPICR4 = 0.
0 = Stop SPI operation when SPICR4 = 1. Input data latched on negative edge and output data changed on positive edge of SPI clock when SPICR4 = 0.

SPICR4 — SPI Operation Enable Bit
This bit determines the functions of SPICR5 and SPICR2.
1 = Enables SPI data register shifting, data and clock arbitration logic, and slave select input logic
0 = Disables SPI data register shifting, data and clock arbitration logic, and slave select input logic

SPICR3 — SPI Data Output Select Bit
1 = Output of the SPI data register is loaded to port B3 data register at the appropriate SPI clock edge selected by SPICR5, during the active transaction mode
0 = Output of the SPI data register is loaded to port B2 data register at the appropriate SPI clock edge selected by SPICR5, during the active transaction mode

SPICR2 — Mode Fault Flag
Dual-function bit controlled by the status of SPICR4.
1 = Start bit is set by negative transition of the data input of the SPI data shift register while the clock is at the idle level when SPICR4 = 1. Start bit set under program control to enable port B1 data register toggle facility when SPICR4 = 0.
0 = Stop SPI operation when SPICR4 = 1. Cleared under program control when SPICR4 = 0.

SPICR1 — Mode Fault Flag
1 = (a) Mode flag is set when SPI data output arbitration occurs on the SPI data output port (PB3 or PB2) selected by SPICR3. The MCU loses data mastership, and the SPI data output port DDR is cleared.
(b) Mode flag is set if a low level is detected on slave input PB0. Then, the MCU loses clock mastership switching to the clock slave mode, and port B1 DDR is cleared.
(c) Mode flag is set during the idle mode when a negative clock edge is detected on the SPI clock input, and the port B1 data register is cleared.
0 = Cleared under program control

SPICR0 — SPI Input Data Select Bit
1 = SPI data from port B3 is latched into the SPI data register
0 = SPI data from port B2 is routed to the input of the SPI data register

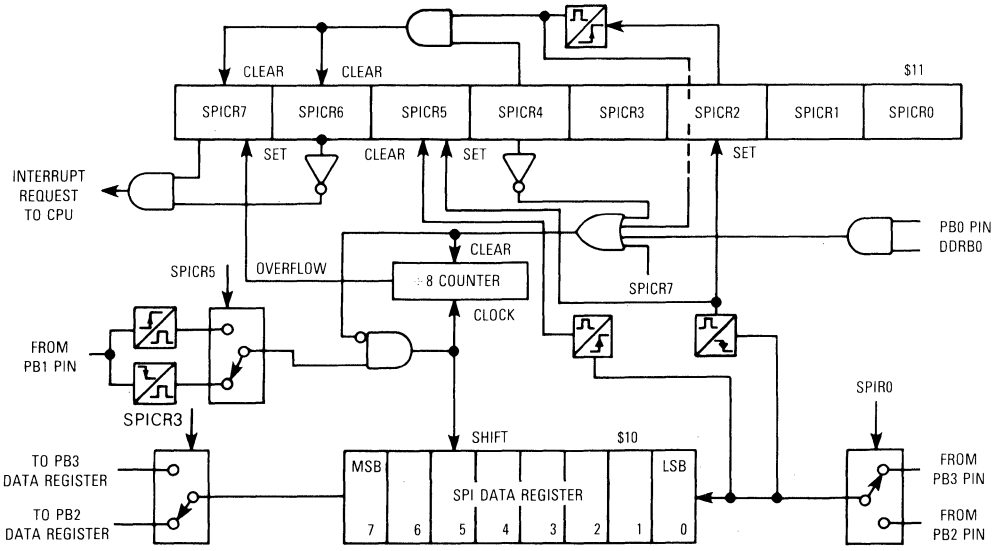


Figure 15. SPI Control Register Operation

SPI DATA REGISTER

This register can be written to any time and can also be read, regardless of serial operations, without disturbing the data. A one bit shift to the left occurs each time there is a data input strobe while the LSB is loaded with data from port B2 or B3. The MSB is loaded every time there is data output strobe. Data input and output strobes are generated internally only during the active transaction time.

SPI DIVIDE-BY-EIGHT COUNTER

The counter is cleared during SPI deselect or idle modes. A count occurs at every data input strobe during the active transaction mode. At overflow, SPICR7 is set which puts the SPI in idle mode and blocks all data input and output strobes. The counter is cleared when PB0 is high if the SPI is in the slave mode or when a "start" condition is detected.

3

SPI OPERATION

The SPI can operate in a variety of modes. Software assisted protocols may be defined to upgrade the hardware versatility and/or system performance of the MCU. Some features common to all operating modes are summarized in Table 1 and in the following paragraphs.

1) SPI data input and output may be individually routed to or from PB2 or PB3 (Table 2). These four routings provide half and full duplex operations, as well as allowing bidirectional information to flow in daisy-chained systems.
2) When data input and output is done on PB2, PB3 is available for any other use and vice versa.
3) Data input is always relative to the port pin logic level regardless of the data direction register status on that pin.
4) In full duplex operation, 16 bits of information may be transferred with eight clock pulses between at least two devices with transmit capability. Both PB2 and PB3 are used for data transfer. The same shift register is used for data in and data out. The byte transmitted replaces the byte received. SPICR7 is used to signify that the I/O operation is complete.
5) SPI clock is always provided on port B1. In the clock slave mode, port B1 DDR is in the input mode (cleared). In the clock master mode, port B1 DDR is set; therefore, the MCU imposes the clock level on PB1 until there is clock arbitration on the clock line or until the MCU loses clock mastership when PB0 goes low.
6) No fixed baud rate generation exists. The clock frequency is dependent on the prescaler clock source option, prescaler divide ratio, and timer divide ratio as well as the port C status in case of external clocking for the timer. Toggling of the port B1 data register is automatically allowed during the active transmission mode.
7) All devices connected to the SPI must have their output and input data strobe on the same clock edge for correct transfer of data.
8) During the active transmission mode, the first clock edge must be the output data strobe. When this occurs, the MSBs of the data registers of all transmitters are copied onto the data output pins, and the MCU copies the MSB of its SPI onto the port B2 or B3 data register.
9) Port B data direction registers and port B data control registers are accessible during SPI operation. During active transaction mode, the PB1 data register, PB2 data register (if SPICR3 = 0), and PB3 (if SPICR3 = 1) are not write accessible under program control.
10) Port B lines not used for SPI can be used for other digital functions.

SELECT INPUT OPERATION

An external device supplies slave select information via port B0. If slave select is not used, set port B0 to output mode to inhibit slave select function.

The following paragraphs describe clock master and clock slave operating modes of the SPI.

Master Mode Slave Select Actions

The MCU monitors slave select input in master mode to assure that it stays false. If slave select goes true, the MCU exits master mode and becomes a slave. This implies that a write collision has occurred which means two devices attempted to become masters. Write collisions normally result from a software error, and the default master must clean up the system. The mode fault flag is set to signal that clock mastership is lost. Slave select actions can take place during either active or idle transaction modes.

Slave Select Input Actions During Slave Mode

The current clock master generates slave select to enable one of several slaves to accept or return data. The $\overline{SS}$ signal must go low before serial clock pulses occur and must remain low until after the eighth serial clock cycle. Individual lines or a daisy chain can be used for multiple slaves. When $\overline{SS}$ is high, the following occur:

- Serial data output is forced to a high-impedance state without affecting the DDR status.
- Serial clock input pulses are inhibited from generating internal data output and input strobe pulses.
- The eight-bit counter is cleared.

SPI OPERATING MODES

Six methods of operating the SPI are discussed in the following paragraphs.

One-Wire Autoclocked Mode

Various SPI devices can be connected on a single wire, with data transmission using an implicit clock, and each device being its own clock master.

Two-Wire Half-Duplex Mode

In this mode, separate data and clock lines connect the elements in the system. Data and clock mastership should be monitored via protocol included in the data patterns. A transmitter can send all zeros to take all other transmitters off the bus.

Table 1. Summary of SPI Operations

DEFINITIONS
- Transmitter — Data Master: DDRB2 or 3 = 1
- Receiver — Data Slave: DDRB2 or 3 = 0
- Clock Master: DDRB1 = 1
- Clock Slave: DDRB1 = 0
- Transaction Mode: SPICR4 = 1
  1) Active: $\overline{SPICR7}\bullet(\overline{DDRB0}\bullet\overline{PB0} + DDRB0)$ if DDRB1 = 0 (clock slave mode) or
     $SPICR7\bullet(\overline{DDRB0}\bullet\overline{PB0} + DDRB0)$ if DDRB1 = 1 (clock master mode)
     Clock Pulses allowed, data shifted
  2) Idle: $SPICR7 + \overline{DDRB0}\bullet PB0$ if DDRB1 = 0 (clock slave mode)
     Clock pulses blocked, data output line in high-impedance state
- Deselect Mode: SPICR4 = 0 – No SPI Operations

SLAVE SELECT INPUT
- Slave Select Input: SPISS – PB0
  - If DDRB0 = 0 then so SPISS action on MCU
    1) Master Mode: SPISS = 1 DDRB1 = 1
       SPISS 1 – 0: Switch to Slave Mode (DDRB1 1 – 0)
       Set SPICR1 (Mode Fault Flag)
    2) Slave Mode: SPISS = 0 DDRB1 = 0
       External clock is allowed to shift data in/out. If SPISS is pulled high, the external clock input pulses are inhibited; no data shift; divide-by-eight counter cleared; SPID (PB2 or PB3) switched to high-impedance state.
  - Used as Chip-Select Input

DATA ARBITRATION

Data master loses data mastership when data collision occurs during internal data strobe time.

If SPID output port (PB2 or PB3) = 1 while actual pin level is pulled low externally — conflict detected at internal data strobe time.

Then SPICR1 (mode fault flag) is set; SPID output port DDR (B2 or B3) 1 ➧ 0 (high-impedance state).

CLOCK ARBITRATION

MCU has clock mastership (DDRB1 = 1)
1) Via SPISS line (DDRB0 = 0). If SPISS is pulled low, then clock mastership lost; DDRB1 1 ➧ 0 (high-impedance state); SPICR1 is set (mode fault flag).
2) Via clock line SPICL (DDRB1 = 1 and DCRB5 = 0)
   Condition: SPICL must have open-drain output (DCRB5 = 0)

   If clock line is held low externally then clock mastership is not lost; minimum t_{CLH} and t_{CLK} times are guaranteed.

   If SPICL goes low during idle mode then SPICR1 = 1 and clock line is switched low to inhibit the system clock.

MODE FAULT FLAGE OPERATION (SPICR1)

Flag set when any of the following conditions occur:
- Data arbitration occurs on SPID output.
- Clock arbitration with SPISS during master to slave switching.
- Clock arbitration via clock line if SPICL 1 ➧ 0 during idle.

START, STOP, AND CLOCK IDLE CONDITIONS

Clock Idle: The clock level just prior to the transition that causes data on the serial output data line to be changed is defined as the SPI clock idle state.
SPICR5 = 0: SPICL Idle = Low State
SPICR5 = 1: SPICL Idle = High State

These definitions are necessary for determining start and stop conditions.

NOTE

Clock idle state can only be defined if SPICR4 = 0 (Deselect Mode)

Start Condition: Any negative transition of the data input line (PB2 or PB3) during an SPICL idle state.
Stop Condition: Any positive transition of the data input line during an SPICL idle state.

3

Table 2. Port B Status During SPI Operation

Port Name	Use	Input	Output	Comments
PB0	SPISS	Yes	No	Used as slave select input
PB0	Data	No	Yes	Used as "busy" signal or any digital output
PB1	SPICL	Yes	No	Clock slave
PB1	SPICL	No	Yes	Clock master
PB2	SPID	Yes	No	SPI data input SPICR0 = 0
PB2	SPID	No	Yes	SPI data output SPICR3 = 0
PB2	Data	Yes	Yes	Any digital signal SPICR3 = 1
PB3	SPID	Yes	No	SPI data input SPICR0 = 1
PB3	SPID	No	Yes	SPI data output SPICR3 = 1
PB3	Data	Yes	Yes	Any digital signal SPICR3 = 0

3

Three-Wire Half-Duplex Mode with Slave Select Input

This mode is the same as the half-duplex mode except that the slave select input allows using the MCU as a peripheral in a system where clock mastership is passed through the slave select line. Typically, the slave select lines can be wired together. The current master sets its slave select line in the output mode prior to a serial transmission and pulls it low to indicate that the system is busy. This allows the clock master to retain mastership until the end of transmission. Software protocol can be arranged so that slaves do not request mastership until their slave select lines go high. At the end of a transmission, the current master pulls $\overline{\text{SPISS}}$ high and puts the $\overline{\text{SPISS}}$ port (PB0) in the input mode. A slave requesting clock mastership pulls the $\overline{\text{SPISS}}$ line low, removing the current master from the line. Time multiplexed protocols may be required to avoid simultaneous mastership requests.

Three-Wire Full-Duplex Mode

This mode allows the MCU to operate simultaneously as transmitter and receiver. Bus or daisy-chain networks are feasible. Protocols in the data stream are required to change:

- Clock masters
- The number of transmitters in the system
- The direction of data flow in daisy-chained systems with collision

It is possible for the MCU to shift out one byte of data while receiving another, as illustrated in Figure 16. This eliminates the need for XMIT EMPTY or REC FULL status bits.

Three-Wire Full-Duplex Mode with Clock Arbitration

This mode is a mix of the three-wire full-duplex mode and two-wire half-duplex mode with clock arbitration, where the SPI clock line operates as a wire-or. Simultaneous masters are allowed, and clock arbitration is via the clock line.

Four-Wire Full-Duplex Mode with Slave-Select Input

This mode is similar to the three-wire full-duplex mode in network construction and to the three-wire half-duplex mode with slave-select input in clock arbitration and slave selection. Refer to Figure 17.

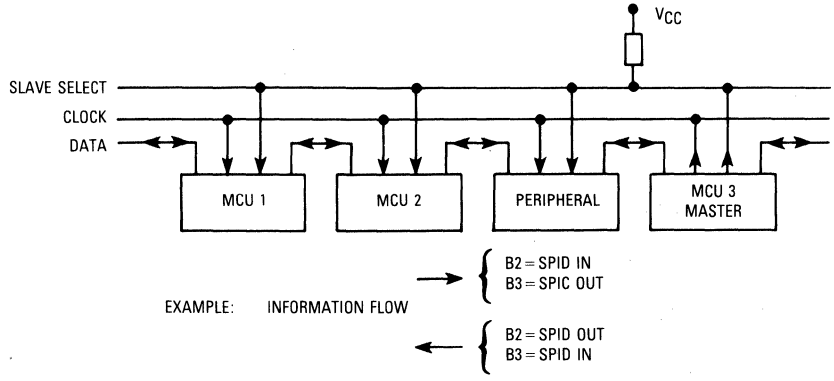


Figure 16. Daisy Chain/Cascade Organization

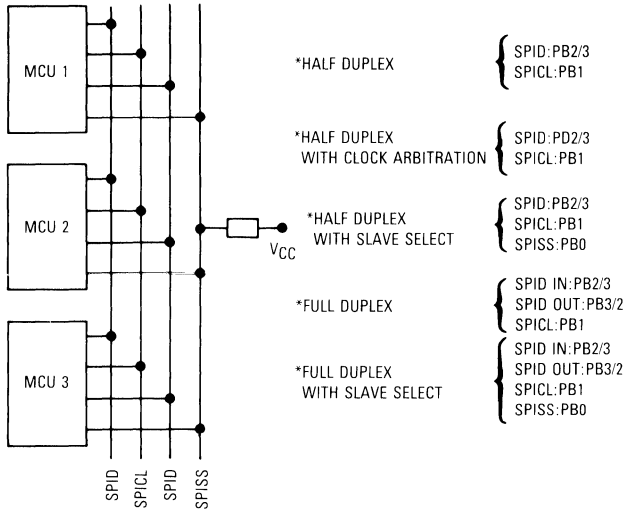


Figure 17. SPI Operation Bus Organization

3

ANALOG-TO-DIGITAL CONVERTER

The chip resident 8-bit analog-to-digital (A/D) converter uses a successive approximation technique as shown in Figure 18. Four external analog inputs can be connected to the A/D through a multiplexer via Port D. Four internal analog channels ($V_{RH} - V_{RL}$, $V_{RH} - V_{RL}/2$, $V_{RH} - V_{RL}/4$, and V_{RL}) may be selected for calibration. The accuracy of these internal channels may not meet the accuracy specifications of the external channels.

A fifth external analog input (AN4) is available via mask option. When selected, it replaces the V_{RH} internal channel. Due to signal routing, the accuracy of this fifth channel may be slightly less than AN0-AN3.

Multiplexer selection is controlled by the A/D control register (ACR) bits 0, 1, and 2. Refer to Table 3 for multiplexer selection. The ACR is shown in Figure 18. The converter uses 30 machine cycles to complete a conversion of a sampled analog input. When the conversion is complete, the digital value is placed in the A/D result register (ARR); the conversion flag is set; selected input is sampled again; and a new conversion begins. When ACR7 is cleared, the conversion in progress is aborted and the selected input, which is held internally, is sampled for five machine cycles.

The converter uses V_{RH} and V_{RL} as reference voltages. An input voltage equal to or greater than V_{RH} converts to $FF. An input voltage equal to or less than V_{RL}, but greater than V_{SS}, converts to $00. Maximum and minimum ratings must not be exceeded. Each analog input source should use V_{RH} as the supply voltage and be referenced to V_{RL} for the ratiometric conversion. To maintain full accuracy of the A/D, three requirements should be followed: (1) V_{RH} should be equal to or less than V_{CC}, (2) V_{RL} should be equal to or greater than V_{SS} but less than maximum specifications, and (3) $V_{RH} - V_{RL}$ should be equal to or greater than 4 volts.

Table 3. A/D Input MUX Selection

A/D Control Register			Input Selected	A/D Output (Hex)		
ACR2	ACR1	ACR0		Min	Typ	Max
0	0	0	AN0			
0	0	1	AN1			
0	1	0	AN2			
0	1	1	AN3			
1	0	0	V_{RH}**	FE**	FF**	FF**
1	0	1	V_{RL}*	00	00	01
1	1	0	$V_{RH}/4$*	3F	40	41
1	1	1	$V_{RH}/2$*	7F	80	81

*Internal (calibration) levels

**AN4 may replace the V_{RH} calibration channel if selected via mask option.

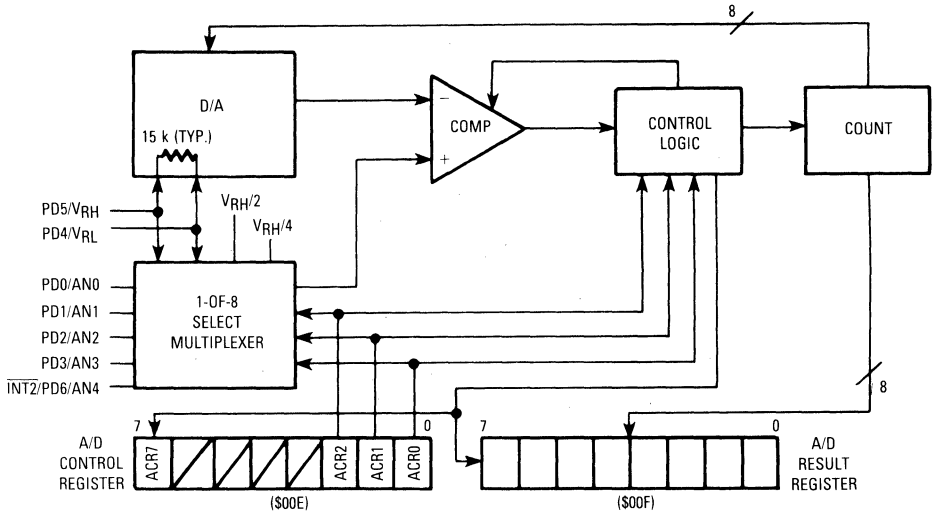


Figure 18. A/D Block Diagram

The A/D has a built-in 1/2 LSB offset intended to reduce the magnitude of the quantizing error to ±1/2 LSB, rather than +0, −1 LSB with no offset. This implies that, ignoring errors, the transition point from $00 to $01 occurs at 1/2 LSB above V_{RL}. Similarly, the transition from $FE to $FF occurs 1-1/2 LSB below V_{RH}, ideally.

INSTRUCTION SET

The MCU has a set of 59 basic instructions which can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the first 256 bytes of the memory space where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear and bit test, and branch functions are all implemented with a single instruction. For test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch IFF Bit n is Set	BRSET n (n=0 . . . 7)
Branch IFF Bit n is Clear	BRCLR n (n=0 . . . 7)
Set Bit n	BSET n (n=0 . . . 7)
Clear Bit n	BCLR n (n=0 . . . 7)

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following instruction list.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified

value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (2's Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch IFF Higher	BHI
Branch IFF Lower or Same	BLS
Branch IFF Carry Clear	BCC
(Branch IFF Higher or Same)	(BHS)
Branch IFF Carry Set	BCS
(Branch IFF Lower)	(BLO)
Branch IFF Not Equal	BNE
Branch IFF Equal	BEQ
Branch IFF Half Carry Clear	BHCC
Branch IFF Half Carry Set	BHCS
Branch IFF Plus	BPL
Branch IFF Minus	BMI
Branch IFF Interrupt Mask Bit is Clear	BMC
Branch IFF Interrupt Mask Bit is Set	BMS
Branch IFF Interrupt Line is Low	BIL
Branch IFF Interrupt Line is High	BIH
Branch to Subroutine	BSR

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No-Operation	NOP

OPCODE MAP SUMMARY

Table 4 is an opcode map for the instructions used on the MCU.

3

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single-byte instructions, while the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. Two-byte direct-addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction.

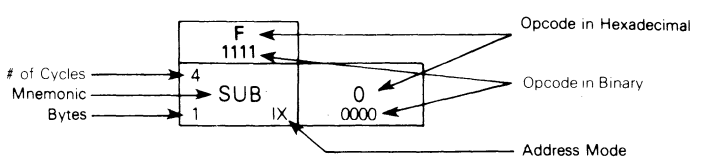
Table 4. Opcode Map

Hi / Low	Bit Manipulation BTB 0 0000	Bit Manipulation BSC 1 0001	Branch REL 2 0010	Read-Modify-Write DIR 3 0011	Read-Modify-Write INH 4 0100	Read-Modify-Write INH 5 0101	Read-Modify-Write IX1 6 0110	Read-Modify-Write IX 7 0111	Control INH 8 1000	Control INH 9 1001	Register/Memory IMM A 1010	Register/Memory DIR B 1011	Register/Memory EXT C 1100	Register/Memory IX2 D 1101	Register/Memory IX1 E 1110	Register/Memory IX F 1111	Hi / Low
0 0000	10 BRSET0 3 BTB	7 BSET0 2 BSC	4 BRA 2 REL	6 NEG 2 DIR	4 NEG 1 INH	4 NEG 1 INH	7 NEG 2 IX1	6 NEG 1 IX	9 RTI 1 INH		2 SUB 2 IMM	4 SUB 2 DIR	5 SUB 3 EXT	6 SUB 3 IX2	5 SUB 2 IX1	4 SUB 1 IX	0 0000
1 0001	10 BRCLR0 3 BTB	7 BCLR0 2 BSC	4 BRN 2 REL						6 RTS 1 INH		2 CMP 2 IMM	4 CMP 2 DIR	5 CMP 3 EXT	6 CMP 3 IX2	5 CMP 2 IX1	4 CMP 1 IX	1 0001
2 0010	10 BRSET1 3 BTB	7 BSET1 2 BSC	4 BHI 2 REL								2 SBC 2 IMM	4 SBC 2 DIR	5 SBC 3 EXT	6 SBC 3 IX2	5 SBC 2 IX1	4 SBC 1 IX	2 0010
3 0011	10 BRCLR1 3 BTB	7 BCLR1 2 BSC	4 BLS 2 REL	6 COM 2 DIR	4 COMA 1 INH	4 COMX 1 INH	7 COM 2 IX1	6 COM 1 IX	11 SWI 1 INH		2 CPX 2 IMM	4 CPX 2 DIR	5 CPX 3 EXT	6 CPX 3 IX2	5 CPX 2 IX1	4 CPX 1 IX	3 0011
4 0100	10 BRSET2 3 BTB	7 BSET2 2 BSC	4 BCC 2 REL	6 LSR 2 DIR	4 LSRA 1 INH	4 LSRX 1 INH	7 LSR 2 IX1	6 LSR 1 IX			2 AND 2 IMM	4 AND 2 DIR	5 AND 3 EXT	6 AND 3 IX2	5 AND 2 IX1	4 AND 1 IX	4 0100
5 0101	10 BRCLR2 3 BTB	7 BCLR2 2 BSC	4 BCS 2 REL								2 BIT 2 IMM	4 BIT 2 DIR	5 BIT 3 EXT	6 BIT 3 IX2	5 BIT 2 IX1	4 BIT 1 IX	5 0101
6 0110	10 BRSET3 3 BTB	7 BSET3 2 BSC	4 BNE 2 REL	6 ROR 2 DIR	4 RORA 1 INH	4 RORX 1 INH	7 ROR 2 IX1	6 ROR 1 IX			2 LDA 2 IMM	4 LDA 2 DIR	5 LDA 3 EXT	6 LDA 3 IX2	5 LDA 2 IX1	4 LDA 1 IX	6 0110
7 0111	10 BRCLR3 3 BTB	7 BCLR3 2 BSC	4 BEQ 2 REL	6 ASR 2 DIR	4 ASRA 1 INH	4 ASRX 1 INH	7 ASR 2 IX1	6 ASR 1 IX		2 TAX 1 INH		5 STA 2 DIR	6 STA 3 EXT	7 STA 3 IX2	6 STA 2 IX1	5 STA 1 IX	7 0111
8 1000	10 BRSET4 3 BTB	7 BSET4 2 BSC	4 BHCC 2 REL	6 LSL 2 DIR	4 LSLA 1 INH	4 LSLX 1 INH	7 LSL 2 IX1	6 LSL 1 IX		2 CLC 1 INH	2 EOR 2 IMM	4 EOR 2 DIR	5 EOR 3 EXT	6 EOR 3 IX2	5 EOR 2 IX1	4 EOR 1 IX	8 1000
9 1001	10 BRCLR4 3 BTB	7 BCLR4 2 BSC	4 BHCS 2 REL	6 ROL 2 DIR	4 ROLA 1 INH	4 ROLX 1 INH	7 ROL 2 IX1	6 ROL 1 IX		2 SEC 1 INH	2 ADC 2 IMM	4 ADC 2 DIR	5 ADC 3 EXT	6 ADC 3 IX2	5 ADC 2 IX1	4 ADC 1 IX	9 1001
A 1010	10 BRSET5 3 BTB	7 BSET5 2 BSC	4 BPL 2 REL	6 DEC 2 DIR	4 DECA 1 INH	4 DECX 1 INH	7 DEC 2 IX1	6 DEC 1 IX		2 CLI 1 INH	2 ORA 2 IMM	4 ORA 2 DIR	5 ORA 3 EXT	6 ORA 3 IX2	5 ORA 2 IX1	4 ORA 1 IX	A 1010
B 1011	10 BRCLR5 3 BTB	7 BCLR5 2 BSC	4 BMI 2 REL							2 SEI 1 INH	2 ADD 2 IMM	4 ADD 2 DIR	5 ADD 3 EXT	6 ADD 3 IX2	5 ADD 2 IX1	4 ADD 1 IX	B 1011
C 1100	10 BRSET6 3 BTB	7 BSET6 2 BSC	4 BMC 2 REL	6 INC 2 DIR	4 INCA 1 INH	4 INCX 1 INH	7 INC 2 IX1	6 INC 1 IX		2 RSP 1 INH		3 JMP 2 DIR	4 JMP 3 EXT	5 JMP 3 IX2	4 JMP 2 IX1	3 JMP 1 IX	C 1100
D 1101	10 BRCLR6 3 BTB	7 BCLR6 2 BSC	4 BMS 2 REL	6 TST 2 DIR	4 TSTA 1 INH	4 TSTX 1 INH	7 TST 2 IX1	6 TST 1 IX		2 NOP 1 INH	8 BSR 2 REL	7 JSR 2 DIR	8 JSR 3 EXT	9 JSR 3 IX2	8 JSR 2 IX1	7 JSR 1 IX	D 1101
E 1110	10 BRSET7 3 BTB	7 BSET7 2 BSC	4 BIL 2 REL								2 LDX 2 IMM	4 LDX 2 DIR	5 LDX 3 EXT	6 LDX 3 IX2	5 LDX 2 IX1	4 LDX 1 IX	E 1110
F 1111	10 BRCLR7 3 BTB	7 BCLR7 2 BSC	4 BIH 2 REL	6 CLR 2 DIR	4 CLRA 1 INH	4 CLRX 1 INH	7 CLR 2 IX1	6 CLR 1 IX		2 TXA 1 INH		5 STX 2 DIR	6 STX 3 EXT	7 STX 3 IX2	6 STX 2 IX1	5 STX 1 IX	F 1111

Abbreviations for Address Modes

INH	Inherent
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −126 to +129 from the opcode address.

INDEXED, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. Thus, this addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this two-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This addressing mode can be used in a manner similar to indexed 8-bit offset except that this three-byte instruction allows tables to be anywhere in memory.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte to which the specified bit is to be set or cleared. Thus, any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single two-byte instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit to be tested, and its condition (set or clear), is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage PC0 in Self-Check Mode All Other	V_{in}	 −0.3 to +15.0 −0.3 to +7.0	V
Port A and C Source Current per Pin (One at a Time)	I_{out}	10	mA
Operating Temperature Range MC6805S3P MC6805S3CP MC6805S3VP	T_A	 0 to 70 −40 to +85 −40 to +105	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature Plastic Package	T_J	150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation, it is recommended the V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{CC}$. Reliability of operation is enhanced if unused inputs except EXTAL are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

3

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic (P Suffix)	θ_{JA}	70	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications, $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

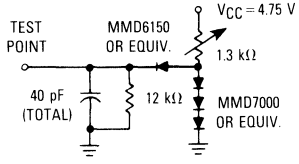


Figure 19. TTL Equivalent Test Load (Port B)

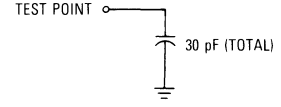


Figure 20. CMOS Equivalent Test Low (Port A)

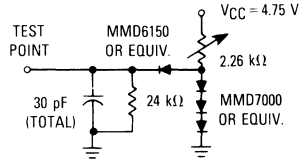


Figure 21. TTL Equivalent Test Load (Ports A and C)

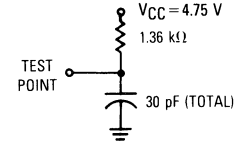


Figure 22. Open-Drain Equivalent Test Load (PB1, PB2, and PB3)

ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage	V_{IH}				V
$\overline{RESET}$ (4.75 ≤ V_{CC} ≤ 5.75)		4.0	—	V_{CC} + 0.7	
V_{CC} ≤ 4.75)		V_{CC} − 0.5	—	V_{CC} + 0.7	
$\overline{INT}$ (4.75 ≤ V_{CC} ≤ 5.75)		4.0	*	V_{CC} + 0.7	
(V_{CC} ≤ 4.75)		V_{CC} − 0.5	*	V_{CC} + 0.7	
All Other		2.0	—	V_{CC} + 0.7	
Input High Voltage PC0	V_{IH}				V
Port/Timer Mode		2.0	—	V_{CC} + 1.0	
Self-Check Mode		9.0	10.0	15.0	
Input Low Voltage	V_{IL}				V
$\overline{RESET}$		V_{SS}	—	0.8	
$\overline{INT}$		V_{SS}	*	1.5	
All Other (Except A/D Inputs)		V_{SS}	—	0.8	
$\overline{RESET}$ Hysteresis Voltages (See RESETS)					V
"Out of Reset"	V_{IRES+}	2.1	—	4.0	
"Into Reset"	V_{IRES-}	0.8	—	2.0	
Standby Supply Voltage ($\overline{INT2}$ Input Option)	V_{STBY}	3.0	—	5.75	V
Standby Current ($\overline{INT2}$ Input Option) (V_{STBY} = 3.0 V)	I_{STBY}	—	1.0	5.0	mA
Power Dissipation — No Port Loading					mW
(V_{CC} = 5.75 V, T_A = 0°C)	P_D	—	600	830	
(V_{CC} = 5.75 V, T_A = −40°C)	P_D	—	670	890	
Input Capacitance (Except Analog Inputs — See Note)	C_{in}	—	10	—	pF
Low Voltage Recover	V_{LVR}	—	—	4.75	V
Low Voltage Inhibit	V_{LVI}	2.75	3.75	4.70	V
Input Current	I_{in}				μA
$\overline{INT}$					
(V_{in} = 2.4 V to V_{CC})		—	20	50	
EXTAL					
(V_{in} = 2.4 V to V_{CC} Crystal Option)		—	—	10	
(V_{in} = 0.4 V Crystal Option)		—	—	−1600	
$\overline{RESET}$					
(V_{in} = 0.8 V) (External Charging Current)		−4.0	—	−50	

TBD = To Be Determined

NOTE: Port D analog inputs, when selected, C_{in} = 25 pF for the first 5 out of 30 cycles.

*This input (when unused) floats to approximately 2.0 V due to internal biasing.

3

SWITCHING CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Oscillator Frequency	f_{osc}	0.4	—	4.2	MHz
Cycle time ($4/f_{osc}$)	t_{cyc}	0.95	—	10	µs
$\overline{INT}$, $\overline{INT2}$, and TIMER Pulse Width $\overline{RESET}$ Pulse Width	t_{WL}, t_{WH} t_{RWL}	$t_{cyc}+250$ $t_{cyc}+250$	— —	— —	ns
$\overline{RESET}$ Delay Time (External Capacitance = 1 µF)	t_{RHL}	—	100	—	ns
$\overline{INT}$ Zero-Crossing Detection Input Frequency (for ±5° Accuracy)	f_{INT}	0.03	—	1	kHz
External Clock Input Duty Cycle (EXTAL)	—	40	50	60	%
Oscillator Startup Time Crystal	t_{su}	—	—	100	ms
SPICL High Time	t_{SPICLH}	4	—	—	t_{cyc}
SPICL Low Time	t_{SPICHL}	4	—	—	t_{cyc}
SPICL Rise and Fall Time	t_{Sr}, t_{Sf}	—	—	1	µs
SPID Input Data Setup Time	t_{SDS}	2	—	—	tcyc
SPID Input Data Hold Time	t_{SDH}	2	—	—	t_{cyc}
SPICL to SPISS Lag Time	t_{SStG}	4	—	—	t_{cyc}
SPISS to SPICL Lead Time	t_{SSLD}	4	—	—	t_{cyc}
Start Bit to First Clock Lead Time	t_{STL}	1	—	—	t_{cyc}
External Timer Input to Timer Change Time	t_{PCT}	3	—	—	t_{cyc}
Timer Change to Port B Toggle Time	t_{TPB}	2	—	—	t_{cyc}
$\overline{INT2}$ to Timer A Load Time	t_{INTL}	3	—	—	t_{cyc}

A/D CONVERTER CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Min	Typ	Max	Unit	Comments
Resolution	8	8	8	Bits	
Non-Linearity*	—	—	±1/2	LSB	After removing zero-offset and full-scale errors
Quantizing Error	—	—	±1/2	LSB	
Conversion Range V_{RH} V_{RL}	 — V_{SS}	 — —	 V_{CC} 0.2	V	A/D accuracy may decrease proportionately as V_{RH}–V_{RL} is reduced below 4.0 V. The sum of V_{RH} and V_{RL} must not exceed V_{CC}
Conversion Time	30	30	30	t_{cyc}	Includes sampling time
Monotonicity	(Inherent within total error)				
Sample Time	5	5	5	t_{cyc}	
Sample/Hold Capacitance, Input	—	—	25	pF	
Analog Input Voltage	V_{RL}	—	V_{RH}	V	Transients on any analog lines are not allowed at any time during sampling or accuracy may be degraded

* For V_{RH} = 0.4 V to 5.0 V and V_{RL} = 0 V.

PORT ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Port A with CMOS Drive Enable					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Output High Voltage, I_{Load} = −10 μA	V_{OH}	V_{CC} − 1.0	—	—	V
Input High Voltage, I_{Load} = −300 μA (max.)	V_{IH}	2.0	—	V_{CC} + 0.7	V
Input Low Voltage, I_{Load} = −500 μA (max.)	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current (V_{in} = 2.0 V to V_{CC})	I_{IH}	—	—	−300	μA
Hi-Z State Input Current (V_{in} = 0.4 V)	I_{IL}	—	—	−500	μA
Port B					
Output Low Voltage, I_{Load} = 3.2 mA	V_{OL}	—	—	0.4	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.0	V
Output High Voltage, I_{Load} = −200 μA	V_{OH}	2.4	8	—	V
Darlington Current Drive (Source)*, V_O = 1.5 V	I_{OH}	−1.0	—	−10	mA
Input High Voltage	V_{IH}	2.0	—	V_{CC} + 0.7	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port C and Port A with CMOS Drive Disabled					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC} + 0.7	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port D (Digital Inputs Only)					
Input High Voltage	V_{IH}	2.0	—	V_{CC} + 0.7	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Input Current**	I_{in}	—	<1	10	μA

*Not applicable if programmed to open-drain state.

**PD4/V_{RL} — PD5/V_{RH}:
The A/D conversion resistor (15 kΩ typical) is connected internally between these two lines, impacting their use as digital inputs in some applications.

3

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola in the following media:

MDOS, disk file
MS-DOS/PC-DOS disk file
EPROM(s) 2532, 2732, or two each: 2516/2716

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, a sales person, or a Motorola representative.

FLEXIBLE DISKS

Several types of flexible disks (MDOS™ or MS™-DOS/PC-DOS disk file), programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. In either case, the diskette should be clearly labeled with the customer's name, date, project or product name, and the name of the file containing the pattern.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to expedite the process in case of any difficulty with the pattern file.

3

MDOS Disk File

MDOS is Motorola's Disk Operating System available on the EXORciser® development system. The disk media submitted must be a single-side, single-density, 8-inch MDOS compatible floppy diskette. The diskette must contain the minimum set of MDOS system files in addition to the pattern file.

The .LO output of the M6805 cross assembler should be furnished. In addition, the file must be produced (using the ROLLOUT command) containing the absolute image of the M6805 memory. Include the entire memory image of both data and program space. All unused bytes, including those in the user space, must be set to zero.

MS-DOS/PC-DOS Disk File

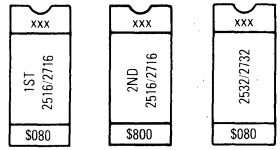
MS-DOS is Microsoft's Disk Operating System. PC-DOS is IBM® Personal Computer (PC) Disk Operating System. Disk media submitted must be a standard density (360K), double-sided 5 1/4 inch compatible floppy diskette. The diskette must contain object file code in Motorola's S-record format. The S-record format is a character-based object file format generated by M6805 cross assemblers and linkers on IBM PC style machines.

EPROMs

An MC68705S3, 2532, 2732, 2516 (2), or 2716 (2) type EPROM(s), programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. Since all program and data space information will fit on one MC68705S3/2532/2732 or two 2516/2716 type EPROM(s), the EPROM(s) must be programmed as described in the following paragraph.

For the 2532 or 2732, the ROM code should be located from $080 to $EFF and the interrupt vectors from $FF8 to $FFF. For the 2516s or 2716s, the ROM code should be located from $080 to $7FF in the first EPROM and from $0 to $6EF in the second EPROM. The interrupt vectors should be in the second EPROM from $7F8 to $7FF.

EPROM MARKING



XXX = CUSTOMER ID

VERIFICATION MEDIA

All original pattern media (EPROMs or floppy disk) are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked and the verification form should be completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for creation of the customer's mask. To aid in the verification process, Motorola will program ***customer supplied*** blank EPROM(s) or DOS disk from the data file used to create the custom mask.

ROM VERIFICATION UNITS (RVUs)

Ten MCUs containing the customer's ROM pattern will be sent for program verification. These units will have been made using the custom mask but are for the purpose of ROM verification only. For expediency, the MCUs are usually unmarked, packaged in ceramic, and tested with five volts and at room temperature. These RVUs are free with the minimum order quantity but are not production parts. These RVUs are not guaranteed by Motorola Quality Assurance.

ORDERING INFORMATION

The following table provides generic information pertaining to the package type, temperature, and MC order numbers for the MC6805S3.

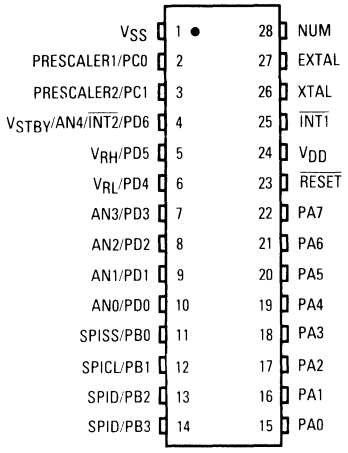
Table 5. Generic Information

Package Type	Temperature	Order Number
Plastic (P Suffix)	0°C to 70°C	MC6805S3P
	−40°C to +85°C	MC6805S3CP
	−40 to +105°C	MC6805S3VP

MDOS is a trademark of Motorola Inc.
MS is a trademark of Microsoft, Inc.
EXORciser is a registered trademark of Motorola Inc.
IBM is a registered trademark of International Business Machines Corporation.

MECHANICAL DATA

PIN ASSIGNMENT



3

MC6805U2

Technical Summary
8-Bit Microcontroller Unit

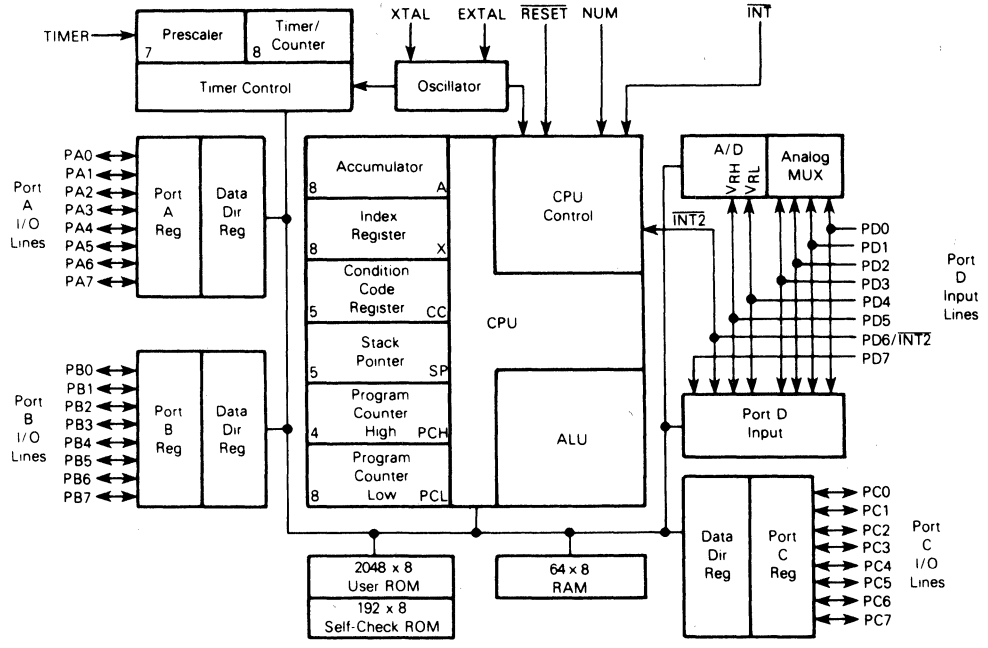
The MC6805U2 (HMOS) Microcontroller Unit (MCU) is a member of the M6805 Family of microcontrollers. This low cost and high-speed MCU has parallel I/O capability with pins programmable as input or output. This publication contains condensed information on the MCU; for detailed information, refer to *M6805 HMOS, M146805 CMOS Family User's Manual* (M6805UM(AD2)) or contact your local Motorola sales office.

Refer to the block diagram for the hardware features and to the list below for additional features available on the MCU.

- Internal 8-Bit Timer with 7-Bit Prescaler
- On-Chip Oscillator
- Memory Mapped I/O
- Versatile Interrupt Handling
- True Bit Manipulation
- Bit Test and Branch Instruction
- Vectored Interrupts
- 2048 Bytes of ROM
- 64 Bytes of RAM
- Self-Check Mode
- 24 Bidirectional I/O Lines

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

V_{CC} AND V_{SS}

Power is supplied to the microcontroller using these two pins. V_{CC} is +5.25 volts (±0.5Δ) power, and V_{SS} is ground.

NUM

This pin is not for user applications and must be connected to V_{SS}.

$\overline{INT}$

This pin provides the capability for asynchronously applying an external interrupt to the MCU. Refer to **INTERRUPTS** for more detailed information.

EXTAL, XTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a resistor/capacitor combination, or an external signal (depending upon selected manufacturing mask option) is connected to these pins to provide a system clock.

RC Oscillator

With this option, a resistor is connected to the oscillator pins as shown in Figure 1. The relationship between R and f_{osc} is shown in Figure 2.

Crystal

The circuit shown in Figure 1 is recommended when using a crystal. Using an external CMOS oscillator is recommended when crystals outside the specified ranges are to be used. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time. Refer to **ELECTRICAL SPECIFICATIONS** for V_{CC} specifications.

External Clock

An external clock should be applied to the EXTAL input with the XTAL input connected to ground, as shown in Figure 1. This option may only be used with the crystal oscillator option selected in the mask option register.

TIMER

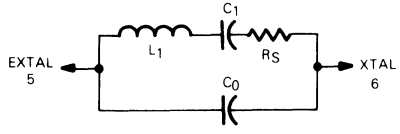
This pin is used as an external input to control the internal timer/counter circuitry. This pin also detects a higher voltage level used to initiate the self-test program.

$\overline{RESET}$

This pin has a Schmitt trigger input and an on-chip pullup. The MCU can be reset by pulling $\overline{RESET}$ low.

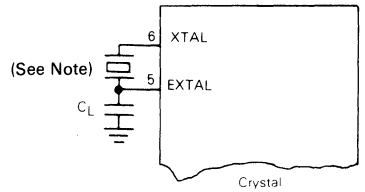
INPUT/OUTPUT LINES (PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD7)

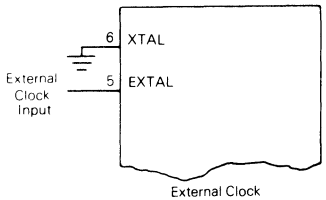
These 32 lines are arranged into four 8-bit ports (A, B, C, and D) Ports A, B, and C are programmable as either

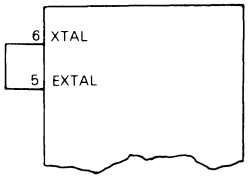


AT – Cut Parallel Resonance Crystal
C_O = 7 pF Max
Freq. = 4.0 MHz @ C_L = 24 pF
R_S = 50 ohms Max.

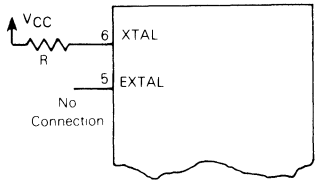
Piezoelectric ceramic resonators which have the equivalent specifications may be used instead of crystal oscillators. Follow ceramic resonator manufacturer's suggestions for C_O, C_1, and R_S values.







Approximately 25% to 50% Accuracy
Typical t_{CYC} = 1.25 μs
External Jumper



Approximately 10% to 25% Accuracy
(Excludes Resistor Tolerance)
External Resistor

NOTE: The recommended C_L value with a 4.0 MHz crystal is 27 pF maximum, including system distributed capacitance. There is an internal capacitance of approximately 25 pF on the XTAL pin. For crystal frequencies other than 4 MHz, the total capacitance on each pin should be scaled as the inverse of the frequency ratio. For example, with a 2 MHz crystal, use approximately 50 pF on EXTAL and approximately 25 pF on XTAL. The exact value depends on the motional-arm parameters of the crystal used.

Figure 1. Oscillator Connections

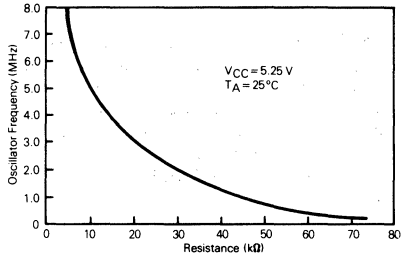


Figure 2. Typical Frequency vs Resistance for RC Oscillator Option Only

inputs or outputs under software control of the data direction registers. Port D is a fixed input port and not controlled by any data register. Port D bit 6 shares input signal $\overline{INT2}$, which is used for external interrupts. Refer to **PROGRAMMING** for additional information.

3

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

Ports A, B, and C are programmable as either input or output under software control of the corresponding write-only direction register (DDR). Port D lines are input only. The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic one for output and a logic zero for input. On reset, all the DDRs are initialized to a logic zero state to put the ports in the input mode. The port output registers are not initialized on reset and should be written to before setting the DDR bits.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This port write may be used to initialize the data registers and avoid undefined outputs. Care must be exercised when using read-modify-write instructions since the data read corresponds to the pin level if the DDR is an input (zero) and also to the latched output when the DDR is an output (one). Refer to Table 1 for I/O functions and to Figure 3 for typical port circuitry.

Table 1. I/O Pin Functions

Data Direction Register Bit	Latch Output Data Bit	Output State	Input To MCU
1	0	0	0
1	1	1	1
0	X	Hi-Z*	Pin

*Port B and C are three-state ports. Port A has optional internal pullup devices to provide CMOS data drive capability.

MEMORY

The MCU is capable of addressing 4096 bytes of memory and I/O registers. The memory map is shown in Figure 4. The locations consist of user ROM, self-check ROM, user RAM, a miscellaneous register (MR), timer registers, and I/O. The interrupt and reset vectors are located from $FF8 to $FFF.

The stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer decrements during pushes and increments during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

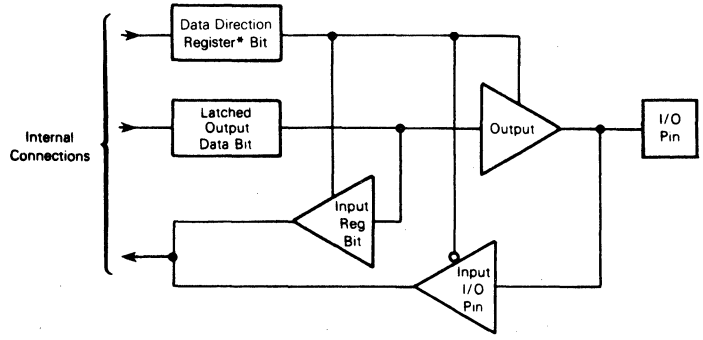
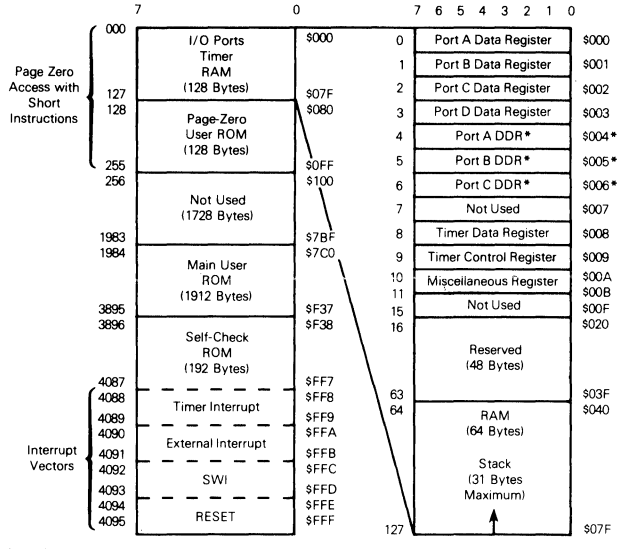


Figure 3. Typical Port I/O Circuitry and Register Configuration



*Caution: Data direction registers (DDRs) are write-only; they read as $FF.

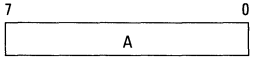
Figure 4. Memory Map

REGISTERS

The MCU contains the registers described in the following paragraphs.

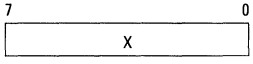
ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



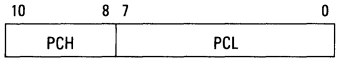
INDEX REGISTER (X)

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16-bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



PROGRAM COUNTER (PC)

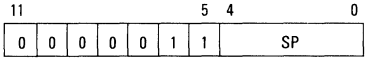
The program counter is an 12-bit register that contains the address of the next byte to be fetched.



STACK POINTER (SP)

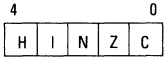
The stack pointer is a 12-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set at location $07F. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

The seven most-significant bits of the stack pointer are permanently set at 0000011. Subroutines and interrupts may be nested down to location $061 (31 bytes maximum), which allows the programmer to use up to 15 levels of subroutine calls (less if interrupts are allowed).



CONDITION CODE REGISTER (CC)

The condition code register is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program, and specifications can be taken as a result of their state. Each bit is explained in the following paragraphs.



Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, the timer and external interrupt is masked (disabled). If an external interrupt occurs while this bit is set, the interrupt is latched and is processed as soon as the interrupt bit is cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic one).

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/Borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions, and during shifts and rotates.

3

SELF-CHECK

The self-check is initiated by connecting the MCU as shown in Figure 5 and then monitoring the output of port C (bit 3) for an oscillation of approximately 7 Hz. The following test are executed automatically:

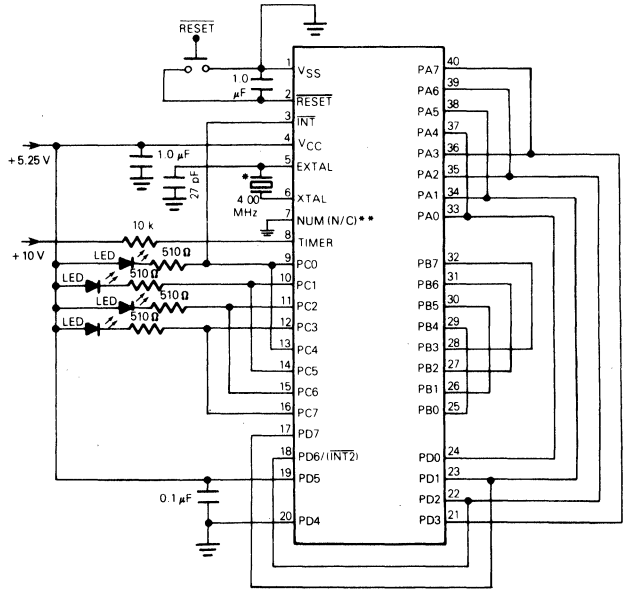
I/O — Functionally exercise I/O ports.
RAM — Walking bit test.
ROM — Exclusive OR with ODD "1s" parity result.
Timer — Functionally exercise timer.
Interrupts — Functionally exercise external and timer interrupts.

The RAM and ROM can be called by a user program. The timer test may be called if the timer input is the internal clock. Table 2 shows the status of the LEDs as a result of a failure. Port C is tested only once (just after reset). If port C fails, no lights will appear.

Table 2. Self-Check Error Patterns

LED Meanings				
PC0	**PC1**	**PC2**	**PC3**	**Remarks (1: LED ON; 0: LED OFF)**
1	0	1	0	Bad I/O
0	0	1	0	Bad Timer
1	1	0	0	Bad RAM
0	1	0	0	Bad ROM
1	0	0	0	Bad A/D
0	0	0	0	Bad Interrupts or Request Flag
All Flashing				Good Device

Anything else Bad Part, Bad Port C, etc.



*This connection depends on clock oscillator user selectable mask option. Use jumper if the RC mask option is selected.
**For the MC6805R2/MC6805U2 pin 7 is not for user application and must be connected to V_{SS}. For the MC6805R3/MC6805U3 pin 7 is not connected.

Figure 5. Self-Check Connections

RESETS

The MCU can be reset three ways: (1) by initial power-up, (2) by the external reset input ($\overline{RESET}$), and (3) by an optional, internal, low-voltage detect circuit. The $\overline{RESET}$ input consists mainly of a Schmitt trigger that senses the $\overline{RESET}$ line logic level.

POWER-ON-RESET (POR)

An internal reset is generated on power-up that allows the internal clock generator to stabilize. The power-on reset is used strictly for power turn-on conditions and should not be used to detect any drop in the power supply voltage. A delay of t_{RHL} milliseconds is required before allowing the $\overline{RESET}$ input to go high. Connecting a capacitor to the $\overline{RESET}$ input (Figure 6) typically provides sufficient delay.

EXTERNAL RESET INPUT

The MCU is reset when a logic zero is applied to the $\overline{RESET}$ input for a period longer than one machine cycle (t_{CYC}). Under this type of reset, the Schmitt trigger switches off at V_{IRES-} to provide an internal reset voltage.

LOW-VOLTAGE INHIBIT (LVI)

The optional low-voltage detection circuit causes a reset of the MCU if the power supply voltage falls below a certain level (V_{LVI}). The only requirement is that the V_{CC} must remain at or below the V_{LVI} threshold for one t_{CYC} minimum.

In typical applications, the V_{CC} bus filter capacitor will eliminate negative-going voltage glitches of less than one t_{CYC}. The output from the low-voltage detector is connected directly to the internal reset circuitry. It also forces the $\overline{RESET}$ pin low via a strong discharge device through a resistor. The internal reset is removed once the power supply voltage rises above a recovery level (V_{LVR}) at which time a normal power-on reset occurs.

INTERRUPTS

The MCU can be interrupted four different ways: (1) through the external interrupt $\overline{INT}$ input pin, (2) with the internal timer interrupt request, (3) using the software interrupt instruction (SWI), or (4) the external port D bit 6 ($\overline{INT2}$) input pin.

Interrupts cause the processor registers to be saved on the stack and the interrupt mask (I bit) set to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack after which normal processing resumes. The stacking order is shown in Figure 7.

Unlike $\overline{RESET}$, hardware interrupts do not cause the current instruction execution to be halted but are considered pending until the current instruction is complete.

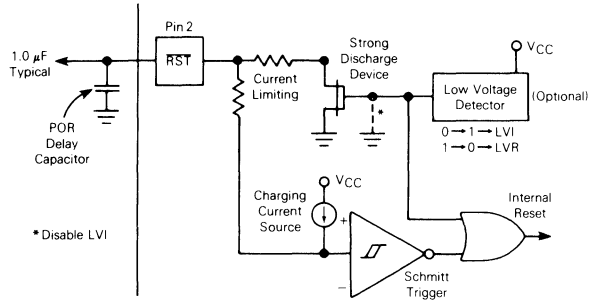
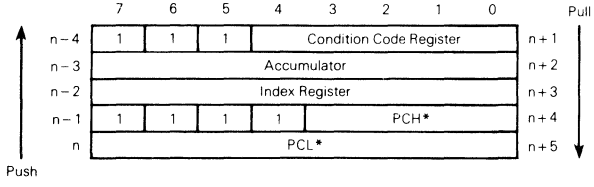


Figure 6. RESET Configurations



*For subroutine calls, only PCH and PCL are stacked.

Figure 7. Interrupt Stacking Order

NOTE

The current instruction is considered to be the one already fetched and being operated on.

When the current instruction is complete, the processor checks all pending hardware interrupts and, if unmasked (I bit clear), proceeds with interrupt processing; otherwise, the next instruction is fetched and executed. Masked interrupts are latched for later interrupt service. If the timer interrupt status bit is cleared before unmasking the interrupt, then the interrupt is not latched.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction regardless of the setting of the I bit. Refer to Figure 8 for the reset and interrupt instruction processing sequence.

TIMER INTERRUPT

If the timer mask bit (TCR6) is cleared, each time the timer decrements to zero (transitions from $01 to $00) an interrupt request is generated. The actual processor interrupt is generated only if the interrupt mask bit of the condition code register (CCR) is also cleared. When the interrupt is recognized, the current state of the machine is pushed onto the stack, and the I bit in the CCR is set, masking further interrupts until the present one is serviced. The contents of the timer interrupt vector, containing the location of the timer interrupt service routine, is then loaded into the program counter. At the end of the timer interrupt service routine, the software normally executes an RTI instruction which restores the machine state and starts executing the interrupted program.

EXTERNAL INTERRUPT

The external interrupt is internally synchronized and then latched on the falling edge of $\overline{INT}$ and $\overline{INT2}$. Clearing the I bit enables the external interrupt. The $\overline{INT2}$ interrupt has an interrupt request bit (bit 7) and a mask bit (bit 6) in the miscellaneous register (MR). The $\overline{INT2}$ interrupt is inhibited when the mask bit is set. The $\overline{INT2}$ is always read as a digital input on port D. The $\overline{INT2}$ and timer

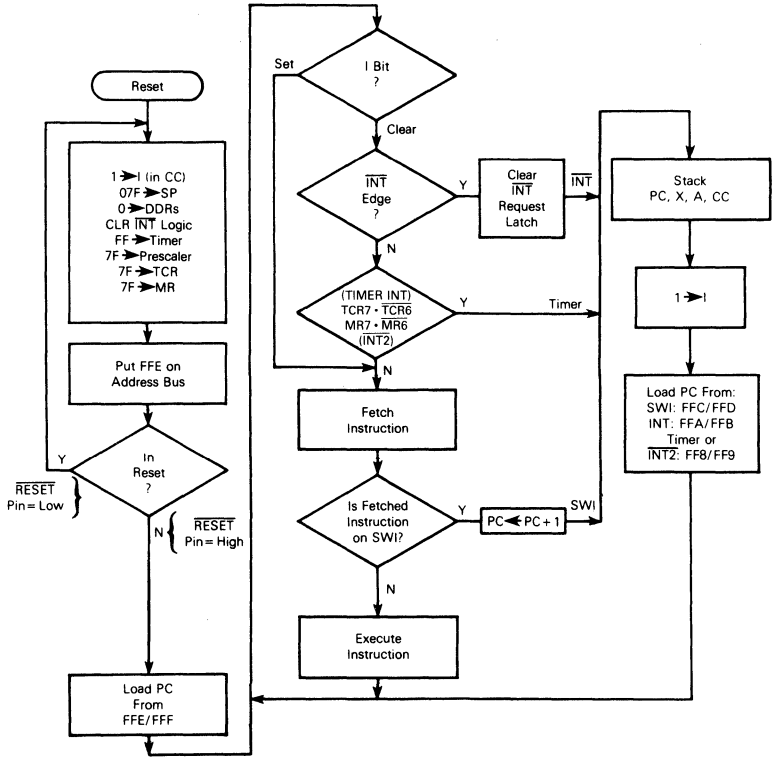


Figure 8. Reset and Interrupt Processing Flowchart

interrupt request bits, if set, cause the MCU to process and interrupt when the condition code I bit is clear. The following paragraphs describe two typical external interrupt circuits.

Zero-Crossing Interrupt

A sinusoidal input signal (f_{INT} maximum) can be used to generate an external interrupt (see Figure 9a) for use as a zero-crossing detector (for negative transitions of the ac sinusoid). This type of circuit allows applications such as servicing time-of-day routines and engaging/disengaging ac power control devices. Off-chip, full-wave rectification provides an interrupt at every zero crossing of the ac signal and thereby provides a 2f clock.

Digital-Signal Interrupt

With this type of circuit (Figure 9b), the $\overline{INT}$ pin can be driven by a digital signal. The maximum frequency of a signal that can be recognized by the TIMER or $\overline{INT}$ pin logic is dependent on the parameter labeled t_{WL}, t_{WH}. Refer to **TIMER** for additional information.

SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. If the I bit is zero, SWI executes after the other interrupts. The SWI execution is similar to the hardware interrupts.

TIMER

The MCU consists of an 8-bit software programmable counter driven by a 7-bit prescaler. The timer source is made during manufacturing as a mask option. The 8-bit counter may be loaded under program control and is decremented toward zero. When the timer reaches zero, the timer interrupt request bit (bit 7) in the timer control register (TCR) is set. Refer to Figure 10 for timer block diagram.

The timer interrupt can be masked (disabled) by setting the timer interrupt mask bit (bit 6) in the TCR. When the I bit in the condition code register is cleared and TCR bit 6 is cleared, the processor receives the interrupt. The MCU responds to this interrupt by 1) saving the present CPU state on the stack, 2) fetching the timer interrupt vector, and 3) executing the interrupt routine. The timer interrupt request bit must be cleared by software. Refer to **RESETS** and **INTERRUPTS** for additional information.

The prescaler is a 7-bit divider which is used to extend the maximum length of the timer. To avoid truncation errors, the prescaler is cleared when TCR bit 3 is set to

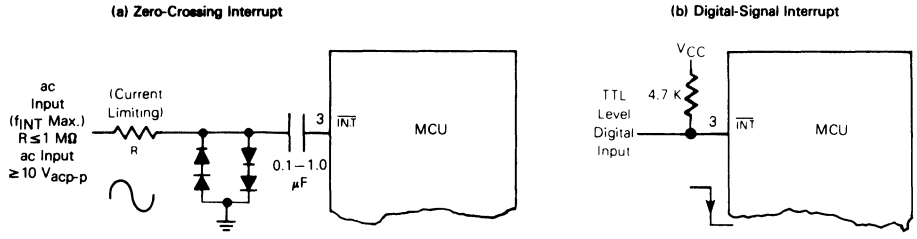


Figure 9. Typical Interrupt Circuits

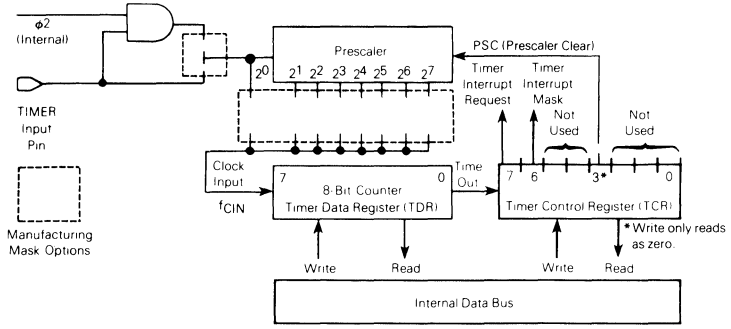


Figure 10. Timer Block Diagram

a logic one; however, the TCR bit 3 always reads as a logic zero to ensure proper operation with read-modify-write instructions.

The timer continues to count past zero, falling from $00 through $FF, and continues the countdown. The counter can be read at any time by reading the timer data register (TDR). This allows a program to determine the length of time since a timer interrupt has occurred without disturbing the counting process. Three machine cycles are required for a change in state of the TIMER pin to decrement the timer prescaler.

Clock input to the timer can be from an external source or from the internal phase two signal. Clock source is one of the mask options. A prescaler mask option is also available to select a divide option of a power of two up to 128.

TIMER CONTROL REGISTER (TCR) $009

This 8-bit register controls various functions such as timer interrupt request, timer interrupt inhibit, and prescaler clear signal. Bit 3 is write only.

7	6	5	4	3	2	1	0
TIR	TIM	1	1	PSC*	1	1	1
RESET:							
0	1	U	U	U	U	U	U

TIR — Timer Interrupt Request
Used to indicate the timer interrupt when it is logic one
1 = Set when the timer data register changes to all zeros
0 = Cleared by external reset, power-on reset, or under program control

TIM — Timer Interrupt Mask
Used to inhibit the timer interrupt
1 = Interrupt inhibited
0 = Interrupt enabled

PSC — Prescaler Clear
Write only bit. Writing a one to this bit resets the prescaler to zero. A read of this location always indicates a zero.

Bits 5, 4, 2–0 — Not used.

INSTRUCTION SET

The MCU has a set of 59 basic instructions which can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following list of instructions.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (2's Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Higher	BHI

— Continued —

Function	Mnemonic
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
(Branch if Higher or Same)	(BHS)
Branch if Carry Set	BCS
(Branch if Lower)	(BLO)
Branch if Not Equal	BNE
Branch if Equal	BEQ
Branch if Half Carry Clear	BHCC
Branch if Half Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Bit is Clear	BMC
Branch if Interrupt Mask Bit is Set	BMS
Branch if Interrupt Line is Low	BIL
Branch if Interrupt Line is High	BIH
Branch to Subroutine	BSR

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the first 256 bytes of the memory space, where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear and bit test, and branch functions are all implemented with a single instruction. For test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch if Bit n is Set	BRSET n (n=0 . . . 7)
Branch if Bit n is Clear	BRCLR n (n=0 . . . 7)
Set Bit n	BSET n (n=0 . . . 7)
Clear Bit n	BCLR n (n=0 . . . 7)

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI

— Continued —

Function	Mnemonic
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No-Operation	NOP

OPCODE MAP SUMMARY

Table 3 is an opcode map for the instructions used on the MCU.

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single byte instructions, while the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. Two byte direct addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction.

RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true.

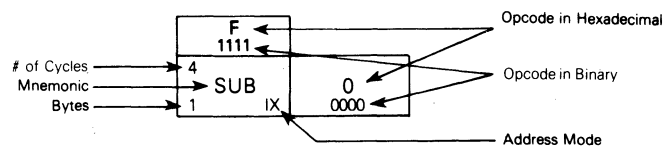
Table 3. Opcode Map

Hi / Low	Bit Manipulation		Branch	Read-Modify-Write					Control		Register/Memory						Hi / Low
	BTB 0 0000	BSC 1 0001	REL 2 0010	DIR 3 0011	INH 4 0100	INH 5 0101	IX1 6 0110	IX 7 0111	INH 8 1000	INH 9 1001	IMM A 1010	DIR B 1011	EXT C 1100	IX2 D 1101	IX1 E 1110	IX F 1111	
0 0000	10 BRSET0 3 BTB	7 BSET0 2 BSC	4 BRA 2 REL	6 NEG 2 DIR	4 NEG 1 INH	4 NEG 1 INH	7 NEG 2 IX1	6 NEG 1 IX	9 RTI 1 INH		2 SUB 2 IMM	4 SUB 2 DIR	5 SUB 3 EXT	6 SUB 3 IX2	5 SUB 2 IX1	4 SUB 1 IX	0 0000
1 0001	10 BRCLR0 3 BTB	7 BCLR0 2 BSC	4 BRN 2 REL						6 RTS 1 INH		2 CMP 2 IMM	4 CMP 2 DIR	5 CMP 3 EXT	6 CMP 3 IX2	5 CMP 2 IX1	4 CMP 1 IX	1 0001
2 0010	10 BRSET1 3 BTB	7 BSET1 2 BSC	4 BHI 2 REL								2 SBC 2 IMM	4 SBC 2 DIR	5 SBC 3 EXT	6 SBC 3 IX2	5 SBC 2 IX1	4 SBC 1 IX	2 0010
3 0011	10 BRCLR1 3 BTB	7 BCLR1 2 BSC	4 BLS 2 REL	6 COM 2 DIR	4 COMA 1 INH	4 COMX 1 INH	7 COM 2 IX1	6 COM 1 IX	11 SWI 1 INH		2 CPX 2 IMM	4 CPX 2 DIR	5 CPX 3 EXT	6 CPX 3 IX2	5 CPX 2 IX1	4 CPX 1 IX	3 0011
4 0100	10 BRSET2 3 BTB	7 BSET2 2 BSC	4 BCC 2 REL	6 LSR 2 DTR	4 LSRA 1 INH	4 LSRX 1 INH	7 LSR 2 IX1	6 LSR 1 IX			2 AND 2 IMM	4 AND 2 DIR	5 AND 3 EXT	6 AND 3 IX2	5 AND 2 IX1	4 AND 1 IX	4 0100
5 0101	10 BRCLR2 3 BTB	7 BCLR2 2 BSC	4 BCS 2 REL								2 BIT 2 IMM	4 BIT 2 DIR	5 BIT 3 EXT	6 BIT 3 IX2	5 BIT 2 IX1	4 BIT 1 IX	5 0101
6 0110	10 BRSET3 3 BTB	7 BSET3 2 BSC	4 BNE 2 REL	6 ROR 2 DIR	4 RORA 1 INH	4 RORX 1 INH	7 ROR 2 IX1	6 ROR 1 IX			2 LDA 2 IMM	4 LDA 2 DIR	5 LDA 3 EXT	6 LDA 3 IX2	5 LDA 2 IX1	4 LDA 1 IX	6 0110
7 0111	10 BRCLR3 3 BTB	7 BCLR3 2 BSC	4 BEQ 2 REL	6 ASR 2 DIR	4 ASRA 1 INH	4 ASRX 1 INH	7 ASR 2 IX1	6 ASR 1 IX		2 TAX 1 INH		5 STA 2 DIR	6 STA 3 EXT	7 STA 3 IX2	6 STA 2 IX1	5 STA 1 IX	7 0111
8 1000	10 BRSET4 3 BTB	7 BSET4 2 BSC	4 BHCC 2 REL	6 LSL 2 DIR	4 LSLA 1 INH	4 LSLX 1 INH	7 LSL 2 IX1	6 LSL 1 IX		2 CLC 1 INH	2 EOR 2 IMM	4 EOR 2 DIR	5 EOR 3 EXT	6 EOR 3 IX2	5 EOR 2 IX1	4 EOR 1 IX	8 1000
9 1001	10 BRCLR4 3 BTB	7 BCLR4 2 BSC	4 BHCS 2 REL	6 ROL 2 DIR	4 ROLA 1 INH	4 ROLX 1 INH	7 ROL 2 IX1	6 ROL 1 IX		2 SEC 1 INH	2 ADC 2 IMM	4 ADC 2 DIR	5 ADC 3 EXT	6 ADC 3 IX2	5 ADC 2 IX1	4 ADC 1 IX	9 1001
A 1010	10 BRSET5 3 BTB	7 BSET5 2 BSC	4 BPL 2 REL	6 DEC 2 DIR	4 DECA 1 INH	4 DECX 1 INH	7 DEC 2 IX1	6 DEC 1 IX		2 CLI 1 INH	2 ORA 2 IMM	4 ORA 2 DIR	5 ORA 3 EXT	6 ORA 3 IX2	5 ORA 2 IX1	4 ORA 1 IX	A 1010
B 1011	10 BRCLR5 3 BTB	7 BCLR5 2 BSC	4 BMI 2 REL							2 SEI 1 INH	2 ADD 2 IMM	4 ADD 2 DIR	5 ADD 3 EXT	6 ADD 3 IX2	5 ADD 2 IX1	4 ADD 1 IX	B 1011
C 1100	10 BRSET6 3 BTB	7 BSET6 2 BSC	4 BMC 2 REL	6 INC 2 DIR	4 INCA 1 INH	4 INCX 1 INH	7 INC 2 IX1	6 INC 1 IX		2 RSP 1 INH		3 JMP 2 DIR	4 JMP 3 EXT	5 JMP 3 IX2	4 JMP 2 IX1	3 JMP 1 IX	C 1100
D 1101	10 BRCLR6 3 BTB	7 BCLR6 2 BSC	4 BMS 2 REL	6 TST 2 DIR	4 TSTA 1 INH	4 TSTX 1 INH	7 TST 2 IX1	6 TST 1 IX		2 NOP 1 INH	8 BSR 2 REL	7 JSR 2 DIR	8 JSR 3 EXT	9 JSR 3 IX2	8 JSR 2 IX1	7 JSR 1 IX	D 1101
E 1110	10 BRSET7 3 BTB	7 BSET7 2 BSC	4 BIL 2 REL								2 LDX 2 IMM	4 LDX 2 DIR	5 LDX 3 EXT	6 LDX 3 IX2	5 LDX 2 IX1	4 LDX 1 IX	E 1110
F 1111	10 BRCLR7 3 BTB	7 BCLR7 2 BSC	4 BIH 2 REL	6 CLR 2 DIR	4 CLRA 1 INH	4 CLRX 1 INH	7 CLR 2 IX1	6 CLR 1 IX		2 TXA 1 INH		5 STX 2 DIR	6 STX 3 EXT	7 STX 3 IX2	6 STX 2 IX1	5 STX 1 IX	F 1111

Abbreviations for Address Modes

INH	Inherent
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



Otherwise, control proceeds to the next instruction. The span of relative addressing is from −126 to +129 from the opcode address.

INDEX, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. This addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this two-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This addressing mode can be used in a manner similar to indexed, 8-bit offset except that this three-byte instruction allows tables to be anywhere in memory.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte to which the specified bit is to be set or cleared. Any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single two-byte instruction.

CAUTION

The corresponding DDRs for ports A, B, and C are write-only registers (registers at $004, $005, and $006). A read operation on these registers is undefined. Since BSET and BCLR are read-modify-write functions, these functions cannot be used to set or clear A DDR bit (all "unaffected" bits would be set). It is recommended that all DDR bits in a port be written using a single-store instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit that is to be tested and its condition (set or clear) is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage Self-Check Mode (TIMER Pin Only)	V_{in}	−0.3 to +7.0 −0.3 to +15.0	V
Operating Temperature Range MC6805U2 MC6805U2C MC6805U2V	T_A	T_L to T_H 0 to +70 −40 to +85 −40 to +105	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature Plastic PLCC Cerdip	T_J	 150 150 175	°C

These device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation it is recommended the V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{CC}$. Reliability of operation is enhanced if unused inputs except EXTAL are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic (P Suffix) PLCC (FN Suffix) Cerdip (S Suffix)	θ_{JA}	 60 100 60	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

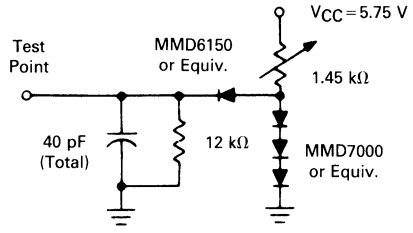


Figure 11. TTL Equivalent Test Load (Port B)

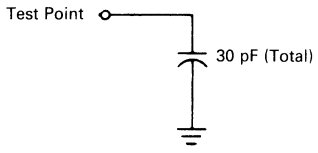


Figure 12. CMOS Equivalent Test Load (Port A)

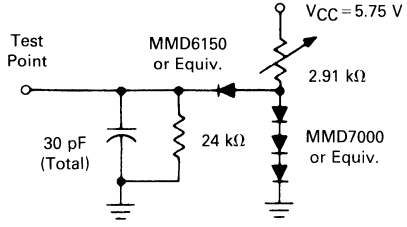


Figure 13. TTL Equivalent Test Load (Ports A and C)

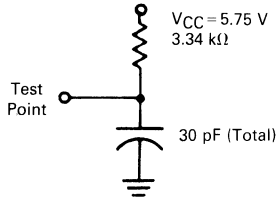


Figure 14. Open-Drain Equivalent Test Load (Port C)

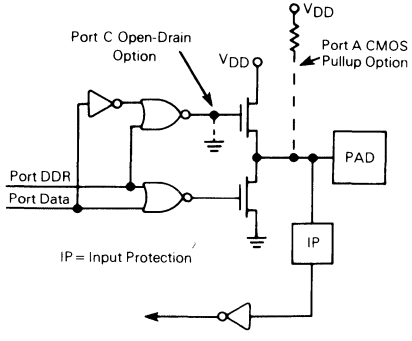


Figure 15. Ports A and C Logic Diagram

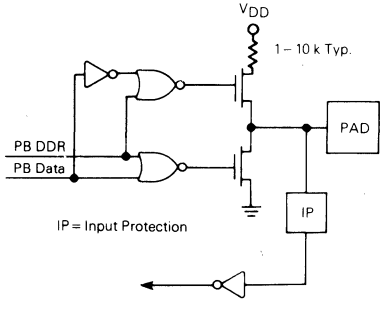


Figure 16. Port B Logic Diagram

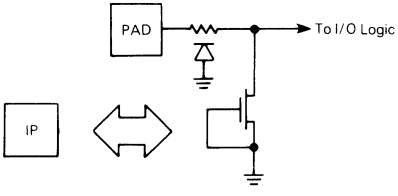


Figure 17. Typical Input Protection

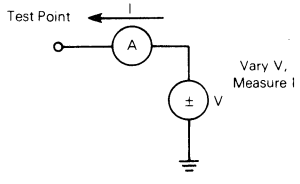


Figure 18. I/O Characteristic Measurement Circuit

ELECTRICAL CHARACTERISTICS (V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage	V_{IH}				V
RESET (4.75≤V_{CC}≤5.75)		4.0	—	V_{CC}	
___ (V_{CC}<4.75)		V_{CC} − 0.5	—	V_{CC}	
INT (4.75≤V_{CC}≤5.75)		4.0	*	V_{CC}	
(V_{CC}<4.75)		V_{CC} − 0.5	*	V_{CC}	
All Other (Except Timer)		2.0	—	V_{CC}	
Input High Voltage Timer	V_{IH}				V
Timer Mode		2.0	—	V_{CC} + 1.0	
Self-Check Mode		9.0	10.0	15.0	
Input Low Voltage	V_{IL}				V
RESET		V_{SS}	—	0.8	
INT		V_{SS}	*	1.5	
All Other		V_{SS}	—	0.8	
Reset Hysteresis Voltages					V
"Out of Reset"	V_{IRES+}	2.1	—	4.0	
"Into Reset"	V_{IRES-}	0.8	—	2.0	
INT Zero-Crossing Voltage, Through a Capacitor	V_{INT}	2	—	4	$V_{ac\ p\text{-}p}$
Internal Power Dissipation (No Port Loading, T_A = 0°C	P_{INT}	—	520	740	mW
V_{CC} = 5.75 V for Steady-State Operation) T_A = −40°C		—	580	800	
Input Capacitance	C_{in}				pF
XTAL		—	25	—	
All Other		—	10	—	
Low Voltage Recover	V_{LVR}	—	—	4.75	V
Low Voltage Inhibit	V_{LVI}	2.75	3.75	4.70	V
Input Current					μA
TIMER (V_{in} = 0.4 V)	I_{in}	—	—	20	
INT (V_{in} = 2.4 V to V_{CC})		—	20	50	
EXTAL (V_{in} = 2.4 V to V_{CC} Crystal Option)		—	—	10	
(V_{in} = 0.4 V Crystal Option)		—	—	−1600	
RESET (V_{in} = 0.8 V)	I_{RES}	−4.0	—	−40	
(External Capacitor Charging Current)					

*Due to internal biasing, this input (when unused) floats to approximately 2.0 V.

SWITCHING CHARACTERISTICS (V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Oscillator Frequency	f_{osc}	0.4	—	4.2	MHz
Cycle Time ($4/f_{osc}$)	t_{cyc}	0.95	—	10	μs
INT, INT2, and TIMER Pulse Width	t_{WL}, t_{WH}	t_{cyc} + 250	—	—	ns
RESET Pulse Width	t_{RWL}	t_{cyc} + 250	—	—	ns
RESET Delay Time (External Cap = 1 μF)	t_{RHL}	—	100	—	ms
INT Zero-Crossing Detection Input Frequency	f_{INT}	0.03	—	1.0	kHz
External Clock input Duty Cycle (EXTAL)	—	40	50	60	%
Crystal Oscillator Start-Up Time	—	—	—	100	ms

PORT ELECTRICAL CHARACTERISTICS (V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Port A with CMOS Drive Enabled					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Output High Voltage, I_{Load} = −10 μA	V_{OH}	V_{CC} − 1.0	—	—	V
Input High Voltage, I_{Load} = −300 μA (max.)	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage, I_{Load} = −500 μA (max.)	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current (V_{in} = 2.0 V to V_{CC})	I_{IH}	—	—	−300	μA
Hi-Z State Input Current (V_{in} = 0.4 V)	I_{IL}	—	—	−500	μA
Port B					
Output Low Voltage, I_{Load} = 3.2 mA	V_{OL}	—	—	0.4	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.0	V
Output High Voltage, I_{Load} = −200 μA	V_{OH}	2.4	—	—	V
Darlington Current Drive (Source), V_O = 1.5 V	I_{OH}	−1.0	—	−10	mA
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port C and Port A with TTL Drive					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port C (Open-Drain Option)					
Input High Voltage	V_{IH}	2.0	—	13.0	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Input Leakage Current (V_{in} = 13.0 V)	I_{LOD}	—	<3	15	μA
Output Low Voltage I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Port D (Digital Inputs Only)					
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Input Current	I_{in}	—	<1	5	μA

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola in the following media:

MDOS™, disk file

MS™-DOS/PC-DOS disk file

EPROM(s)MC68705R3, 2532, 2732, or two 2516/2716

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, a sales person, or Motorola representative.

FLEXIBLE DISKS

Several types of flexible disks (MDOS or MS-DOS/PC-DOS disk file), programmed with the customer program (positive logic sense for address and data), may be submitted for pattern generation. In either case, the diskette should be clearly labeled with the customers name, date, project or product name, and the name of the file containing the pattern.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to speed up the process in case of any difficulty with the pattern file.

MDOS Disk File

MDOS is Motorla's Disk Operating System available on the EXORciser® development system. The disk media submitted must be a single-sided, single-density, 8-inch MDOS compatible floppy diskette. The diskette must contain the minimum set of MDOS system files in addition to the pattern file.

The .LO output of the M6805 cross assembler should be furnished. In addition, the file must be produced (using the ROLLOUT command) containing the absolute image of the M6805 memory. Include the entire memory image of both date and program space. All unused bytes, including those in the user space, must be set to zero.

MS-DOS/PC-DOS Disk File

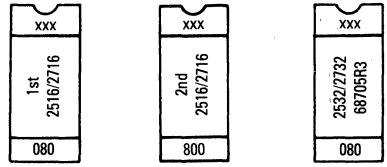
MS-DOS is Microsoft's Disk Operating System. PC-DOS is IBM® Personal Computer (PC) Disk Operating System. Disk media submitted must be a standard density (360K) double-sided 5 1/4 inch compatible floppy diskette. The diskette must contain object file code in Motorola's S-record format. The S-record format is a character-based object file format generated by M6805 cross assemblers and linkers on IBM-PC style machines.

EPROMs

An MC68705R3, 2532, 2732, 2516 (2), or 2716 (2) type EPROM(s), programmed with the customer program (positive logic sense for address and data) may be submitted for pattern generation. Since all program and data space information will fit on one MC68705R3/2532/2732 or two 2516/2716 type EPROM(s), the EPROM(s) must be programmed as described in the following paragraph.

For the 2532, 2732, or the MC68705R3, the ROM code should be located from $080 to $FF and $7C0 to $F37 and the interrupt vectors from $FF8 to $FFF. For the 2516's or 2716's, the ROM code should be located from $080 to $FF and $7C0 to $7FF in the first EPROM and from $0 to $737 in the second EPROM. The interrupt vectors should be in the second EPROM from $7F8 to $FFF.

EPROM MARKING



xxx = Customer ID

VERIFICATION MEDIA

All original pattern media (EPROMs or floppy disk) are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked and the verification form completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for creation of the customer mask. If desired, Motorola will program (customer supplied) blank EPROM(s) or DOS disk from the data file used to create the custom mask to aid in the verification process.

ROM VERIFICATION UNITS (RVUs)

Ten MCUs containing the customer's ROM pattern will be sent for program verification. These units will have been made using the custom mask but are for the purpose of ROM verification only. For expediency the MCUs are unmarked, packaged in ceramic, and tested at room temperature and five volts. These RVUs are free with the minimum order quantity but are not production parts. These RVUs are not guaranteed by Motorola Quality Assurance.

ORDERING INFORMATION

The following table provides generic information pertaining to the package type, temperature, and MC part numbers for the MC6805U2.

Table 4. Generic Information

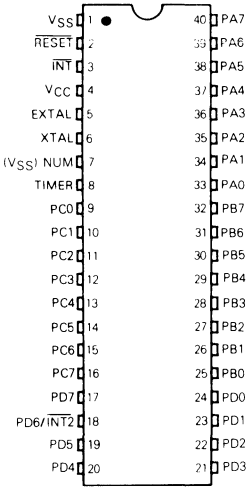
Package Type	Temperature	Part Number
PLCC FN Suffix	0°C to 70°C −40°C to +85°C	MC6805U2FN MC6805U2CFN
Plastic P Suffix	0°C to 70°C −40°C to +85°C	MC6805U2P MC6805U2CP
Cerdip S Suffix	0°C to 70°C −40°C to +85°C	MC6805U2S MC6805U2CS

MDOS is a trademark of Motorola Inc.
MS-DOS is a trademark of Microsoft, Inc.
EXORciser is a registered trademark of Motorola Inc.
IBM is a registered trademark of International Business Machines Corporation.

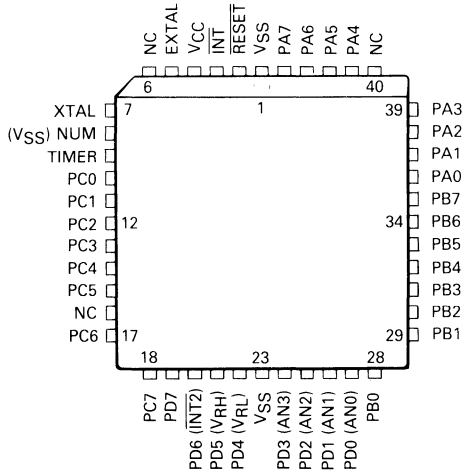
MECHANICAL DATA

PIN ASSIGNMENTS

Dual-in-Line Package



PLCC Package



MC6805U3

Technical Summary
8-Bit Microcontroller Unit

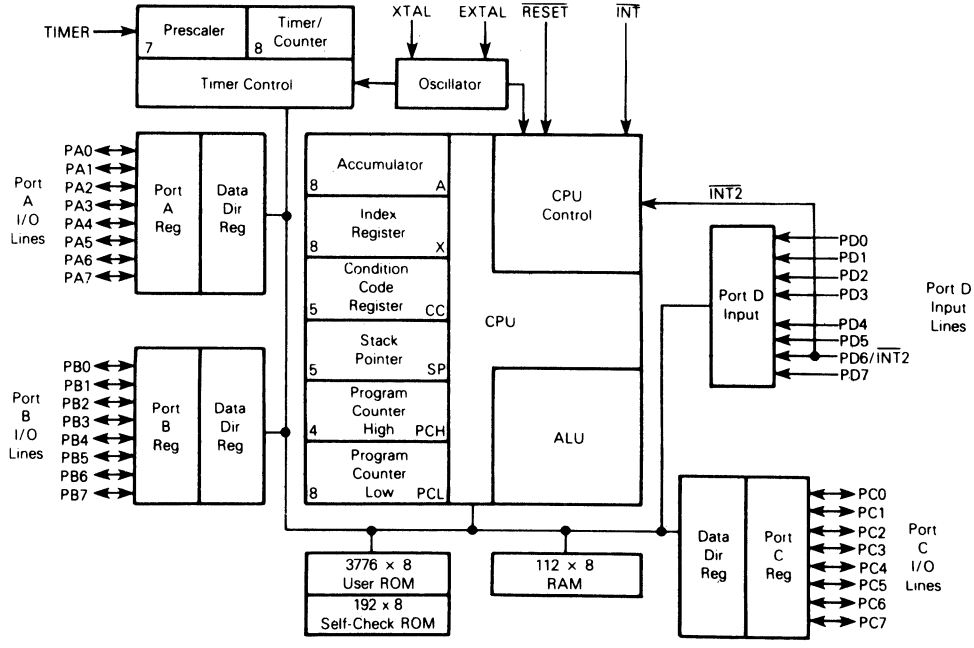
The MC6805U3 (HMOS) Microcontroller Unit (MCU) is a member of the MC6805 Family of microcomputers. This low cost and high-speed MCU has parallel I/O capability with pins programmable as input or output. This publication contains condensed information on the MCU; for detailed information, refer to *M6805 HMOS, M146805 CMOS Family User's Manual* (M6805UM(AD2)) or contact your local Motorola sales office.

Refer to the block diagram for the hardware features and to the list below for additional features available on the MCU.

- Internal 8-Bit Timer with 7-Bit Prescaler
- On-chip Oscillator
- Memory Mapped I/O
- Versatile Interrupt Handling
- True Bit Manipulation
- Bit Test and Branch Instruction
- Vectored Interrupts
- 3776 Bytes of ROM
- 112 Bytes of RAM
- Self-Check Mode
- 24 Bidirectional I/O Lines

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

V_{CC} AND V_{SS}

Power is supplied to the microcomputer using these two pins. V_{CC} is +5.25 volts (±0.5Δ) power, and V_{SS} is ground.

$\overline{INT}$

This pin provides the capability for asynchronously applying an external interrupt to the MCU. Refer to **INTERRUPTS** for more detailed information.

EXTAL, XTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a resistor/capacitor combination, or an external signal (depending upon user selected manufacturing mask option) is connected to these pins to provide a system clock.

RC Oscillator

With this option, a resistor is connected to the oscillator pins as shown in Figure 1. The relationship between R and f_{osc} is shown in Figure 2.

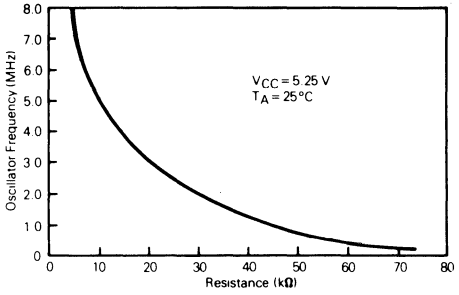
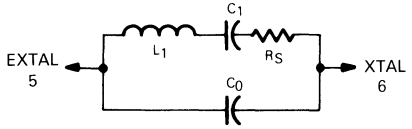


Figure 2. Typical Frequency vs Resistance for RC Oscillator Option Only

Crystal

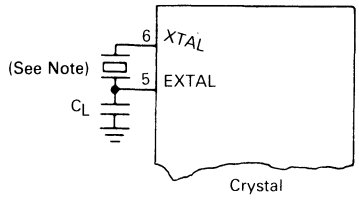
The circuit shown in Figure 1 is recommended when using a crystal. Using an external CMOS oscillator is recommended when crystals outside the specified ranges are to be used. The crystal and components should be

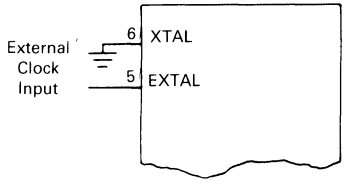
3

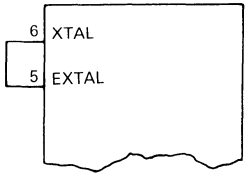


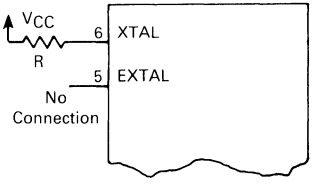
AT – Cut Parallel Resonance Crystal
C_0 = 7 pF Max
Freq. = 4.0 MHz @ C_L = 24 pF
R_S = 50 ohms Max.

Piezoelectric ceramic resonators which have the equivalent specifications may be used instead of crystal oscillators. Follow ceramic resonator manufacturer's suggestions for C_0, C_1, and R_S values.









NOTE: The recommended C_L value with a 4.0 MHz crystal is 27 pF maximum, including system distributed capacitance. There is an internal capacitance of approximately 25 pF on the XTAL pin. For crystal frequencies other than 4 MHz, the total capacitance on each pin should be scaled as the inverse of the frequency ratio. For example, with a 2 MHz crystal, use approximately 50 pF on EXTAL and approximately 25 pF on XTAL. The exact value depends on the motional-arm parameters of the crystal used.

Figure 1. Oscillator Connections

mounted as close as possible to the input pins to minimize output distortion and startup stabilization time. Refer to **ELECTRICAL SPECIFICATIONS** for V_{CC} specifications.

External Clock

An external clock should be applied to the EXTAL input with the XTAL input connected to ground, as shown in Figure 1. This option may only be used with the crystal oscillator option selected in the mask option register. The t_{OXOV} or t_{ILCH} specifications do not apply when using an external clock input.

TIMER

This pin is used as an external input to control the internal timer/counter circuitry. This pin also detects a higher voltage level used to initiate the self-test program.

$\overline{\text{RESET}}$

This pin has a Schmitt trigger input and an on-chip pullup. The MCU can be reset by pulling $\overline{\text{RESET}}$ low.

INPUT/OUTPUT LINES (PA0-PA7, PB0-PB7, PC0-PC7,PD0-PD7)

These 32 lines are arranged into four 8-bit ports (A, B, C, and D). Ports A, B, and C are programmable as either inputs or outputs under software control of the data direction registers. Port D bit 6 shares input signal $\overline{\text{INT2}}$, which is used by external interrupts. Port D is a fixed input port and not controlled by any data register. Refer to **PROGRAMMING** for additional information.

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

Ports A, B, and C are programmable as either input or output under software control of the corresponding data direction register (DDR). Port D lines are input only. The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic one for output and a logic zero for input. On reset, all the DDRs are initialized to a logic zero state to put the ports in the input mode. The port output registers are not initialized on reset and should be written to before setting the DDR bits.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This port write may be used to initialize the data registers and avoid undefined outputs. Care must be exercised when using read-modify-write instructions since the data read corresponds to the pin level if the DDR is an input (zero) and, also, to the latched output when the DDR is an output (one). Refer to Table 1 for I/O functions and to Figure 3 for typical port circuitry.

Table 1. I/O Pin Functions

Data Direction Register Bit	Latched Output Data Bit	Output State	Input To MCU
1	0	0	0
1	1	1	1
0	X	Hi-Z**	Pin

**Ports B and C are three state ports. Port A has optional internal pullup devices to provide CMOS data drive capability.

MEMORY

The MCU is capable of addressing 4096 bytes of memory and I/O registers. The memory map is shown in Figure 4. The locations consist of user ROM, self-check ROM, user RAM, a miscellaneous control register (MR), timer

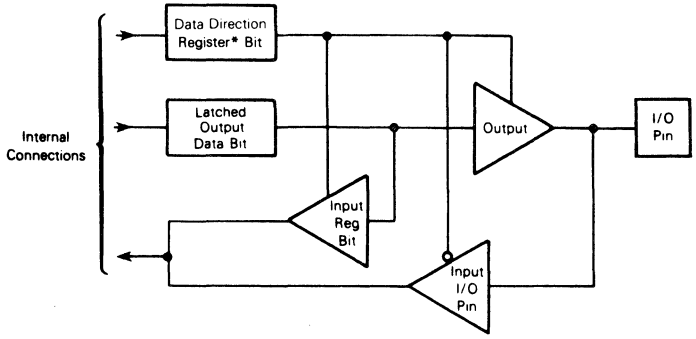
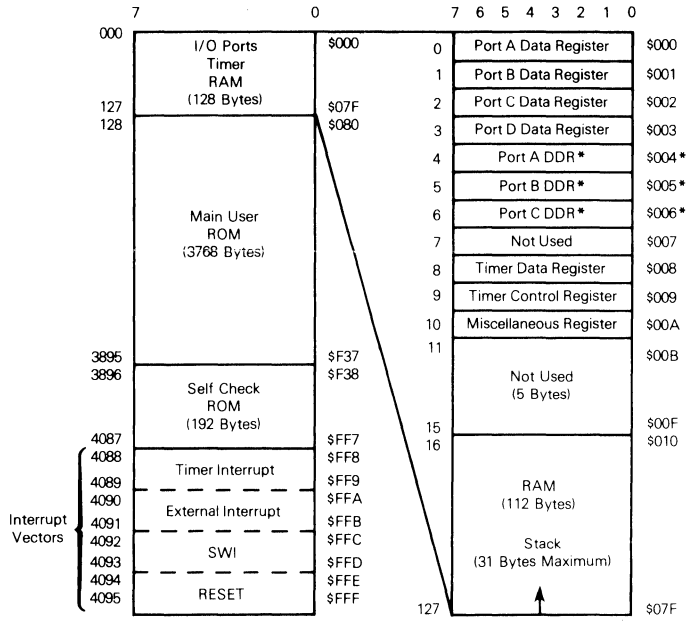


Figure 3. Typical Port I/o Circuitry and Register Configuration



* Caution: Data direction registers (DDRs) are write-only; they read as $FF.

Figure 4. Memory Map

registers, and I/O. The interrupt and reset vectors are located from $FF8 to $FFF.

The stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer decrements during pushes and increments during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

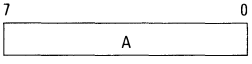
Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

REGISTERS

The MCU contains the registers described in the following paragraphs.

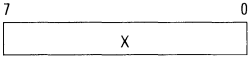
ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



INDEX REGISTER (X)

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16-bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



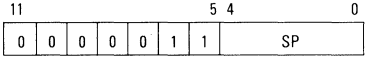
PROGRAM COUNTER (PC)

The program counter is a 12-bit register that contains the address of the next byte to be fetched.

STACK POINTER (SP)

The stack pointer is a 12-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set at location $07F. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

The seven most-significant bits of the stack pointer are permanently set at 0000011. Subroutines and interrupts may be nested down to location $061 (31 bytes maximum), which allows the programmer to use up to 15 levels of subroutine calls (less if interrupts are allowed).



CONDITION CODE REGISTER (CC)

The condition code register is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program, and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.

4				0
H	I	N	Z	C

Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, the timer and external interrupt is masked (disabled). If an external interrupt occurs while this bit is set, the interrupt is latched and is processed as soon as the interrupt bit is cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic one).

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/Borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions, and during shifts and rotates.

SELF CHECK

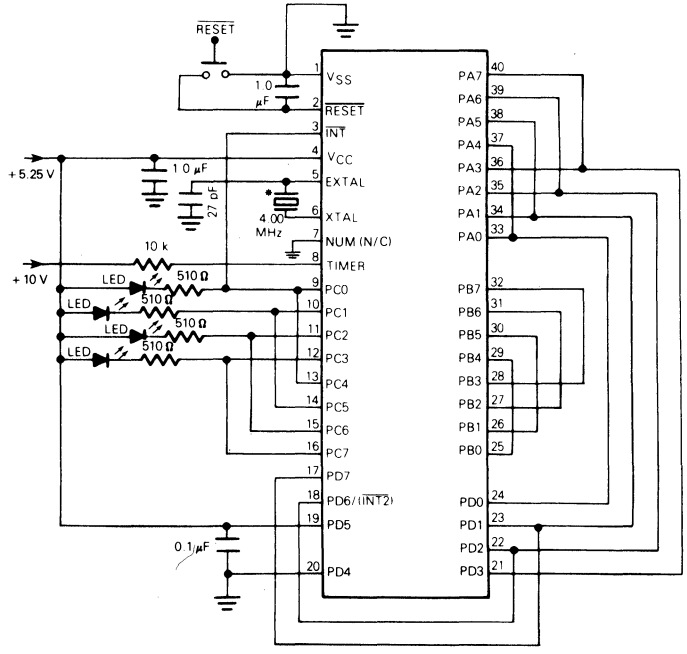
The self check is initiated by connecting the MCU as shown in Figure 5 and then monitoring the output of port C (bit 3) for an oscillation of approximately 7 Hz. The following test are executed automatically:

I/O — Functionally exercise I/O ports;
RAM — Walking bit test;
ROM — Exclusive OR with ODD "1st" parity result;
Timer — Functionally exercise timer; and
Interrupts — Functionally exercise external and timer interrupts.

The RAM and ROM can be called by a user program. The Timer test may be called if the timer input is the internal clock. Table 2 shows the status of the LEDs as a result of a failure. Port C is tested only once (just after reset). If port C fails, no lights will appear.

RESETS

The MCU can be reset three ways: (1) by initial power-up, (2) by the external result input ($\overline{\text{RESET}}$), and (3) by



* This connection depends on clock oscillator user selectable mask option. Use jumper if the RC mask option is selected.

Figure 5. Self-Check Connections

Table 2. Self-Check Error Patterns

LED Meanings				
PC0	PC1	PC2	PC3	Remarks (1:LED ON; 0:LED OFF)
1	0	1	0	Bad I/O
0	0	1	0	Bad Timer
1	1	0	0	Bad RAM
0	1	0	0	Bad ROM
1	0	0	0	Bad A/D
0	0	0	0	Bad Interrupt or Request Flag
All Flashing				Good Device

Anything else Bad Part, Bad Port C, etc.

an optional, internal, low-voltage detect circuit. The $\overline{\text{RESET}}$ input consists mainly of a Schmitt trigger that senses the line logic level.

POWER-ON-RESET (POR)

An internal reset is generated on power-up that allows the internal clock generator to stabilize. The power-on reset is used strictly for power turn-on conditions and should not be used to detect any drop in the power supply voltage. A delay of t_{RHL} milliseconds is required before allowing the $\overline{\text{RESET}}$ input to go high. Connecting a capacitor to the $\overline{\text{RESET}}$ input (Figure 6) typically provides sufficient delay.

EXTERNAL RESET INPUT

The MCU is reset when a logic zero is applied to the $\overline{\text{RESET}}$ input for a period longer than one machine cycle (t_{cyc}). Under this type of reset, the Schmitt trigger switches off at V_{IRES-} to provide an internal reset voltage.

LOW-VOLTAGE INHIBIT (LVI)

The optional low-voltage detection circuit causes a reset of the MCU if the power supply voltage falls below a certain level (V_{LVI}). The only requirement is that the V_{CC} must remain at or below the V_{LVI} threshold for one t_{cyc} minimum.

In typical applications, the V_{CC} bus filter capacitor will eliminate negative-going voltage glitches of less than one t_{cyc}. The output from the low-voltage detector is connected directly to the internal reset circuitry. It also forces the $\overline{\text{RESET}}$ pin low via a strong discharge device through a resistor. The internal reset is removed once the power supply voltage rises above a recovery level (V_{LVR}) at which time a normal power-on reset occurs.

INTERRUPTS

The MCU can be interrupted four different ways: (1) through the external interrupt $\overline{\text{IRQ}}$ input pin, (2) with the internal timer interrupt request, (3) using the software interrupt instruction (SWI), or (4) the external port D bit 6 ($\overline{\text{INT2}}$) input pin.

Interrupts cause the processor registers to be saved on the stack and the interrupt mask (I bit) to be set preventing additional interrupts. The RTI instruction causes the register contents to be recovered from the stack, and then normal processing resumes. The stacking order is shown in Figure 7.

Unlike $\overline{\text{RESET}}$, hardware interrupts do not cause the current instruction execution to be halted but are considered pending until the current instruction is complete.

NOTE

The current instruction is considered to be the one already fetched and being operated on.

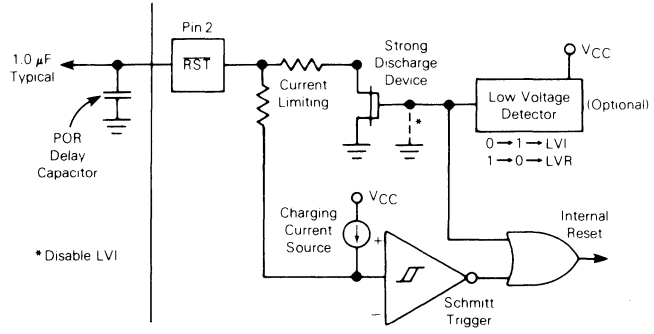
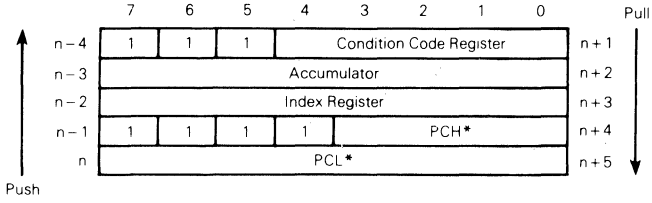


Figure 6. $\overline{\text{RESET}}$ Configuration



*For subroutine calls, only PCH and PCL are stacked.

Figure 7. Interrupt Stacking Order

When the current instruction is complete, the processor checks all pending hardware interrupts and, if unmasked (I bit clear), proceeds with interrupt processing; otherwise, the next instruction is fetched and executed. Masked interrupts are latched for later interrupt service. If the timer interrupt status bit is cleared before unmasking the interrupt, then the interrupt is not latched.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction regardless of the setting of the I bit. Refer to Figure 8 for the reset and interrupt instruction processing sequence.

TIMER INTERRUPT

If the timer mask bit (TCR6) is cleared, then, each time the timer decrements to zero (transitions from $01 to $00), an interrupt request is generated. The actual processor interrupt is generated only if the interrupt mask bit of the condition code register (CCR) is also cleared. When the

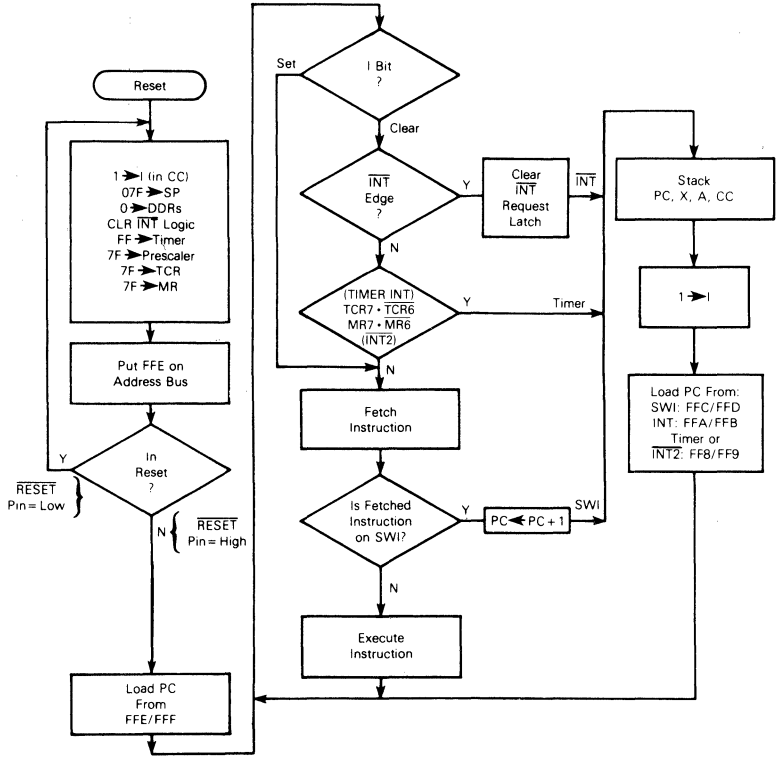


Figure 8. Reset and Interrupt Processing Flowchart

interrupt is recognized, the current state of the machine is pushed onto the stack and the I bit in the CCR is set, masking further interrupts until the present one is serviced. The contents of the timer interrupt vector, containing the location of the timer interrupt service routine, is then loaded into the program counter. At the end of the timer interrupt service routine, the software normally executes an RTI instruction which restores the machine state and starts executing the interrupted program.

EXTERNAL INTERRUPT

The external interrupt is internally synchronized and then latched on the falling edge of $\overline{INT}$ and $\overline{INT2}$. Clearing the I bit enables the external interrupt. The $\overline{INT2}$ interrupt has an interrupt request bit (bit 7) and a mask bit (bit 6) in the miscellaneous register (MR). The $\overline{INT2}$ interrupt is inhibited when the mask bit is set. The $\overline{INT2}$ is always read as a digital input on port D. The $\overline{INT2}$ and timer interrupt request bits, if set, cause the MCU to process and interrupt when the condition code I bit is clear. The following paragraphs describe two typical external interrupt circuits.

Zero-Crossing Interrupt

A sinusoidal input signal (f_{INT} maximum) can be used to generate an external interrupt (see Figure 9a) for use as a zero-crossing detector (for negative transitions of the ac sinusoid). This type of circuit allows applications such as servicing time-of-day routines and engaging/disengaging ac power control devices. Off-chip, full-wave rectification provides an interrupt at every zero crossing of the ac signal and, thereby, provides a 2f clock.

Digital-Signal Interrupt

With this type of circuit (Figure 9b), the $\overline{INT}$ pin can be driven by a digital signal. The maximum frequency of a signal that can be recognized by the TIMER or $\overline{INT}$ pin logic is dependent on the parameter labeled t_{WL}, t_{WH}. Refer to **TIMER** for additional information.

SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. If the I bit is zero, SWI executes after the other interrupts. The SWI execution is similar to the hardware interrupts.

TIMER

The MCU consists of an 8-bit software programmable counter driven by a 7-bit software programmable prescaler. The various timer sources are made via the timer control register (TCR) and mask option register (MOR). The 8-bit counter may be loaded under program control and is decremented toward zero. When the timer reaches zero, the timer interrupt request bit (bit 7) in the timer control register (TCR) is set. Refer to Figure 10 for timer block diagram.

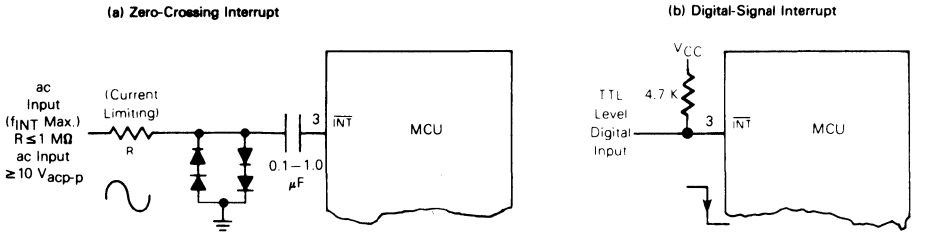
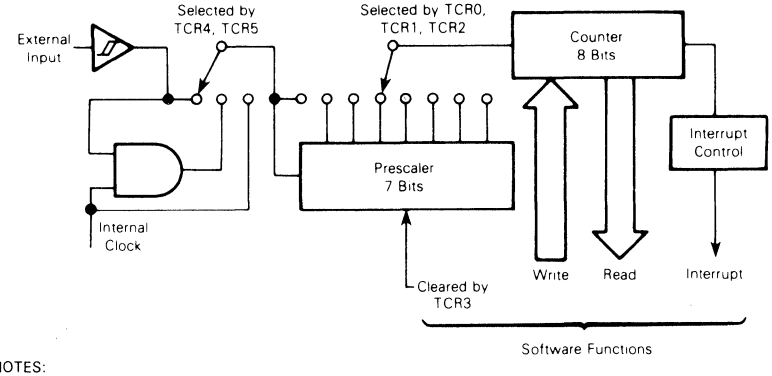


Figure 9. Typical Interrupt Circuits



NOTES:
1. The prescaler and 8-bit counter are clocked on the rising edge of the internal clock (phase two) or external input.
2. The counter is written to during data strobe (DS) and counts down continuously.

Figure 10. Timer Block Diagram

The timer interrupt can be masked (disabled) by setting the timer interrupt mask bit (bit 6) in the TCR. When the I bit in the condition code register is cleared, the processor receives the interrupt. The MCU responds to this interrupt by (1) saving the present CPU state on the stack, (2) fetching the timer interrupt vector, and (3) executing the interrupt routine. The timer interrupt request bit must be cleared by software. Refer to **RESETS** and **INTERRUPTS** for additional information.

The prescaler is a 7-bit divider which is used to extend the maximum length of the timer. To avoid truncation errors, the prescaler is cleared when TCR bit 3 is set to a logic one; however, the TCR bit 3 always reads as a logic zero to ensure proper operation with read-modify-write instructions.

The timer continues to count past zero, falling from $00 through $FF, and continues the countdown. The counter can be read at any time by reading the timer data register (TDR). This allows a program to determine the length of time since a timer interrupt has occurred without disturbing the counting process.

3

SOFTWARE CONTROLLED MODE

The timer prescaler input can be configured for three different operating modes plus a disable mode, depending on the value written to TCR control bits 4 and 5 (TCR4 and TCR5). The following paragraphs describe the different modes.

Timer Input Mode 1

When TCR4 and TCR5 are both programmed to zero, the timer input is from the internal clock (phase two) and the timer input pin is disabled. The internal clock mode can be used for periodic interrupt generation as well as a reference for frequency and event measurement. During the WAIT instruction, the internal clock to the timer continues to run at its normal rate.

Timer Input Mode 2

When TCR4 = 1 and TCR5 = 0, the internal clock and the timer input signals are ANDed to form the timer input. This mode can be used to measure external pulse widths. The active high, external pulse gates in the internal clock for the duration of the external pulse. The accuracy of the count is ±1.

Timer Input Mode 3

When TCR4 = 0 and TCR5 = 1, no prescaler input frequency is applied to the prescaler and the timer is disabled.

Timer Input Mode 4

When TCR4 and TCR5 are both one, the timer input is from the external clock. The external clock can be used to count external events as well as to provide an external frequency for generating periodic interrupts.

TIMER CONTROL REGISTER (TCR) $009

This is an 8-bit register that controls various functions such as configuring operation mode, setting ratio of the prescaler, and generating timer interrupt request signal. Bit 3 is write only.

7	6	5	4	3	2	1	0
TCR7	TCR6	TCR5	TCR4	TCR3	TCR2	TCR1	TCR0
RESET:							
0	1	U	U	U	U	U	U

TCR7 — Timer Interrupt Request
Used to indicate the timer interrupt when it is logic one
1 = Set when the timer data register changes to all zeros
0 = Cleared by external reset, power-on reset, or under program control

TCR6 — Timer Interrupt Mask
Used to inhibit the timer interrupt
1 = Interrupt inhibited
0 = Interrupt enabled

TCR5 — External or Internal
Selects input clock source
1 = External clock selected
0 = Internal clock selected ($f_{OSC}/4$)

TCR4 — TIMER External Enable
Used to enable external TIMER pin
1 = Enables external timer pin
0 = Disables external timer pin

TCR3 — Prescaler Clear
Write only bit. Writing a one to this bit resets the prescaler to zero. A read of this location always indicates a zero.

TCR2, TCR1, TCR0 — Prescaler Select Bits
Decoded to select one of eight outputs of the prescaler

Prescaler

TCR2	TCR1	TCR0	Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

INSTRUCTION SET

The MCU has a set of 59 basic instructions which can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and

jump to subroutine (JSR) instructions have no register operand. Refer to the following list of instructions.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Higher	BHI
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
(Branch if Higher or Same)	(BHS)
Branch if Carry Set	BCS
(Branch if Lower)	(BLO)
Branch if Not Equal	BNE
Branch if Equal	BEQ
Branch if Half Carry Clear	BHCC
Branch if Half Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Bit is Clear	BMC
Branch if Interrupt Mask Bit is Set	BMS
Branch if Interrupt Line is Low	BIL
Branch if Interrupt Line is High	BIH
Branch to Subroutine	BSR

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (2's Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST

3

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No-Operation	NOP

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the first 256 bytes of the memory space where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear and bit test, and branch functions are all implemented with a single instruction. For test and branch instructions, the value of

the bit tested is also placed in the carry bit of the condition code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch if Bit n is Set	BRSET n (n=0 . . . 7)
Branch if Bit n is Clear	BRCLR n (n=0 . . . 7)
Set Bit n	BSET n (n=0 . . . 7)
Clear Bit n	BCLR n (n=0 . . . 7)

OPCODE MAP SUMMARY

Table 3 is an opcode map for the instructions used on the MCU.

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single byte instructions while the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. Two-byte direct addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction.

RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −126 to +129 from the opcode address.

INDEX, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. This addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this two-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This addressing mode can be used in a manner similar to indexed, 8-bit offset except that this three-byte instruction allows tables to be anywhere in memory.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte to which the specified bit is to be set or cleared. Any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single two-byte instruction.

CAUTION

The corresponding DDRs for ports A, B, and C are write only registers (registers at $004, $005, and $006). A read operation on these registers is undefined. Since BSET and BCLR are read-modify-write functions, these instructions cannot be used to set or clear a DDR bit (all "unaffected" bits would be set). It is recommended that all DDR bits in a port be written using a single-store instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit to be tested, and its condition (set or clear), is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte.

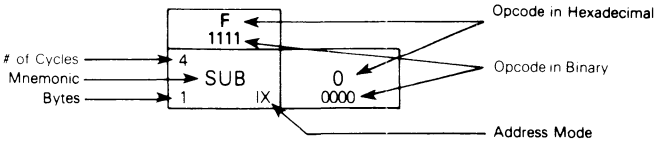
Table 3. Opcode Map

Hi / Low	Bit Manipulation BTB 0 0000	Bit Manipulation BSC 1 0001	Branch REL 2 0010	Read-Modify-Write DIR 3 0011	Read-Modify-Write INH 4 0100	Read-Modify-Write INH 5 0101	Read-Modify-Write IX1 6 0110	Read-Modify-Write IX 7 0111	Control INH 8 1000	Control INH 9 1001	Register/Memory IMM A 1010	Register/Memory DIR B 1011	Register/Memory EXT C 1100	Register/Memory IX2 D 1101	Register/Memory IX1 E 1110	Register/Memory IX F 1111	Hi / Low
0 0000	10 BRSET0 3 BTB	7 BSET0 2 BSC	4 BRA 2 REL	6 NEG 2 DIR	4 NEG 1 INH	4 NEG 1 INH	7 NEG 2 IX1	6 NEG 1 IX	9 RTI 1 INH		2 SUB 2 IMM	4 SUB 2 DIR	5 SUB 3 EXT	6 SUB 3 IX2	5 SUB 2 IX1	4 SUB 1 IX	0 0000
1 0001	10 BRCLR0 3 BTB	7 BCLR0 2 BSC	4 BRN 2 REL						6 RTS 1 INH		2 CMP 2 IMM	4 CMP 2 DIR	5 CMP 3 EXT	6 CMP 3 IX2	5 CMP 2 IX1	4 CMP 1 IX	1 0001
2 0010	10 BRSET1 3 BTB	7 BSET1 2 BSC	4 BHI 2 REL								2 SBC 2 IMM	4 SBC 2 DIR	5 SBC 3 EXT	6 SBC 3 IX2	5 SBC 2 IX1	4 SBC 1 IX	2 0010
3 0011	10 BRCLR1 3 BTB	7 BCLR1 2 BSC	4 BLS 2 REL	6 COM 2 DIR	4 COMA 1 INH	4 COMX 1 INH	7 COM 2 IX1	6 COM 1 IX	11 SWI 1 INH		2 CPX 2 IMM	4 CPX 2 DIR	5 CPX 3 EXT	6 CPX 3 IX2	5 CPX 2 IX1	4 CPX 1 IX	3 0011
4 0100	10 BRSET2 3 BTB	7 BSET2 2 BSC	4 BCC 2 REL	6 LSR 2 DIR	4 LSRA 1 INH	4 LSRX 1 INH	7 LSR 2 IX1	6 LSR 1 IX			2 AND 2 IMM	4 AND 2 DIR	5 AND 3 EXT	6 AND 3 IX2	5 AND 2 IX1	4 AND 1 IX	4 0100
5 0101	10 BRCLR2 3 BTB	7 BCLR2 2 BSC	4 BCS 2 REL								2 BIT 2 IMM	4 BIT 2 DIR	5 BIT 3 EXT	6 BIT 3 IX2	5 BIT 2 IX1	4 BIT 1 IX	5 0101
6 0110	10 BRSET3 3 BTB	7 BSET3 2 BSC	4 BNE 2 REL	6 ROR 2 DIR	4 RORA 1 INH	4 RORX 1 INH	7 ROR 2 IX1	6 ROR 1 IX			2 LDA 2 IMM	4 LDA 2 DIR	5 LDA 3 EXT	6 LDA 3 IX2	5 LDA 2 IX1	4 LDA 1 IX	6 0110
7 0111	10 BRCLR3 3 BTB	7 BCLR3 2 BSC	4 BEQ 2 REL	6 ASR 2 DIR	4 ASRA 1 INH	4 ASRX 1 INH	7 ASR 2 IX1	6 ASR 1 IX		2 TAX 1 INH		5 STA 2 DIR	6 STA 3 EXT	7 STA 3 IX2	6 STA 2 IX1	5 STA 1 IX	7 0111
8 1000	10 BRSET4 3 BTB	7 BSET4 2 BSC	4 BHCC 2 REL	6 LSL 2 DIR	4 LSLA 1 INH	4 LSLX 1 INH	7 LSL 2 IX1	6 LSL 1 IX		2 CLC 1 INH	2 EOR 2 IMM	4 EOR 2 DIR	5 EOR 3 EXT	6 EOR 3 IX2	5 EOR 2 IX1	4 EOR 1 IX	8 1000
9 1001	10 BRCLR4 3 BTB	7 BCLR4 2 BSC	4 BHCS 2 REL	6 ROL 2 DIR	4 ROLA 1 INH	4 ROLX 1 INH	7 ROL 2 IX1	6 ROL 1 IX		2 SEC 1 INH	2 ADC 2 IMM	4 ADC 2 DIR	5 ADC 3 EXT	6 ADC 3 IX2	5 ADC 2 IX1	4 ADC 1 IX	9 1001
A 1010	10 BRSET5 3 BTB	7 BSET5 2 BSC	4 BPL 2 REL	6 DEC 2 DIR	4 DECA 1 INH	4 DECX 1 INH	7 DEC 2 IX1	6 DEC 1 IX		2 CLI 1 INH	2 ORA 2 IMM	4 ORA 2 DIR	5 ORA 3 EXT	6 ORA 3 IX2	5 ORA 2 IX1	4 ORA 1 IX	A 1010
B 1011	10 BRCLR5 3 BTB	7 BCLR5 2 BSC	4 BMI 2 REL							2 SEI 1 INH	2 ADD 2 IMM	4 ADD 2 DIR	5 ADD 3 EXT	6 ADD 3 IX2	5 ADD 2 IX1	4 ADD 1 IX	B 1011
C 1100	10 BRSET6 3 BTB	7 BSET6 2 BSC	4 BMC 2 REL	6 INC 2 DIR	4 INCA 1 INH	4 INCX 1 INH	7 INC 2 IX1	6 INC 1 IX		2 RSP 1 INH		3 JMP 2 DIR	4 JMP 3 EXT	5 JMP 3 IX2	4 JMP 2 IX1	3 JMP 1 IX	C 1100
D 1101	10 BRCLR6 3 BTB	7 BCLR6 2 BSC	4 BMS 2 REL	6 TST 2 DIR	4 TSTA 1 INH	4 TSTX 1 INH	7 TST 2 IX1	6 TST 1 IX		2 NOP 1 INH	8 BSR 2 REL	7 JSR 2 DIR	8 JSR 3 EXT	9 JSR 3 IX2	8 JSR 2 IX1	7 JSR 1 IX	D 1101
E 1110	10 BRSET7 3 BTB	7 BSET7 2 BSC	4 BIL 2 REL								2 LDX 2 IMM	4 LDX 2 DIR	5 LDX 3 EXT	6 LDX 3 IX2	5 LDX 2 IX1	4 LDX 1 IX	E 1110
F 1111	10 BRCLR7 3 BTB	7 BCLR7 2 BSC	4 BIH 2 REL	6 CLR 2 DIR	4 CLRA 1 INH	4 CLRX 1 INH	7 CLR 2 IX1	6 CLR 1 IX		2 TXA 1 INH		5 STX 2 DIR	6 STX 3 EXT	7 STX 3 IX2	6 STX 2 IX1	5 STX 1 IX	F 1111

Abbreviations for Address Modes

INH	Inherent
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage Self-Check Mode (TIMER Pin Only)	V_{in}	−0.3 to +7.0 −0.3 to +15.0	V
Operating Temperature Range MC6805U3 MC6805U3C MC6805U3V	T_A	T_L to T_H 0 to 70 −40 to +85 −40 to +105	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature Plastic PLCC Cerdip	T_J	 150 150 175	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation, it is recommended the V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{CC}$. Reliability of operation is enhanced if unused inputs except EXTAL are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic (P Suffix) PLCC (FN Suffix) Cerdip (S Suffix)	θ_{JA}	 60 100 60	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

ELECTRICAL CHARACTERISTICS (V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage	V_{IH}				V
$\overline{RESET}$ (4.75 ≤ V_{CC} ≤ 5.75)		4.0	—	V_{CC}	
V_{CC} < 4.75)		V_{CC} − 0.5	—	V_{CC}	
$\overline{INT}$ (4.75 ≤ V_{CC} ≤ 5.75)		4.0	*	V_{CC}	
(V_{CC} < 4.75)		V_{CC} − 0.5	*	V_{CC}	
(All Other (Except Timer)		2.0	—	V_{CC}	
Input High Voltage Timer	V_{IH}				V
Timer Mode		2.0	—	V_{CC} + 1.0	
Self-Check Mode		9.0	10.0	15.0	
Input Low Voltage	V_{IL}				V
$\overline{RESET}$		V_{SS}	—	0.8	
$\overline{INT}$		V_{SS}	*	1.5	
All Other		V_{SS}	—	0.8	
$\overline{RESET}$ Hysteresis Voltages					V
"Out of Reset"	V_{IRES+}	2.1	—	4.0	
"Into Reset"	V_{IRES-}	0.8	—	2.0	
$\overline{INT}$ Zero-Crossing Voltage, Through a Capacitor	V_{INT}	2	—	4	$V_{ac\ p\text{-}p}$
Internal Power Dissipation — (No Port Loading, T_A = 0°C	P_{INT}	—	520	740	mW
V_{CC} = 5.75 V for Steady-State Operation) T_A = −40°C		—	580	800	
Input Capacitance	C_{in}				pF
XTAL		—	25	—	
All Other		—	10	—	
Low Voltage Recover	V_{LVR}	—	—	4.75	V
Low Voltage Inhibit	V_{LVI}	2.75	3.75	4.70	V
Input Current					μA
$\overline{TIMER}$ (V_{in} = 0.4)		—	—	20	
$\overline{INT}$ (V_{in} = 2.4 V to V_{CC})		—	20	50	
EXTAL (V_{in} = 2.4 V to V_{CC} Crystal Option)	I_{in}	—	—	10	
(V_{in} = 0.4 V Crystal Option)		—	—	−1600	
$\overline{RESET}$ (V_{in} = 0.8 V)	I_{RES}	−4.0	—	−40	
(External Capacitor Charging Current)					

*Due to internal biasing this input (when unused) floats to approximately 2.0 V.

SWITCHING CHARACTERISTICS (V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Characteristic	Symbol	Min	Typ	Max	Unit
Oscillator Frequency	f_{osc}	0.4	—	4.2	MHz
Cycle time ($4/f_{osc}$)	t_{cyc}	0.95	—	10	μs
$\overline{INT}$, $\overline{INT2}$, and TIMER Pulse Width	t_{WL}, t_{WH}	t_{cyc} + 250	—	—	ns
$\overline{RESET}$ Pulse Width	t_{RWL}	t_{cyc} + 250	—	—	ns
$\overline{RESET}$ Delay Time (External Capacitance = 1 μF)	t_{RHL}	—	100	—	ms
$\overline{INT}$ Zero-Crossing Detection Input Frequency	f_{INT}	0.03	—	1.0	kHz
External Clock Input Duty Cycle (EXTAL)	—	40	50	60	%
Crystal Oscillator Start-Up Time	—	—	—	100	ms

PORT ELECTRICAL CHARACTERISTICS (V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Port A with CMOS Drive Enabled					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Output High Voltage, I_{Load} = −10 μA	V_{OH}	V_{CC} − 1.0	—	—	V
Input High Voltage, I_{Load} = −300 μA (max.)	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage, I_{Load} = −500 μA (max.)	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current (V_{in} = 2.0 V to V_{CC})	I_{IH}	—	—	−300	μA
Hi-Z State Input Current (V_{in} = 0.4 V)	I_{IL}	—	—	−500	μA
Port B					
Output Low Voltage, I_{Load} = 3.2 mA	V_{OL}	—	—	0.4	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.0	V
Output High Voltage, I_{Load} = −200 μA	V_{OH}	2.4	—	—	V
Darlington Current Drive (Source), V_O = 1.5 V	I_{OH}	−1.0	—	−10	mA
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port C and Port A with TTL Drive					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port C (Open-Drain Option)					
Input High Voltage PC0-PC6	V_{IH}	2.0	—	13.0	V
Input High Voltage PC7	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Input Leakage Current (V_{in} = 13.0 V)	I_{LOD}	—	<3	15	μA
Output Low Voltage I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Port D (Digital Inputs Only)					
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Input Current	I_{in}	—	<1	5	μA

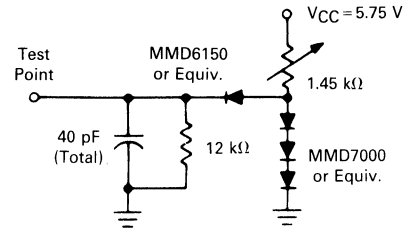


Figure 11. TTL Equivalent Test Load (Port B)

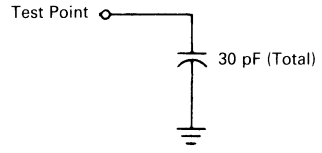


Figure 12. CMOS Equivalent Test Load (Port A)

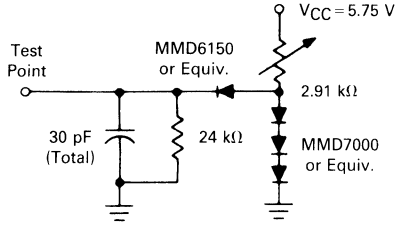


Figure 13. TTL Equivalent Test Load (Ports A and C)

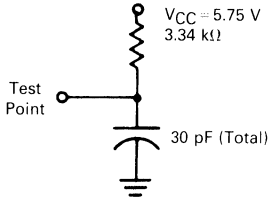


Figure 14. Open-Drain Equivalent Test Load (Port C)

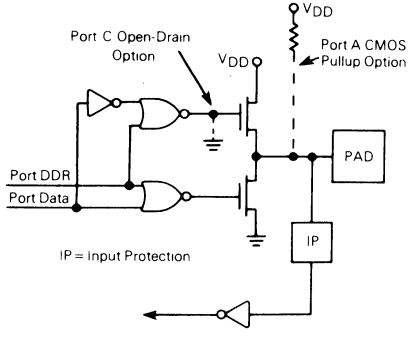


Figure 15. Ports A and C Logic Diagram

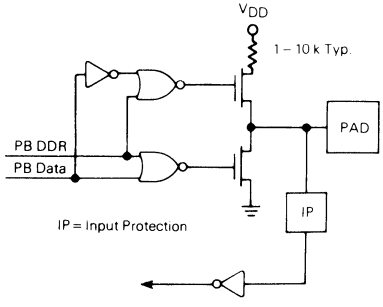


Figure 16. Port B Logic Diagram

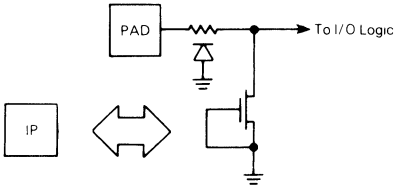


Figure 17. Typical Input Protection

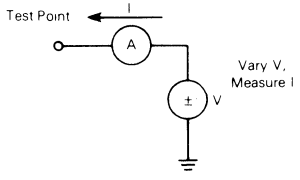


Figure 18. I/O Characteristic Measurement Circuit

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola in the following media:

MDOS, disk file
MS-DOS/PC-DOS disk file
EPROM(s) MC68705U3, 2532, 2732, or two 2516/2716

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, a sales person, or a Motorola representative.

FLEXIBLE DISKS

Several types of flexible disks (MDOS™ or MS™-DOS/PC-DOS disk file), programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. In either case, the diskette should be clearly labeled with the customer's name, date, project or product name, and the name of the file containing the pattern.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to expedite the process in case of any difficulty with the pattern file.

MDOS Disk File

MDOS is Motorola's Disk Operating System available on the EXORciser® development system. The disk media submitted must be a single-side, single-density, 8-inch MDOS compatible floppy diskette. The diskette must contain the minimum set of MDOS system files in addition to the pattern file.

The .LO output of the M6805 cross assembler should be furnished. In addition, the file must be produced (using the ROLLOUT command) containing the absolute image of the M6805 memory. Include the entire memory image of both data and program space. All unused bytes, including those in the user space, must be set to zero.

MS-DOS/PC-DOS Disk File

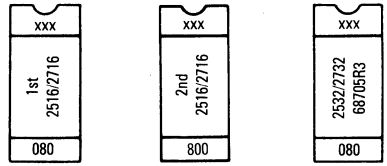
MS-DOS is Microsoft's Disk Operating System. PC-DOS is IBM® Personal Computer (PC) Disk Operating System. Disk media submitted must be a standard density (360K) double-sided 5 1/4 inch compatible floppy diskette. The diskette must contain object file code in Motorola's S-record format. The S-record format is a character-based object file format generated by M6805 cross assemblers and linkers on IBM PC style machines.

EPROMs

A MC68705U3, 2532, 2732, 2516 (2), or 2716 (2) type EPROM(s), programmed with the customer's program (positive logic sense for address and data) may be submitted for pattern generation. Since all program and data space information will fit on one MC68705U3/2532/2732 or two 2516/2716 type EPROM(s), the EPROM(s) must be programmed as described in the following paragraph.

For the 2532, 2732, or MC68705U3, the ROM code should be located from $080 to $F37 and the interrupt vectors from $FF8 to $FFF. For the 2516s or 2716s, the ROM code should be located from $080 to $7FF in the first EPROM and from $0 to $737 in the second EPROM. The interrupt vectors should be in the second EPROM from $7F8 to $7FF.

EPROM MARKING



xxx = Customer ID

VERIFICATION MEDIA

All original pattern media (EPROMs or floppy disk) are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked and the verification form should be completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for creation of the customer's mask. To aid in the verification process, Motorola will program ***customer supplied*** blank EPROM(s) or DOS disk from the data file used to create the custom mask.

ROM VERIFICATION UNITS (RVUs)

Ten MCUs containing the customer's ROM pattern will be sent for program verification. These units will have been made using the custom mask but are for the purpose of ROM verification only. For expediency, the MCUs are usually unmarked, packaged in ceramic, and tested at room temperature and at five volts. These RVUs are free with the minimum order quantity but are not production parts. These RVUs are not guaranteed by Motorola Quality Assurance.

ORDERING INFORMATION

The following table provides generic information pertaining to the package type, temperature, and MC order numbers for the MC6805U3.

Table 5. Generic Information

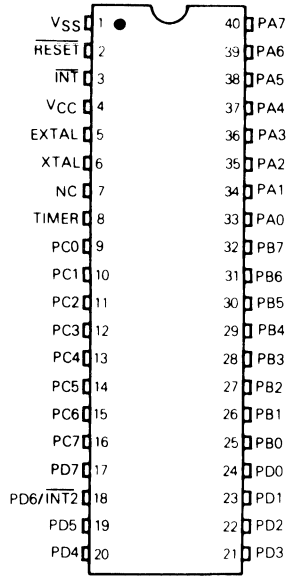
Package Type	Temperature	Order Number
Plastic (P Suffix)	0°C to 70°C −40°C to +85°C	MC6805U3P MC6805U3CP
Cerdip S Suffix	0°C to 70°C −40°C to +85°C	MC6805U3S MC6805U3CS
PLCC FN Suffix	0°C to 70°C −40°C to +85°C	MC6805U3FN MC6805U3CFN

MDOS is a trademark of Motorola Inc.
MS is a trademark of Microsoft, Inc.
EXORciser is a registered trademark of Motorola Inc.
IBM is a registered trademark of International Business Machines Corporation.

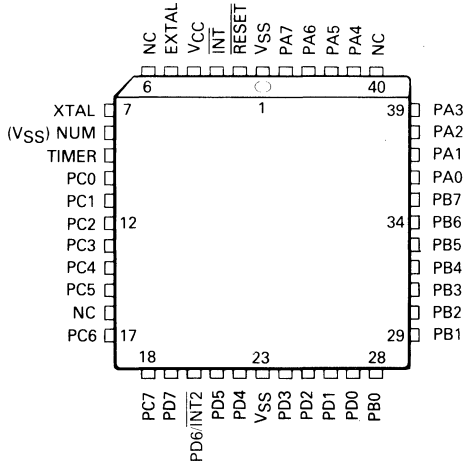
MECHANICAL DATA

PIN ASSIGNMENTS

Dual-in-Line Package



PLCC Package



MC68705P3

Technical Summary

8-Bit EPROM Microcomputer Unit

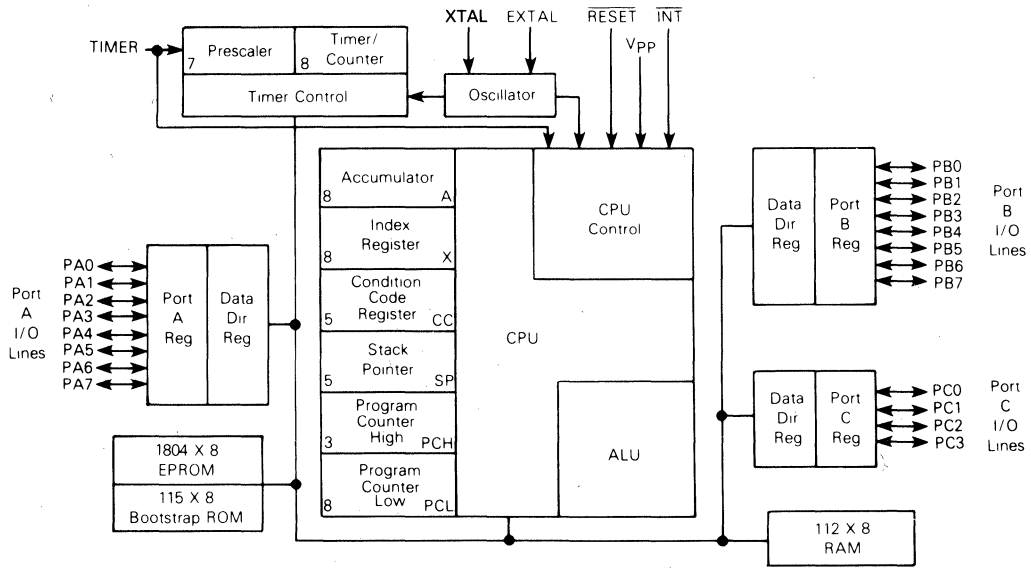
The MC68705P3 (High-Density NMOS) Microcomputer Unit (MCU) is an EPROM member of the MC6805 Family of microcomputers. The user programmable EPROM allows program changes and lower volume applications. This low cost MCU has parallel I/O capability with pins programmable as input or output. This publication contains condensed information on the MCU; for detailed information, refer to *M6805 HMOS, M146805 CMOS Family User's Manual* (M6805UM(AD2)) or contact your local Motorola sales office.

Refer to the block diagram for the hardware features and to the list below for additional features available on the MCU.

- Internal 8-Bit Timer with 7-Bit Programmable Prescaler
- On-chip Oscillator
- Memory Mapped I/O
- Versatile Interrupt Handling
- Bit Manipulation
- Bit Test and Branch Instruction
- Vectored Interrupts
- Bootstrap program in ROM
- 1804 Bytes EPROM
- 112 Bytes RAM
- 20 TTL/CMOS Compatible Bidirectional I/O Lines

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

V_{CC} AND V_{SS}

Power is supplied to the microcomputer using these two pins. V_{CC} is +5.25 volts (±0.5Δ) power, and V_{SS} is ground.

V_{PP}

This pin is used when programming the EPROM. In normal operation, this pin is connected to V_{CC}.

$\overline{INT}$

This pin provides the capability for asynchronously applying an external interrupt to the MCU. Refer to **INTERRUPTS** for more detailed information.

EXTAL, XTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a resistor/capacitor combination, or an external signal is connected to these pins to provide a system clock. Selection is made by the CLK bit in the mask option register.

RC Oscillator

With this option, a resistor is connected to the oscillator pins as shown in Figure 1. The relationship between R and f_{osc} is shown in Figure 2.

Crystal

The circuit shown in Figure 1 is recommended when using a crystal. Using an external CMOS oscillator is recommended when crystals outside the specified ranges

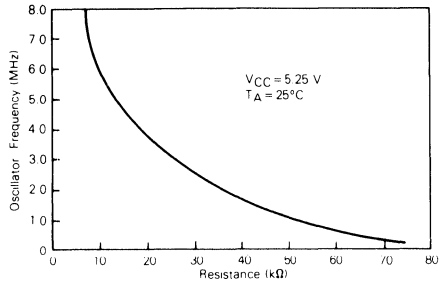


Figure 2. Typical Frequency vs Resistance for RC Oscillator Option only

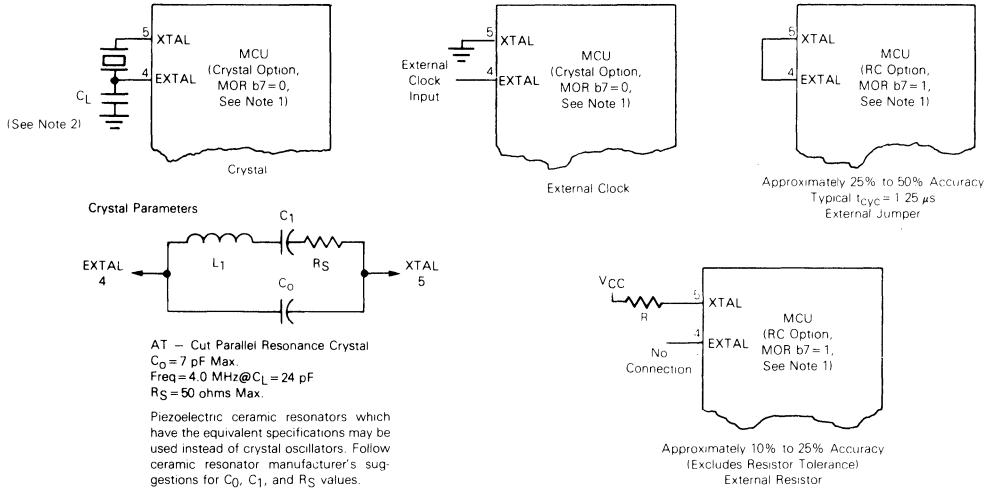
are to be used. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time.

External Clock

An external clock should be applied to the EXTAL input with the XTAL input connected to ground, as shown in Figure 1. This option may only be used with the crystal oscillator option selected in the mask option register.

TIMER

This pin is used as an external input to control the internal timer/counter circuitry. This pin also detects a



NOTES:

1. When the TIMER input pin is in the V_{IHTP} range (in the bootstrap EPROM programming mode), the crystal option is forced. When the TIMER input is at or below V_{CC}, the clock generator option is determined by bit 7 of the mask option register (CLK).
2. The recommended C_L value with a 4.0 MHz crystal is 27 pF maximum including system distributed capacitance. There is an internal capacitance of approximately 25 pF on the XTAL pin. For crystal frequencies other than 4 MHz, the total capacitance on each pin should be scaled as the inverse of the frequency ratio. For example, with a 2 MHz crystal, use approximately 50 pF on EXTAL and approximately 25 pF on XTAL. The exact value depends on the motional-arm parameters of the crystal used.

Figure 1. Oscillator Connections

higher voltage level used to initiate the bootstrap program.

$\overline{\text{RESET}}$

This pin has a Schmitt trigger input and an on-chip pullup. The MCU can be reset by pulling $\overline{\text{RESET}}$ low. Refer to **RESETS** section for more detail.

INPUT/OUTPUT LINES (PA0-PA7, PB0-PB7, PC0-PC3)

These 20 lines are arranged into two 8-bit ports (A and B) and one 4-bit port (C). All lines are programmable as either inputs or outputs under software control of the data direction registers. Refer to **PROGRAMMING** for additional information.

PROGRAMMING

3

INPUT/OUTPUT PROGRAMMING

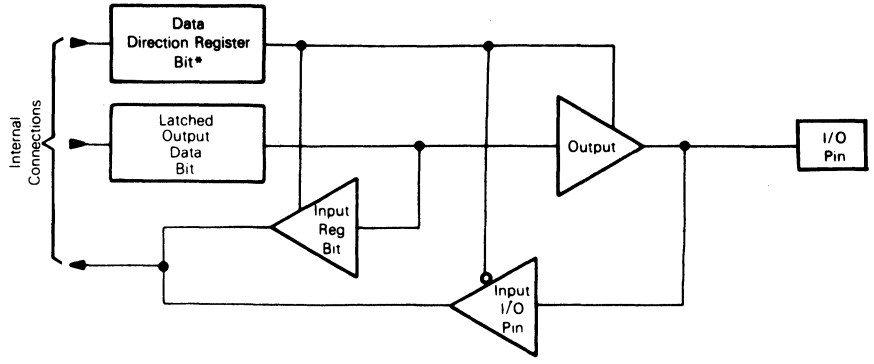
Any port pin is programmable as either input or output under software control of the corresponding write-only data direction register (DDR); DDRs always read "1". The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic 1 for output and a logic 0 for input. On reset, all the DDRs are initialized to a logic 0 state to put the ports in the input mode. The port output registers are not initialized on reset and should be written to before setting the DDR bits.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This port write may be used to initialize the data registers and avoid undefined outputs. Care must be exercised when using read-modify-write instructions since the data read corresponds to the pin level if the DDR is an input (0) and also the latched output when the DDR is an output (1). Refer to Table 1 for I/O functions and to Figure 3 for typical port circuitry.

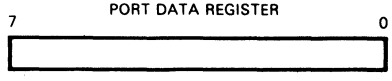
Table 1. I/O Pin Functions

Data Direction Register Bit	Output Data Bit	Output State	Input To MCU
1	0	0	0
1	1	1	1
0	X	Hi-Z**	Pin

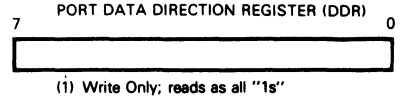
**Ports A (with CMOS drive disabled), B, and C are three state ports. Port A has optional internal pullup devices to provide CMOS drive capability. See Electrical Characteristic tables for complete information.



*DDR is a write-only register and reads as all "1s".



Port A Addr = $000
Port B Addr = $001
Port C Addr = $002 (Bits 0→3)



(1) Write Only; reads as all "1s"
(2) 1 = Output; 0 = Input. Cleared to 0 by reset.
(3) Port A Addr = $004
Port B Addr = $005
Port C Addr = $006 (Bits 0→3)

Figure 3. Typical Port I/O Circuitry and Register Configuration

MEMORY

The MCU is capable of addressing 2048 bytes of memory and I/O registers. The memory map is shown in Figure 4. The locations consist of user EPROM, bootstrap ROM, RAM, a mask option register (MOR), a program control register, and I/O. The interrupt vectors are located from $7F8 to $7FF. The bootstrap is a mask-programmed ROM that allows the MCU to program its own EPROM.

The stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer decrements during pushes and increments during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

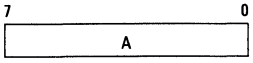
Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

REGISTERS

The MCU contains the registers described in the following paragraphs.

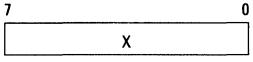
ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



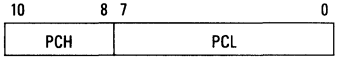
INDEX REGISTER (X)

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16-bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



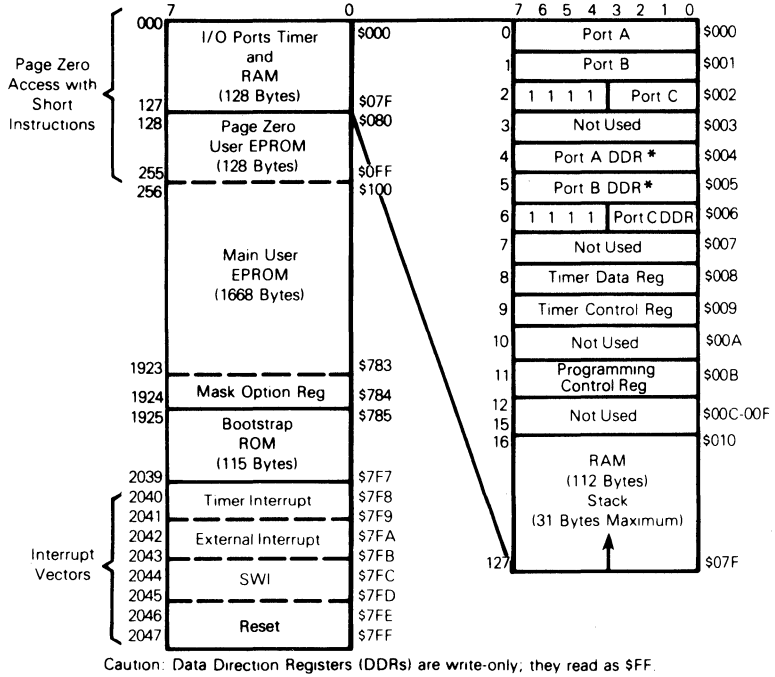
PROGRAM COUNTER (PC)

The program counter is an 11-bit register that contains the address of the next byte to be fetched.



STACK POINTER (SP)

The stack pointer is an 11-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set at location $07F. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.



Caution: Data Direction Registers (DDRs) are write-only; they read as $FF.

Figure 4. Memory Map

The six most-significant bits of the stack pointer are permanently set at 000011. Subroutines and interrupts may be nested down to location $061 (31 bytes maximum), which allows the programmer to use up to 15 levels of subroutine calls (less if interrupts are allowed).

10					5	4	0
0	0	0	0	1	1	SP	

CONDITION CODE REGISTER (CC)

The condition code register is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program, and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.

4				0
H	I	N	Z	C

3

Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, the timer and external interrupt is masked (disabled). If an external interrupt occurs while this bit is set, the interrupt is latched and is processed as soon as the interrupt bit is cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic 1).

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/Borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions, and during shifts and rotates.

RESETS

The MCU can be reset two ways: by initial power-up and by the external reset input ($\overline{\text{RESET}}$). The $\overline{\text{RESET}}$ input consists mainly of a Schmitt trigger that senses the $\overline{\text{RESET}}$ line logic level.

POWER-ON-RESET (POR)

An internal reset is generated on power-up that allows the internal clock generator to stabilize. The power-on reset is used strictly for power turn-on conditions and should not be used to detect any drop in the power supply voltage. A delay of t_{RHL} milliseconds is required before allowing the $\overline{\text{RESET}}$ input to go high. Connecting a capacitor to the $\overline{\text{RESET}}$ input (Figure 5) typically provides sufficient delay.

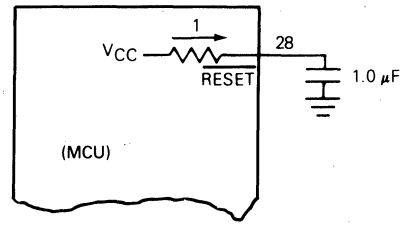


Figure 5. Power-up $\overline{\text{RESET}}$ Delay Circuit

EXTERNAL RESET INPUT

The MCU is reset when a logic zero is applied to the $\overline{\text{RESET}}$ input for a period longer than one machine cycle (t_{cyc}). Under this type of reset, the Schmitt trigger switches off at V_{IRES-} to provide an internal reset voltage.

INTERRUPTS

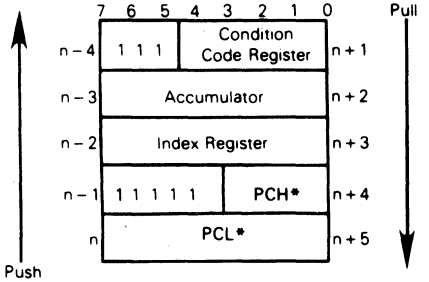
The MCU can be interrupted three different ways: (1) through the external interrupt $\overline{\text{INT}}$ input pin, (2) with the internal timer interrupt request, or (3) using the software interrupt instruction (SWI).

Interrupts cause the processor registers to be saved on the stack and the interrupt mask (I bit) set to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack and then normal processing resumes. The stacking order is shown in Figure 6.

Unlike $\overline{\text{RESET}}$, hardware interrupts do not cause the current instruction execution to be halted but are considered pending until the current instruction is complete.

NOTE

The current instruction is considered to be the one already fetched and being operated on.



*For subroutine calls, only PCH and PCL are stacked.

Figure 6. Interrupt Stacking Order

When the current instruction is complete, the processor checks all pending hardware interrupts and, if unmasked (I bit clear), proceeds with interrupt processing; otherwise, the next instruction is fetched and executed. Masked interrupts are latched for later interrupt service. If the timer interrupt status bit is cleared before unmasking the interrupt, then the interrupt is not latched.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction regardless of the setting of the I bit. Refer to Figure 7 for the reset and interrupt instruction processing sequence.

TIMER INTERRUPT

If the timer mask bit (TCR6) is cleared, then, each time the timer decrements to zero (transitions from $01 to $00), an interrupt request is generated. The actual processor interrupt is generated only if the interrupt mask bit of the condition code register (CCR) is also cleared. When the interrupt is recognized, the current state of the machine is pushed onto the stack, and the I bit in the CCR is set, masking further interrupts until the present one is serviced. The contents of the timer interrupt vector, containing the location of the timer interrupt service routine, is then loaded into the program counter. At the end of the timer interrupt service routine, the software normally executes an RTI instruction which restores the machine state and starts executing the interrupted program.

EXTERNAL INTERRUPT

The external interrupt is internally synchronized and then latched on the falling edge of $\overline{INT}$. Clearing the I bit enables the external interrupt. The following paragraphs describe two typical external interrupt circuits.

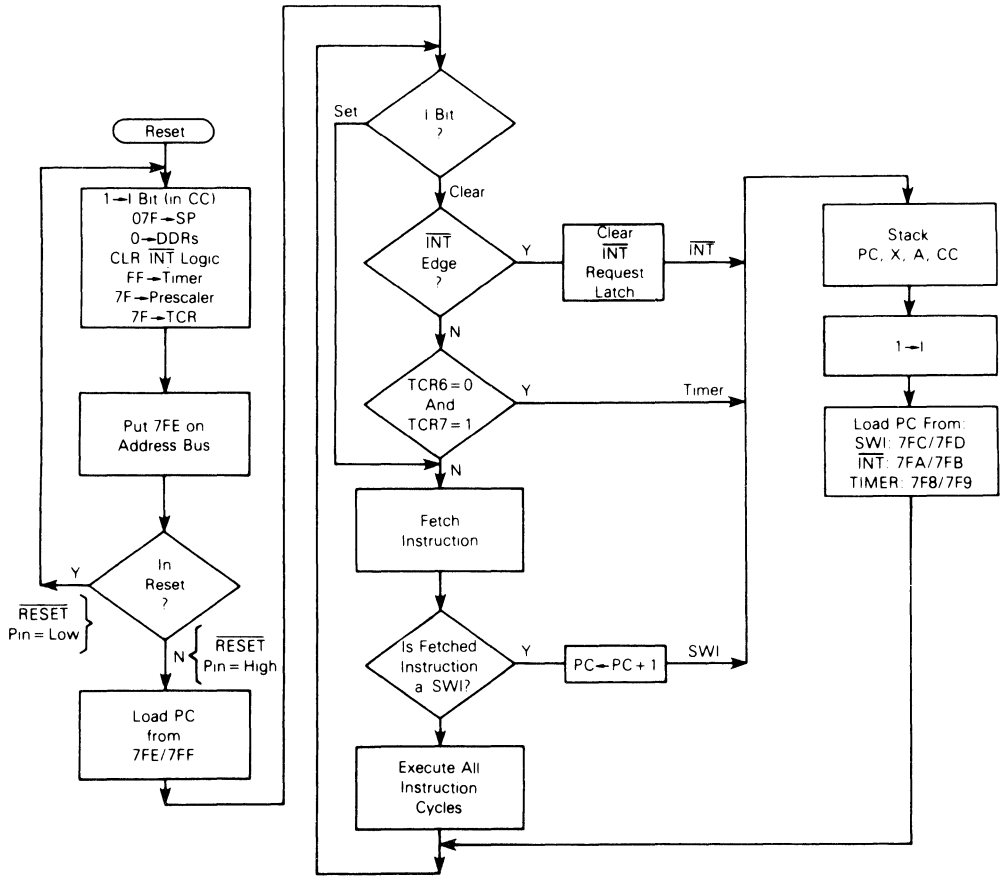


Figure 7. Reset and Interrupt Processing Flowchart

Zero-Crossing

A sinusoidal input signal (f_{INT} maximum) can be used to generate an external interrupt (see Figure 8a) for use as a zero-crossing detector (for negative transitions of the ac sinusoid). This type of circuit allows applications such as servicing time-of-day routines and engaging/disengaging ac power control devices. Off-chip, full-wave rectification provides an interrupt at every zero crossing of the ac signal and thereby provides a 2f clock.

Digital-Signal Interrupt

With this type of circuit (Fibure 8b), the $\overline{INT}$ pin can be driven by a digital signal. The maximum frequency of a signal that can be recognized by the TIMER or $\overline{INT}$ pin logic is dependent on the parameter labeled t_{WL}, t_{WH}. Refer to **TIMER** for additional information.

SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. If the I bit is zero, SWI executes after the other interrupts. The SWI execution is similar to the hardware interrupts.

TIMER

The MCU consists of an 8-bit software programmable counter driven by a 7-bit software programmable prescaler. Various timer sources are made via the timer control register (TCR). The 8-bit counter may be loaded under program control and is decremented toward zero. When the timer reaches zero, the timer interrupt request bit (bit 7) in the timer control register (TCR) is set. Refer to Figure 9 for timer block diagram.

Timer interrupt can be masked (disabled) by setting the timer interrupt mask bit (bit 6) in the TCR. When the I bit in the condition code register is cleared, and TCR bit 6 is cleared, the processor receives the interrupt. The MCU responds to this interrupt by 1) saving the present CPU state on the stack, 2) fetching the timer interrupt vector,

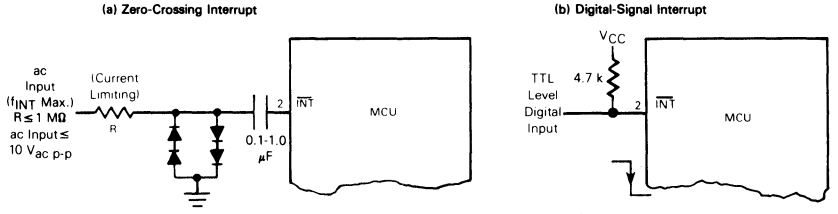


Figure 8. Typical Interrupt Circuits

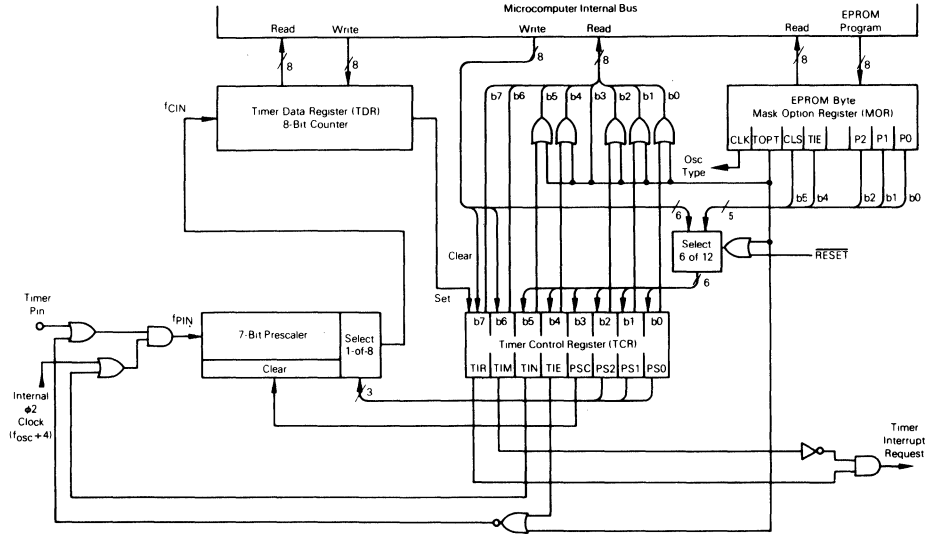


Figure 9. Timer Block Diagram

and 3) executing the interrupt routine. Timer interrupt request bit must be cleared by software. Refer to **RESETS** and **INTERRUPTS** for additional information.

The prescaler is a 7-bit divider which is used to extend the maximum length of the timer. To avoid truncation errors, the prescaler is cleared when TCR bit 3 is set to a logic 1; however, TCR bit 3 always reads as a logic 0 to ensure proper operation with read-modify-write instructions.

The timer continues to count past zero, falling from $00 through $FF, and continues the countdown. The counter can be read at any time by reading the timer data register (TDR). This allows a program to determine the length of time since a timer interrupt has occurred without disturbing the counting process. TDR is unaffected by reset.

SOFTWARE CONTROLLED MODE

The timer prescaler input can be configured for three different operating modes plus a disable mode, depending on the value written to TCR control bits 4 and 5 (TIE and TIN). The following paragraphs describe the different modes.

Timer Input Mode 1

When TIE and TIN are both programmed to zero, the timer input is from the internal clock (phase 2) and the timer input pin is disabled. The internal clock mode can be used for periodic interrupt generation as well as a reference for frequency and event measurement.

Timer Input Mode 2

When TIE = 1 and TIN = 0, the internal clock and the timer input signals are ANDed to form the timer input. This mode can be used to measure external pulse widths. The active high, external pulse gates in the internal clock for the duration of the external pulse. The accuracy of the count is ±1.

Timer Input Mode 3

When TIE = 0 and TIN = 1, no prescaler input frequency is applied to the prescaler and the timer is disabled.

Timer Input Mode 4

When TIE and TIN are both one, the timer input is from the external clock. The external clock can be used to count external events as well as to provide an external frequency for generating periodic interrupts.

MOR CONTROLLED MODE

This mode is selected when TOPT (bit 6) in the MOR is programmed to logic 1. The timer circuits are the same as described in **SOFTWARE CONTROLLED MODE**. The logic levels of TCR bits 0, 1, 2, and 5 are determined during EPROM programming by the same bits in the MOR. Therefore, bits 0, 1, 2, and 5 in the MOR control the prescaler division and the timer clock selection. TIE (bit 4) and PSC (bit 3) in the TCR are set to a logic 1 when in the MOR controlled mode. TIM (bit 6) and TIR (bit 7) are controlled by the counter and software.

TIMER CONTROL REGISTER (TCR) $009

This is an 8-bit register that controls various functions such as configuring operation mode, setting ratio of the prescaler, and generating timer interrupt request signal. All bits are read/write except bit 3. When the MOR TOPT = 1, then bits 5, 2, 1, and 0 in the TCR take on the corresponding bits of the MOR during reset.

7	6	5	4	3	2	1	0
TIR	TIM	1	1	1	1	1	1
RESET: 0	1		U	U			

TCR with MOR TOPT = 1 (MC6805P2/P6 Emulation)

7	6	5	4	3	2	1	0
TIR	TIM	TIN	TIE	PSC	PS2	PS1	PS0
RESET: 0	1						

TCR with MOR TOPT = 0 (Software Programmable Timer)

TIR — Timer Interrupt Request
Used to indicate the timer interrupt when it is logic one
1 = Set when the timer data register changes to all zeros
0 = Cleared by external reset, power-on reset, or under program control

TIM — Timer Interrupt Mask
Used to inhibit the timer interrupt.
1 = Interrupt inhibited
0 = Interrupt enabled

TIN — External or Internal
Selects input clock source
1 = External clock selected
0 = Internal clock selected ($f_{OSC}/4$)

TIE — TIMER External Enable
Used to enable external TIMER pin
1 = Enables external timer pin
0 = Disables external timer pin

PSC — Prescaler Clear
Write only bit. Writing a 1 to this bit resets the prescaler to zero. A read of this location always indicates a zero.

PS2, PS1, PS0 — Prescaler Select Bits
Decoded to select one of eight outputs of the prescaler

PS2	PS1	PS0	Prescaler Division
0	0	0	1 (Bypass Prescaler)
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

MASK OPTION REGISTER (MOR)

The MOR is implemented in EPROM and contains all zeros prior to programming. This register is not affected by reset. The MOR bits are described in the following paragraphs.

7	6	5	4	3	2	1	0
CLK	TOPT	CLS	TIE		P2	P1	P0

CLK — Clock (oscillator type)
1 = Resistor Capacitor (RC)
0 = Crystal

TOPT — Timer Option
1 = MC6805P2/P6 type timer/prescaler. All bits except 6 and 7 of the TCR are invisible to the user. Bits 5, 2, 1, and 0 of the MOR determine the equivalent MC6805P2/P6 mask options.
0 = All TCR bits are implemented as a software programmable timer. The state of MOR bits 5, 4, 2, 1, and 0 sets the initial values of their respective TCR bits.

CLS — Timer/Prescaler Clock Source
1 = External TIMER pin
0 = Internal clock

TIE — Timer External Enable
Not used if TOPT = 1. Sets the initial value of TIE in the TCR if TOPT = 0.
1 = Not used
0 = Sets initial value of TIE in the TCR

P2, P1, P0
The logical levels of these bits, when decoded, select one of eight outputs on the timer prescaler.

P2	P1	P0	Prescaler Division
0	0	0	1 (Bypass Prescaler)
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

PROGRAMMING CONTROL REGISTER (PCR)

The PCR is an 8-bit register which provides the necessary control bits to program the EPROM. The bootstrap program manipulates the PCR when programming, so the user need not be concerned with PCR in most applications.

7	6	5	4	3	2	1	0
1	1	1	1	1	$\overline{VPON}$	$\overline{PGE}$	$\overline{PLE}$

RESET:

U	U	U	U	U	U	1	1

$\overline{PLE}$ — Programming Latch Enable
Controls address and data being latched into the EPROM. Set during reset, but may be cleared anytime.
1 = Read EPROM
0 = Latch address and data on EPROM

$\overline{PGE}$ — Program Enable
Enables programming of EPROM. Must be set when changing the address and data. Set during reset.
1 = Inhibit EPROM programming
0 = Enable EPROM programming (if $\overline{PLE}$ is low)

$\overline{VPON}$ — V_{PP} On
A read-only bit that indicates high voltage at the V_{PP} pin. When set to "1", disconnects $\overline{PGE}$ and $\overline{PLE}$ from the chip.
1 = No high voltage on V_{PP} pin
0 = High voltage on V_{PP} pin

NOTE

$\overline{VPON}$ being "0" does not indicate that the V_{PP} level is correct for programming. It is used as a safety interlock for the user in the normal operating mode.

$\overline{VPON}$	$\overline{PGE}$	$\overline{PLE}$	Programming Conditions
0	0	0	Programming Mode (Program EPROM Byte)
1	0	0	PGE and PLE Disabled from System
0	1	0	Programming Disabled (Latch Address and Data in EPROM)
1	1	0	PGE and PLE Disabled from System
0	0	1	Invalid State: PGE = 0 if PLE = 0
1	0	1	Invalid State: PGE = 0 if PLE = 0
0	1	1	"High Voltage" on V_{PP}
1	1	1	PGE and PLE Disabled from System (Operating Mode)

EPROM PROGRAMMING

PROGRAMMING

The MCU bootstrap program can be used to program the MCU EPROM.

A 2764 UV EPROM must first be programmed with the same information that is to be transferred to the MCU EPROM. Refer to application note, *MC68705P3/R3/U3 8-bit EPROM Microcomputer Programming Module* (AN-857 Rev 2) for a schematic diagram and instructions on programming the MCU EPROM.

EMULATION

The MC68705P3 emulates the MC6805P2 and MC6805P6 "exactly." The MC6805P2/P6 mask features are implemented in the mask-option register (MOR) EPROM byte on the MC68705P3. A few minor exceptions to the exactness of emulation are listed below:

1. The MC68705P2/P6 "future ROM" area is implemented in the MC68705P3, and these 704 bytes must

be left unprogrammed to accurately simulate the MC6805P2/P6. The MC6805P2/P6 read all "0s" from this area.

2. The reserved ROM areas in the MC6805P2/P6 and the MC68705P3 have different data stored in them. This data is subject to change without notice. The MC6805P2/P6 use the reserved ROM for the self-check feature, and the MC68705P3 uses this area for the bootstrap program.
3. The MC6805P2/P6 read all "1s" in its 48-byte "future RAM" area. This RAM is not implemented in the MC6805P2/P6 mask ROM versions but is implemented in the MC68705P3.
4. The V_{PP} line (pin 6) in the MC68705P3 must be tied to V_{CC} for normal operation. In the MC6805P2/P6, pin 6 is the NUM pin and is grounded in normal operation.
5. The LVI feature is not available in the MC68705P3. Processing differences are not presently compatible with proper design of this feature in the EPROM version.

The operation of all other circuitry has been exactly duplicated or designed to function identically in both devices including interrupts, timer, data ports, and data direction registers (DDRs). A design goal has been to provide the user with a safe, inexpensive way to verify a program and system design before committing to a factory programmed ROM.

INSTRUCTION SET

The MCU has a set of 59 basic instructions which can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following list of instructions.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (2's Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Higher	BHI
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
(Branch if Higher or Same)	(BHS)
Branch if Carry Set	BCS
(Branch if Lower)	(BLO)

— Continued —

— Continued —

Function	Mnemonic
(Branch if Higher or Same)	(BHS)
Branch if Carry Set	BCS
(Branch if Lower)	(BLO)
Branch if Not Equal	BNE
Branch if Equal	BEQ
Branch if Half Carry Clear	BHCC
Branch if Half Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Bit is Clear	BMC
Branch if Interrupt Mask Bit is Set	BMS
Branch if Interrupt Line is Low	BIL
Branch if Interrupt Line is High	BIH
Branch to Subroutine	BSR

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No-Operation	NOP

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the first 256 bytes of the memory space, where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear and bit test, and branch functions are all implemented with a single instruction. For test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch if Bit n is Set	BRSET n (n=0 . . . 7)
Branch if Bit n is Clear	BRCLR n (n=0 . . . 7)
Set Bit n	BSET n (n=0 . . . 7)
Clear Bit n	BCLR n (n=0 . . . 7)

OPCODE MAP SUMMARY

Table 2 is an opcode map for the instructions used on the MCU.

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single byte instructions, while the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. Two byte direct addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction.

RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true.

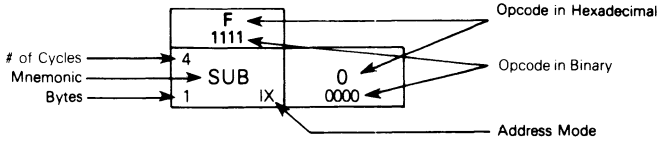
Table 2. Opcode Map

Hi / Low	Bit Manipulation BTB 0 0000	Bit Manipulation BSC 1 0001	Branch REL 2 0010	Read-Modify-Write DIR 3 0011	Read-Modify-Write INH 4 0100	Read-Modify-Write INH 5 0101	Read-Modify-Write IX1 6 0110	Read-Modify-Write IX 7 0111	Control INH 8 1000	Control INH 9 1001	Register/Memory IMM A 1010	Register/Memory DIR B 1011	Register/Memory EXT C 1100	Register/Memory IX2 D 1101	Register/Memory IX1 E 1110	Register/Memory IX F 1111	Hi / Low
0 0000	10 BRSET0 3 BTB	7 BSET0 2 BSC	4 BRA 2 REL	6 NEG 2 DIR	4 NEG 1 INH	4 NEG 1 INH	7 NEG 2 IX1	6 NEG 1 IX	9 RTI 1 INH		2 SUB 2 IMM	4 SUB 2 DIR	5 SUB 3 EXT	6 SUB 3 IX2	5 SUB 2 IX1	4 SUB 1 IX	0 0000
1 0001	10 BRCLR0 3 BTB	7 BCLR0 2 BSC	4 BRN 2 REL						6 RTS 1 INH		2 CMP 2 IMM	4 CMP 2 DIR	5 CMP 3 EXT	6 CMP 3 IX2	5 CMP 2 IX1	4 CMP 1 IX	1 0001
2 0010	10 BRSET1 3 BTB	7 BSET1 2 BSC	4 BHI 2 REL								2 SBC 2 IMM	4 SBC 2 DIR	5 SBC 3 EXT	6 SBC 3 IX2	5 SBC 2 IX1	4 SBC 1 IX	2 0010
3 0011	10 BRCLR1 3 BTB	7 BCLR1 2 BSC	4 BLS 2 REL	6 COM 2 DIR	4 COMA 1 INH	4 COMX 1 INH	7 COM 2 IX1	6 COM 1 IX	11 SWI 1 INH		2 CPX 2 IMM	4 CPX 2 DIR	5 CPX 3 EXT	6 CPX 3 IX2	5 CPX 2 IX1	4 CPX 1 IX	3 0011
4 0100	10 BRSET2 3 BTB	7 BSET2 2 BSC	4 BCC 2 REL	6 LSR 2 DTR	4 LSRA 1 INH	4 LSRX 1 INH	7 LSR 2 IX1	6 LSR 1 IX			2 AND 2 IMM	4 AND 2 DIR	5 AND 3 EXT	6 AND 3 IX2	5 AND 2 IX1	4 AND 1 IX	4 0100
5 0101	10 BRCLR2 3 BTB	7 BCLR2 2 BSC	4 BCS 2 REL								2 BIT 2 IMM	4 BIT 2 DIR	5 BIT 3 EXT	6 BIT 3 IX2	5 BIT 2 IX1	4 BIT 1 IX	5 0101
6 0110	10 BRSET3 3 BTB	7 BSET3 2 BSC	4 BNE 2 REL	6 ROR 2 DIR	4 RORA 1 INH	4 RORX 1 INH	7 ROR 2 IX1	6 ROR 1 IX			2 LDA 2 IMM	4 LDA 2 DIR	5 LDA 3 EXT	6 LDA 3 IX2	5 LDA 2 IX1	4 LDA 1 IX	6 0110
7 0111	10 BRCLR3 3 BTB	7 BCLR3 2 BSC	4 BEQ 2 REL	6 ASR 2 DIR	4 ASRA 1 INH	4 ASRX 1 INH	7 ASR 2 IX1	6 ASR 1 IX		2 TAX 1 INH		5 STA 2 DIR	6 STA 3 EXT	7 STA 3 IX2	6 STA 2 IX1	5 STA 1 IX	7 0111
8 1000	10 BRSET4 3 BTB	7 BSET4 2 BSC	4 BHCC 2 REL	6 LSL 2 DIR	4 LSLA 1 INH	4 LSLX 1 INH	7 LSL 2 IX1	6 LSL 1 IX		2 CLC 1 INH	2 EOR 2 IMM	4 EOR 2 DIR	5 EOR 3 EXT	6 EOR 3 IX2	5 EOR 2 IX1	4 EOR 1 IX	8 1000
9 1001	10 BRCLR4 3 BTB	7 BCLR4 2 BSC	4 BHCS 2 REL	6 ROL 2 DIR	4 ROLA 1 INH	4 ROLX 1 INH	7 ROL 2 IX1	6 ROL 1 IX		2 SEC 1 INH	2 ADC 2 IMM	4 ADC 2 DIR	5 ADC 3 EXT	6 ADC 3 IX2	5 ADC 2 IX1	4 ADC 1 IX	9 1001
A 1010	10 BRSET5 3 BTB	7 BSET5 2 BSC	4 BPL 2 REL	6 DEC 2 DIR	4 DECA 1 INH	4 DECX 1 INH	7 DEC 2 IX1	6 DEC 1 IX		2 CLI 1 INH	2 ORA 2 IMM	4 ORA 2 DIR	5 ORA 3 EXT	6 ORA 3 IX2	5 ORA 2 IX1	4 ORA 1 IX	A 1010
B 1011	10 BRCLR5 3 BTB	7 BCLR5 2 BSC	4 BMI 2 REL							2 SEI 1 INH	2 ADD 2 IMM	4 ADD 2 DIR	5 ADD 3 EXT	6 ADD 3 IX2	5 ADD 2 IX1	4 ADD 1 IX	B 1011
C 1100	10 BRSET6 3 BTB	7 BSET6 2 BSC	4 BMC 2 REL	6 INC 2 DIR	4 INCA 1 INH	4 INCX 1 INH	7 INC 2 IX1	6 INC 1 IX		2 RSP 1 INH		3 JMP 2 DIR	4 JMP 3 EXT	5 JMP 3 IX2	4 JMP 2 IX1	3 JMP 1 IX	C 1100
D 1101	10 BRCLR6 3 BTB	7 BCLR6 2 BSC	4 BMS 2 REL	6 TST 2 DIR	4 TSTA 1 INH	4 TSTX 1 INH	7 TST 2 IX1	6 TST 1 IX		2 NOP 1 INH	8 BSR 2 REL	7 JSR 2 DIR	8 JSR 3 EXT	9 JSR 3 IX2	8 JSR 2 IX1	7 JSR 1 IX	D 1101
E 1110	10 BRSET7 3 BTB	7 BSET7 2 BSC	4 BIL 2 REL								2 LDX 2 IMM	4 LDX 2 DIR	5 LDX 3 EXT	6 LDX 3 IX2	5 LDX 2 IX1	4 LDX 1 IX	E 1110
F 1111	10 BRCLR7 3 BTB	7 BCLR7 2 BSC	4 BIH 2 REL	6 CLR 2 DIR	4 CLRA 1 INH	4 CLRX 1 INH	7 CLR 2 IX1	6 CLR 1 IX		2 TXA 1 INH		5 STX 2 DIR	6 STX 3 EXT	7 STX 3 IX2	6 STX 2 IX1	5 STX 1 IX	F 1111

Abbreviations for Address Modes

INH	Inherent
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



Otherwise, control proceeds to the next instruction. The span of relative addressing is from −126 to +129 from the opcode address.

INDEX, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. This addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this 2-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This addressing mode can be used in a manner similar to indexed, 8-bit offset except that this 3-byte instruction allows tables to be anywhere in memory.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte to which the specified bit is to be set or cleared. Any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single 2-byte instruction.

CAUTION

The corresponding DDRs for ports A, B, and C are write only registers (registers at $004, $005, and $006). A read operation on these registers always returns "1". Since BSET and BCLR are read-modify-write functions, they cannot be used to set or clear a DDR bit (all "unaffected" bits would be set). It is recommended that all DDR bits in a port be written using a single-store instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit that is to be tested and its condition (set or clear) is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single 3-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltages			
EPROM Programming Voltage (V_{PP} Pin)	V_{PP}	−0.3 to +22.0	V
	V_{in}	−0.3 to +7.0	V
TIMER Pin (Normal Mode)			
TIMER Pin (Bootstrap Programming Mode)	V_{in}	−0.3 to +15.0	V
	V_{in}	−0.3 to +7.0	V
All Others			
Operating Temperature Range	T_A	T_L to T_H 0 to +70	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature Cerdip	T_J	150	°C/W

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation, V_{in} and V_{out} should be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{CC}$. Reliability of operation is enhanced if unused inputs except EXTAL are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Cerdip	θ_{JA}	60	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
$P_D = P_{INT} + P_{I/O}$
$P_{INT} = I_{CC} \times V_{CC}$, Watts – Chip Internal Power
$P_{I/O}$ = Power Dissipation on Input and Output Pins – User Determined

For most applications $P_{I/O} < P_{INT}$ and can be neglected.

The following is an approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected):

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

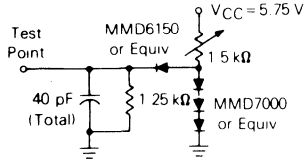


Figure 10. TTL Equivalent Test Load (Port B)

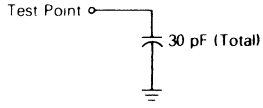


Figure 11. CMOS Equivalent Test Load (Port A)

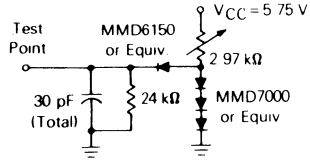


Figure 12. TTL Equivalent Test Load (Ports A and C)

ELECTRICAL CHARACTERISTICS (V_{CC} = 5.25 ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage	V_{IH}				V
$\overline{RESET}$ (4.75 ≤ V_{CC} ≤ 5.75)		4.0	—	V_{CC}	
(V_{CC} < 4.75)		V_{CC} −0.5	—	V_{CC}	
$\overline{INT}$ (4.75 ≤ V_{CC} ≤ 5.75)		4.0	**	V_{CC}	
(V_{CC} < 4.75)		V_{CC} −0.5	**	V_{CC}	
All Other		2.0	—	V_{CC}	
Input High Voltage (TIMER Pin)	V_{IH}				V
Timer Mode		2.0	—	V_{CC}	
Bootstrap Programming Mode		9.0	12.0	15.0	
Input Low Voltage	V_{IL}				V
$\overline{RESET}$		−0.3	—	0.8	
$\overline{INT}$		−0.3	**	1.5	
All Other		−0.3	—	0.8	
Internal Power Dissipation (No Port Loading, V_{CC} = 5.25 V, T_A = 0°C)	P_{INT}	—	450	TBD	mW
Input Capacitance	C_{in}				pF
XTAL		—	25	—	
All Other		—	10	—	
$\overline{INT}$ Zero-Crossing Voltage, through a Capacitor	V_{INT}	2.0	—	4.0	V_{acp-p}
$\overline{RESET}$ Hysteresis Voltage					V
Out of Reset Voltage	V_{IRES+}	2.1	—	4.0	
Into Reset Voltage	V_{IRES-}	0.8	—	2.0	
Programming Voltage (V_{PP} Pin)	V_{PP}*				V
Programming EPROM		20.0	21.0	22.0	
Operating Mode		4.0	V_{CC}	5.75	
Input Current	I_{in}				μA
TIMER (V_{in} = 0.4 V)		—	—	20	
$\overline{INT}$ (V_{in} = 0.4 V)		—	20	50	
EXTAL (V_{in} = 2.4 V to V_{CC} Crystal Option)		—	—	10	
(V_{in} = 0.4 V Crystal Option)		—	—	−1600	
$\overline{RESET}$ (V_{in} = 0.8 V)		−4.0	—	−40	
(External Capacitor Changing Current)					

*V_{PP} is pin 6 on the MC68705P3 and is connected to V_{CC} in the normal operating mode. In the MC6805P2, pin 6 is NUM and is connected to V_{SS} in the normal operating mode. The user must allow for this difference when emulating the MC6805P2 ROM-based MCU.

**Due to internal biasing, this input (when not used) floats to approximately 2.0 V.

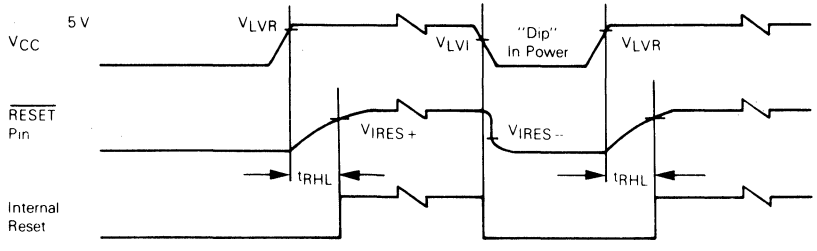


Figure 13. Power and Reset Timing

PORT DC ELECTRICAL CHARACTERISTICS (V_{CC} = +5.25 ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0° to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Port A					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Output High Voltage, I_{Load} = −10 μA	V_{OH}	V_{CC} −10	—	—	V
Input High Voltage, I_{Load} = −300 μA (Max)	V_{IH}	2.0	—	V_{CC} +0.7	V
Input Low Voltage, I_{Load} = −500 μA (Max)	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current (V_{in} = 2.0 V to V_{CC})	I_{IH}	—	—	−300	μA
Hi-Z State Input Current (V_{in} = 0.4 V)	I_{IL}	—	—	−500	μA
Port B					
Output Low Voltage, I_{Load} = 3.2 mA	(V_{OL}	—	—	0.4	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.0	V
Output High Voltage, I_{Load} = −200 μA	V_{OH}	2.4	—	—	V
Darlington Current Drive (Source), V_O = 1.5 V	I_{OH}	−1.0	—	−10	mA
Input High Voltage	V_{IH}	2.0	—	V_{CC} +0.7	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	2	20	μA
Port C					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC} +0.7	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	2	20	μA

SWITCHING CHARACTERISTICS (V_{CC} = +5.25 ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0° to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Oscillator Frequency Normal	f_{osc}	 0.4	 —	 4.2	MHz
Instruction Cycle Time ($4/f_{osc}$)	t_{cyc}	0.950	—	10	μs
$\overline{INT}$ or Timer Pulse Width (See Interrupt Section)	t_{WL}, t_{WH}	t_{cyc} + 250	—	—	ns
$\overline{RESET}$ Pulse Width	t_{RWL}	t_{cyc} + 250	—	—	ns
$\overline{RESET}$ Delay Time (External Cap = 1.0 μF)	t_{RHL}	100	—	—	ms
$\overline{INT}$ Zero Crossing Detection Input Frequency	f_{INT}	0.03	—	1.0	kHz
External Clock Duty Cycle (EXTAL)	—	40	50	60	%

PROGRAMMING OPERATION ELECTRICAL CHARACTERISTICS (V_{CC} = +5.25 ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 20° to 30°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Programming Voltage (V_{PP} Pin)	V_{PP}	20.0	21.0	22.0	V
V_{PP} Supply Current V_{PP} = 5.25 V V_{PP} = 21.0 V	I_{PP}	 — —	 — —	 8 30	mA
Programming Oscillator Frequency	f_{oscp}	0.9	1.0	1.1	MHz
Bootstrap Programming Mode Voltage (TIMER Pin) I_{in} = 100 μA Max	V_{IHTP}	9.0	12.0	15.0	V

3

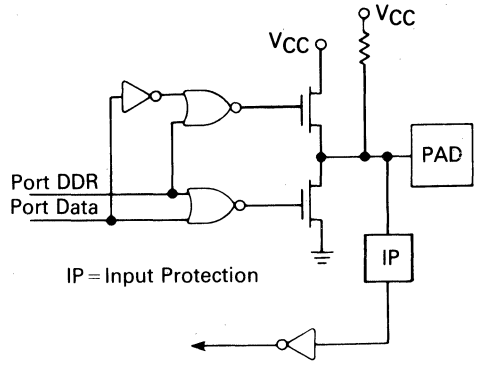


Figure 14. Port A Logic Diagram

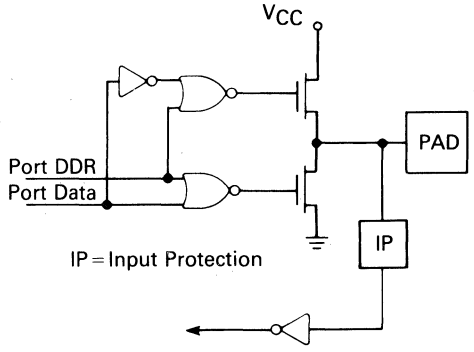


Figure 15. Port B and Port C Logic Diagram

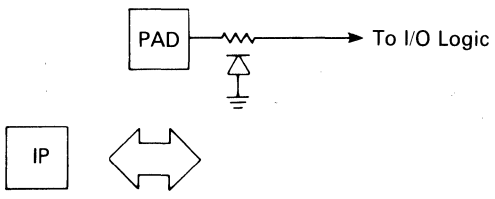


Figure 16. Typical Input Protection

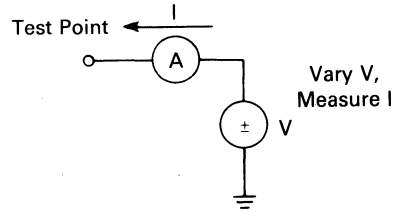


Figure 17. I/O Characteristic Measurement Circuit

ORDERING INFORMATION

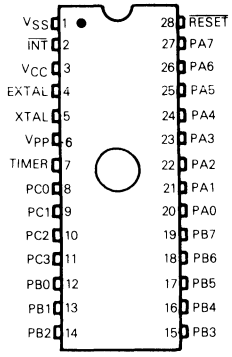
The following table provides generic information pertaining to the package type, temperature, and MC order numbers for the MC68705P3.

Table 3. Generic Information

Package Type	Internal Clock Frequency (MHz)	Temperature	Order Number
Cerdip (S Suffix)	1.0	0° to 70°C	MC68705P3S
Cerdip (S Suffix)	1.0	−40° to +85°C	MC68705P3CS

MECHANICAL DATA

PIN ASSIGNMENTS



3

MC68705P5

Technical Summary
8-Bit EPROM Microcomputer Unit

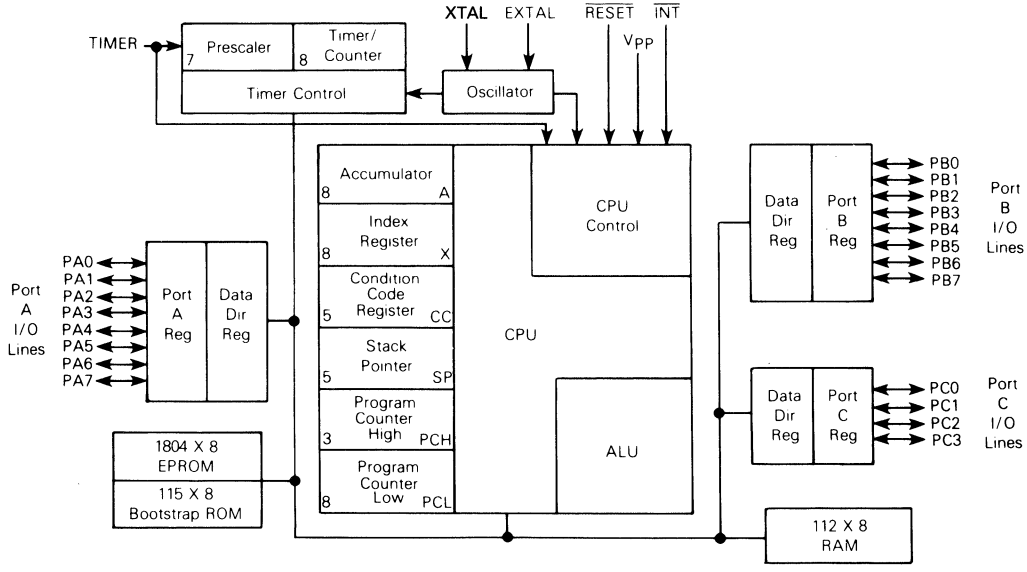
The MC68705P5 (High-Density NMOS) Microcomputer Unit (MCU) is an EPROM member of the MC6805 Family of microcomputers. The user programmable EPROM allows program changes and lower volume applications. This low cost MCU has parallel I/O capability with pins programmable as input or output. This publication contains condensed information on the MCU; for detailed information, refer to *M6805 HMOS, M146805 CMOS Family User's Manual* (M6805UM(AD2)) or contact your local Motorola sales office.

Refer to the block diagram for the hardware features and to the list below for additional features available on the MCU.

- Internal 8-Bit Timer with 7-Bit Programmable Prescaler
- On-chip Oscillator
- 112 Bytes RAM
- 1804 Bytes EPROM
- 20 TTL/CMOS Compatible Bidirectional I/O Lines
- EPROM Security Features (Hardware and Software)
- Memory Mapped I/O
- Versatile Interrupt Handling
- Bit Manipulation
- Bit Test and Branch Instruction
- Vectored Interrupts

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

V_{CC} AND V_{SS}

Power is supplied to the microcomputer using these two pins. V_{CC} is +5.25 volts (±0.5Δ) power, and V_{SS} is ground.

V_{PP}

This pin is used when programming the EPROM. In normal operation, this pin is connected to V_{CC}.

$\overline{INT}$

This pin provides the capability for asynchronously applying an external interrupt to the MCU. Refer to **INTERRUPTS** for more detail information.

EXTAL, XTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a resistor/capacitor combination, or an external signal is connected to these pins to provide a system clock. Selection is made by the CLK bit in the mask option register.

RC Oscillator

With this option, a resistor is connected to the oscillator pins as shown in Figure 1. The relationship between R and f_{osc} is shown in Figure 2.

Crystal

The circuit shown in Figure 1 is recommended when using a crystal. Using an external CMOS oscillator is recommended when crystals outside the specified ranges are to be used. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and startup statilization time.

External Clock

An external clock should be applied to the EXTAL input with the XTAL input connected to ground, as shown in Figure 1. This option may only be used with the crystal oscillator option selected in the mask option register.

TIMER

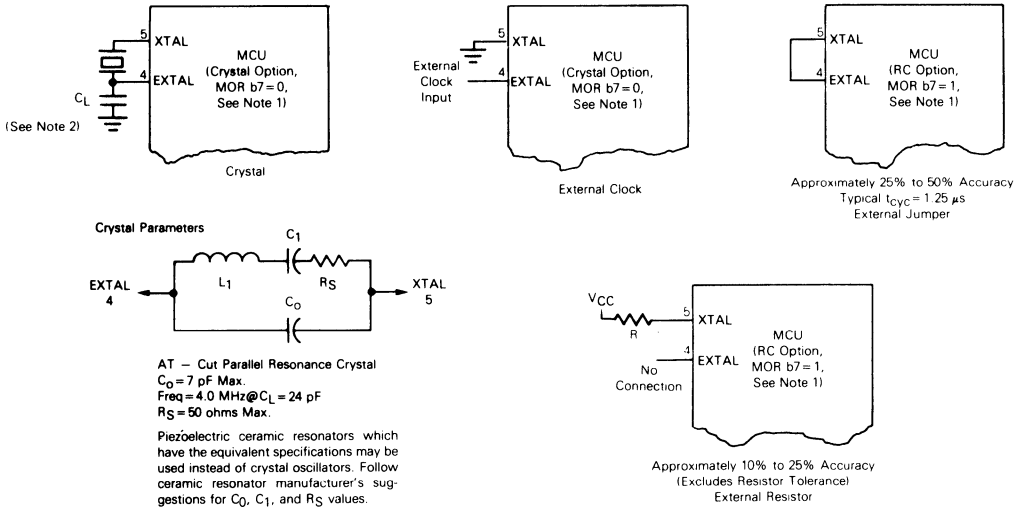
This pin is used as an external input to control the internal timer/counter circuitry. This pin also detects a higher voltage level used to initiate the bootstrap program.

$\overline{RESET}$

This pin has a Schmitt trigger input and an on-chip pullup. The MCU can be reset by pulling $\overline{RESET}$ low. Refer to **RESETS** for addition information.

INPUT/OUTPUT LINES (PA0-PA7, PB0-PB7, PC0-PC3)

These 20 lines are arranged into two 8-bit ports (A and B) and one 4-bit port (C). All lines are programmable as



NOTES:

1. When the TIMER input pin is in the V_{IHTP} range (in the bootstrap EPROM programming mode), the crystal option is forced. When the TIMER input is at or below V_{CC}, the clock generator option is determined by bit 7 of the mask option register (CLK).
2. The recommended C_L value with a 4.0 MHz crystal is 27 pF maximum including system distributed capacitance. There is an internal capacitance of approximately 25 pF on the XTAL pin. For crystal frequencies other than 4 MHz, the total capacitance on each pin should be scaled as the inverse of the frequence ratio. For example, with a 2 MHz crystal, use approximately 50 pF on EXTAL and approximately 25 pF on XTAL. The exact value depends on the motional-arm parameters of the crystal used.

Figure 1. Oscillator Connections

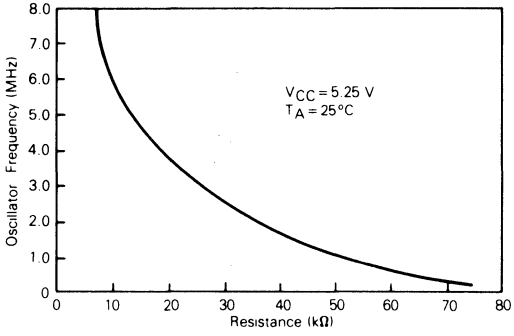


Figure 2. Typical Frequency vs Resistance for RC Oscillator Option Only

either inputs or outputs under software control of the data direction registers. Refer to **PROGRAMMING** for additional information.

3

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

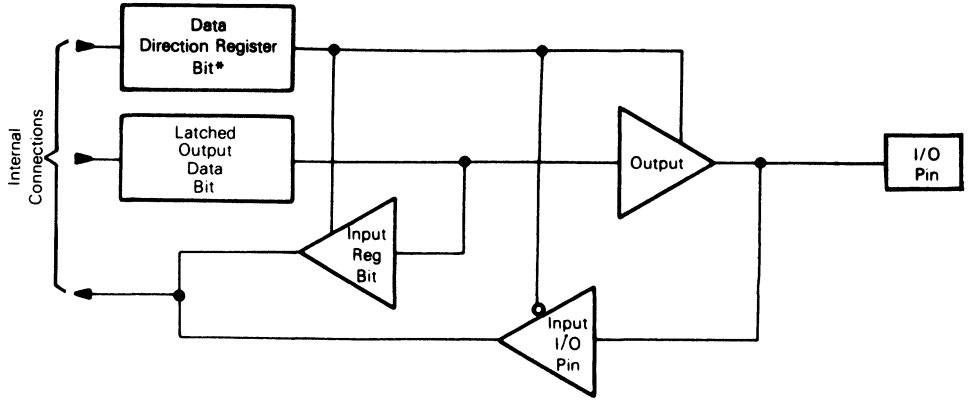
Any port pin is programmable as either input or output under software control of the corresponding write-only data direction register (DDR). DDRs always read "one". The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic one for output and a logic zero for input. On reset, all the DDRs are initialized to a logic zero state to put the ports in the input mode. The port output registers are not initialized on reset and should be written to before setting the DDR bits.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This port write may be used to initialize the data registers and avoid undefined outputs. Care must be exercised when using read-modify-write instructions since the data read corresponds to the pin level if the DDR is an input (zero) and also to the latched output when the DDR is an output (one). Refer to Table 1 for I/O functions and to Figure 3 for typical port circuitry.

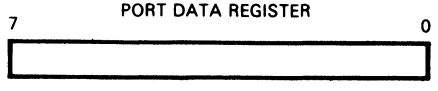
Table 1. I/O Pin Functions

R/$\overline{W}$*	DDR	I/O Pin Functions
0	0	The I/O pin is in input mode. Data is written into the output data latch.
0	1	Data is written into the output data latch and output to the I/O pin.
1	0	The state of the I/O pin is read.
1	1	The I/O pin is in an output mode. The output data latch is read.

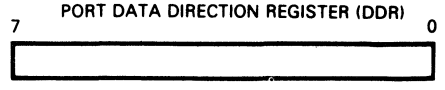
*R/$\overline{W}$ is an internal signal.



*DDR is a write-only register and reads as all "1s".



Port A Addr = $000
Port B Addr = $001
Port C Addr = $002 (Bits 0→3)



(1) Write Only; reads as all "1s"
(2) 1 = Output; 0 = Input. Cleared to 0 by reset.
(3) Port A Addr = $004
Port B Addr = $005
Port C Addr = $006 (Bits 0→3)

Figure 3. Typical Port I/O Circuitry and Register Configuration

MEMORY

The MCU is capable of addressing 2048 bytes of memory and I/O registers. The memory map is shown in Figure 4. The locations consist of user EPROM, bootstrap ROM, RAM, a mask option register (MOR), a program control register, and I/O. The interrupt vectors are located from $7F8 to $7FF. The bootstrap is a mask-programmed ROM that allows the MCU to program its own EPROM.

The stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer decrements during pushes and increments during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

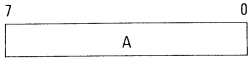
Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

REGISTERS

The MCU contains the registers described in the following paragraphs.

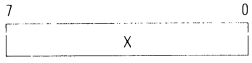
ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



INDEX REGISTER (X)

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16-bit immediate value to create an effective address. The index register may also be used as a temporary storage area.

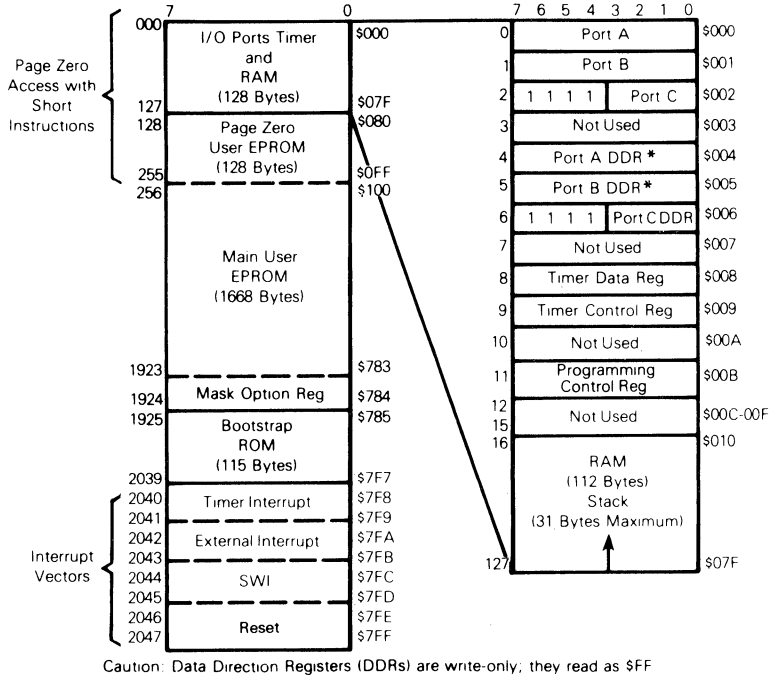


PROGRAM COUNTER (PC)

The program counter is an 11-bit register that contains the address of the next byte to be fetched.

STACK POINTER (SP)

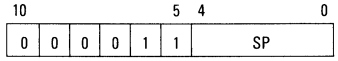
The stack pointer is an 11-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set at location $07F. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.



Caution: Data Direction Registers (DDRs) are write-only; they read as $FF.

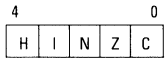
Figure 4. Memory Map

The six most-significant bits of the stack pointer are permanently set at 000011. Subroutines and interrupts may be nested down to location $061 (31 bytes maximum), which allows the programmer to use up to 15 levels of subroutine calls (less if interrupts are allowed).



CONDITION CODE REGISTER (CC)

The condition code register is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program, and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.



Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, the timer and external interrupt is masked (disabled). If an external interrupt occurs while this bit is set, the interrupt is latched and is processed as soon as the interrupt bit is cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic one).

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/Borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions, and during shifts and rotates.

RESETS

The MCU can be reset two ways: by initial power-up and by the external reset input ($\overline{\text{RESET}}$). The $\overline{\text{RESET}}$ input consists mainly of a Schmitt trigger that senses the $\overline{\text{RESET}}$ line logic level.

POWER-ON-RESET (POR)

An internal reset is generated on power-up that allows the internal clock generator to stabilize. The power-on reset is used strictly for power turn-on conditions and should not be used to detect any drop in the power supply voltage. A delay of t_{RHL} milliseconds is required before allowing the $\overline{\text{RESET}}$ input to go high. Connecting a capacitor to the $\overline{\text{RESET}}$ input (Figure 5) typically provides sufficient delay.

EXTERNAL RESET INPUT

The MCU is reset when a logic zero is applied to the $\overline{\text{RESET}}$ input for a period longer than one machine cycle (t_{cyc}). Under this type of reset, the Schmitt trigger switches off at V_{IRES-} to provide an internal reset voltage.

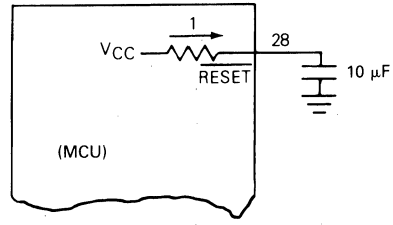


Figure 5. Power-up $\overline{\text{RESET}}$ Delay Circuit

INTERRUPTS

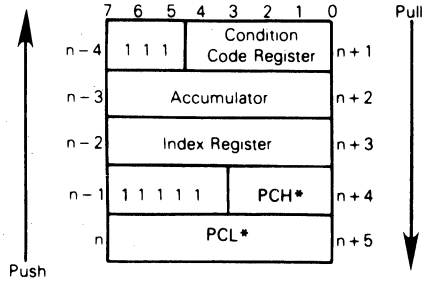
The MCU can be interrupted three different ways: (1) through the external interrupt $\overline{\text{INT}}$ input pin, (2) with the internal timer interrupt request, or (3) using the software interrupt instruction (SWI).

Interrupts cause the processor registers to be saved on the stack and the interrupt mask (I bit) set to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack after which-normal processing resumes. The stacking order is shown in Figure 6.

Unlike $\overline{\text{RESET}}$, hardware interrupts do not cause the current instruction execution to be halted but are considered pending until the current instruction is complete.

NOTE

The current instruction is considered to be the one already fetched and being operated on.



*For subroutine calls, only PCH and PCL are stacked.

Figure 6. Interrupt Stacking Order

When the current instruction is complete, the processor checks all pending hardware interrupts and, if unmasked, (I bit clear) proceeds with interrupt processing; otherwise, the next instruction is fetched and executed. Masked interrupts are latched for later interrupt service. If the timer interrupt status bit is cleared before unmasking the interrupt, then the interrupt is not latched.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction regardless of the setting of the I bit. Refer to Figure 7 for the reset and interrupt instruction processing sequence.

TIMER INTERRUPT

If the timer mask bit (TCR6) is cleared, then, each time the timer decrements to zero (transitions from $01 to $00), an interrupt request is generated. The actual processor interrupt is generated only if the interrupt mask bit of the condition code register (CCR) is also cleared. When the interrupt is recognized, the current state of the machine is pushed onto the stack and the I bit in the CCR is set, masking further interrupts until the present one is serviced. The contents of the timer interrupt vector, containing the location of the timer interrupt service routine, is then loaded into the program counter. At the end of the timer interrupt service routine, the software normally executes an RTI instruction which restores the machine state and starts executing the interrupted program.

EXTERNAL INTERRUPT

The external interrupt is internally synchronized and then latched on the falling edge of $\overline{INT}$. Clearing the I bit enables the external interrupt. The following paragraphs describes two typical external interrupt circuits.

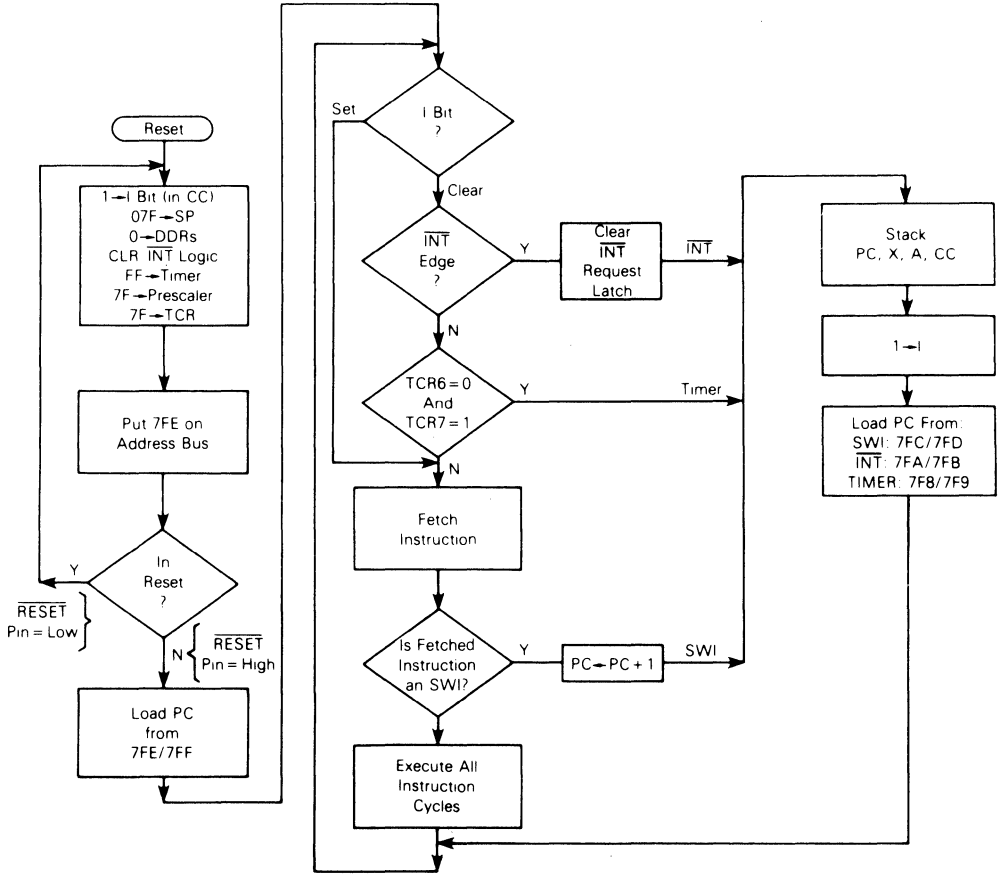


Figure 7. Reset and Interrupt Processing Flowchart

Zero-Crossing

A sinusoidal input signal (f_{INT} maximum) can be used to generate an external interrupt (See Figure 8a) for use as a zero-crossing detector (for negative transitions of the ac sinusoid). This type of circuit allows applications such as servicing time-of-day routines and engaging/disengaging ac power control devices. Off-chip, full-wave rectification provides an interrupt at every zero crossing of the ac signal and thereby provides a 2f clock.

Digital-Signal Interrupt

With this type of circuit (Figure 8b), the $\overline{INT}$ pin can be driven by a digital signal. The maximum frequency of a signal that can be recognized by the TIMER or $\overline{INT}$ pin logic is dependent on the parameter labeled t_{WL}, t_{WH}. Refer to **TIMER** for additional information.

SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. If the I bit is zero, SWI executes after the other interrupts. The SWI execution is similar to the hardware interrupts.

TIMER

The MCU consists of an 8-bit software programmable counter driven by a 7-bit software programmable prescaler. The various timer sources are made via the timer control register (TCR). The 8-bit counter may be loaded under program control and is decremented toward zero. When the timer reaches zero, the timer interrupt request bit (bit 7) in the timer control register (TCR) is set. Refer to Figure 9 for timer block diagram.

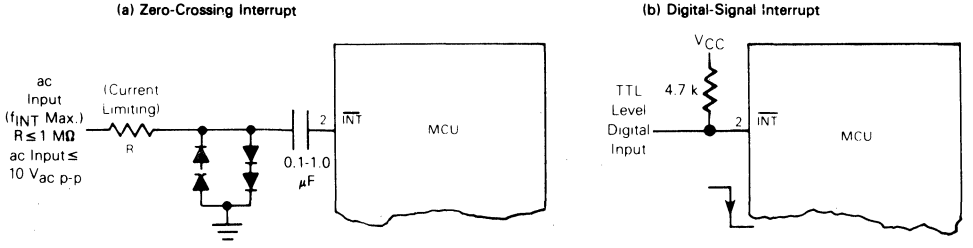


Figure 8. Typical Interrupt Circuits

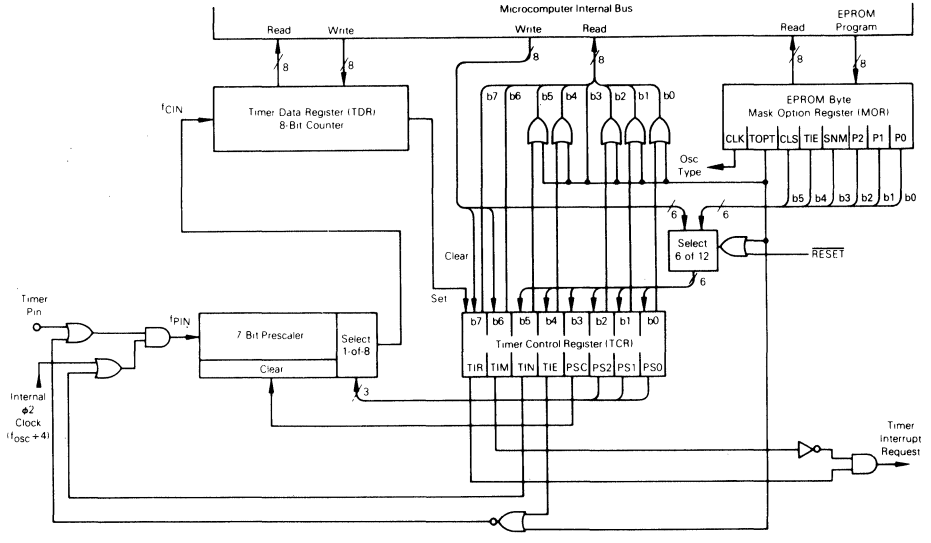


Figure 9. Timer Block Diagram

The timer interrupt can be masked (disabled) by setting the timer interrupt mask bit (bit 6) in the TCR. When the I bit in the condition code register is cleared and TCR bit 6 is cleared, the processor receives the interrupt. The MCU responds to this interrupt by 1) saving the present CPU state on the stack, 2) fetching the timer interrupt vector, and 3) executing the interrupt routine. The timer interrupt request bit must be cleared by software. Refer to **RESETS** and **INTERRUPTS** for additional information.

The prescaler is a 7-bit divider which is used to extend the maximum length of the timer. To avoid truncation errors, the prescaler is cleared when TCR bit 3 is set to a logic one; however, TCR bit 3 always reads as a logic zero to ensure proper operation with read-modify-write instructions.

The timer continues to count past zero, falling from $00 through $FF, and continues the countdown. The counter can be read at any time by reading the timer data register (TDR). This allows a program to determine the length of time since a timer interrupt has occurred without disturbing the counting process. TDR is unaffected by reset.

SOFTWARE CONTROLLED MODE

The timer prescaler input can be configured for three different operating modes plus a disable mode, depending on the value written to TCR control bits 4 and 5 (TIE and TIN). The following paragraphs describe the different modes.

Timer Input Mode 1

When TIE and TIN are both programmed to zero, the timer input is from the internal clock (phase two) and the timer input pin is disabled. The internal clock mode can be used for periodic interrupt generation as well as a reference for frequency and event measurement.

Timer Input Mode 2

When TIE = 1 and TIN = 0, the internal clock and the timer input signals are ANDed to form the timer input. This mode can be used to measure external pulse widths. The active high, external pulse gates in the internal clock for the duration of the external pulse. The accuracy of the count is ±1.

Timer Input Mode 3

When TIE = 0 and TIN = 1, no prescaler input frequency is applied to the prescaler and the timer is disabled.

Timer Input Mode 4

When TIE and TIN are both one, the timer input is from the external clock. The external clock can be used to count external events as well as to provide an external frequency for generating periodic interrupts.

MOR CONTROLLED MODE

This mode is selected when TOPT (bit 6) in the MOR is programmed to logic one. The timer circuits are the same as described in **SOFTWARE CONTROLLED MODE.** The logic levels of TCR bits 0, 1, 2, and 5 are determined during EPROM programming by the same bits in the MOR. Therefore, bits 0, 1, 2, and 5 in the MOR control the prescaler division and the timer clock selection. TIE (bit 4) and PSC (bit 3) in the TCR are set to a logic one when in the MOR controlled mode. TIM (bit 6) and TIR (bit 7) are controlled by the counter and software.

TIMER CONTROL REGISTER (TCR) $009

This is an 8-bit register that controls various functions such as configuring operation mode, setting ratio of the prescaler, and generating timer interrupt request signal. All bits are read/write except bit 3. When the MOR TOPT = 1, then bits 5, 2, 1, and 0 in the TCR take on the corresponding bits of the MOR during reset.

7	6	5	4	3	2	1	0
TIR	TIM	1	1	1	1	1	1
RESET: 0	1		U	U			

TCR with MOR TOPT = 1 (MC6805P2/P6 Emulation)

7	6	5	4	3	2	1	0
TIR	TIM	TIN	TIE	PSC	PS2	PS1	PS0
RESET: 0	1						

TCR with MOR TOPT = 0 (Software Programmable Timer)

TIR — Timer Interrupt Request
Used to indicate the timer interrupt when it is logic one
1 = Set when the timer data register changes to all zeros
0 = Cleared by external reset, power-on reset, or under program control

TIM — Timer Interrupt Mask
Used to inhibit the timer interrupt
1 = Interrupt inhibited
0 = Interrupt enabled

TIN — External or Internal
Selects input clock source
1 = External clock selected
0 = Internal clock selected ($f_{OSC}/4$)

TIE — TIMER External Enable
Used to enable external TIMER pin
1 = Enables external timer pin
0 = Disables external timer pin

PSC — Prescaler Clear
Write only bit. Writing a 1 to this bit resets the prescaler to zero. A read of this location always indicates a zero.

PS2, PS1, PS0 — Prescaler Select Bits
Decoded to select one of eight outputs of the prescaler

PS2	PS1	PS0	Prescaler Division
0	0	0	1 (Bypass Prescaler)
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

MASK OPTION REGISTER (MOR)

The MOR is implemented in EPROM and contains all zeros prior to programming. This register is not affected by reset. The MOR bits are described in the following paragraphs.

7	6	5	4	3	2	1	0
CLK	TOPT	CLS	TIE	SNM	P2	P1	P0

CLK — Clock (oscillator type)
1 = Resistor capacitor (RC)
0 = Crystal

TOPT — Timer Option
1 = MC6805P2/P6 type timer/prescaler. All bits, except 6 and 7, of the TCR are invisible to the user. Bits 5, 2, 1, and 0 of the MOR determine the equivalent MC6805P2/P6 mask options.
0 = All TCR bits are implemented as a software programmable timer. The state of MOR bits 5, 4, 2, 1, and 0 sets the initial values of their respective TCR bits.

CLS — Timer/Prescaler Clock Source
1 = External TIMER pin
0 = Internal clock

TIE — Timer External Enable
Not used if TOPT = 1. Sets the initial value of TIE in the TCR if TOPT = 0
1 = Not used
0 = Sets initial value of TIE in the TCR

SNM — Secure Mode
When programmed to one, EPROM contents cannot be access externally.

P2, P1, P0
The logical levels of these bits, when decoded, select one of eight outputs on the timer prescaler.

P2	P1	P0	Prescaler Division
0	0	0	1 (Bypass Prescaler)
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

PROGRAMMING CONTROL REGISTER (PCR)

The PCR is an 8-bit register which provides the necessary control bits to program the EPROM. The bootstrap program manipulates the PCR when programming, so the user need not be concerned with PCR in most applications.

7	6	5	4	3	2	1	0
1	1	1	1	1	$\overline{VPON}$	$\overline{PGE}$	$\overline{PLE}$
RESET:							
U	U	U	U	U	U	1	1

$\overline{PLE}$ — Programming Latch Enable
Controls address and data being latched into the EPROM. Set during reset, but may be cleared anytime
1 = Read EPROM
0 = Latch address and data on EPROM

$\overline{PGE}$ — Program Enable
Enables programming of EPROM. Must be set when changing the address and data. Set during reset
1 = Inhibit EPROM programming
0 = Enable EPROM programming (if $\overline{PLE}$ is low)

$\overline{VPON}$ — V_{PP} On
A read-only bit that indicates high voltage at the V_{PP} pin. When set to "one", disconnects $\overline{PGE}$ and $\overline{PLE}$ from the chip.
1 = No high voltage on V_{PP} pin
0 = High voltage on V_{PP}

NOTE

$\overline{VPON}$, being "zero", does not indicate that the V_{PP} level is correct for programming. It is used as a safety interlock for the user in the normal operating mode.

$\overline{VPON}$	$\overline{PGE}$	$\overline{PLE}$	Programming Conditions
0	0	0	Programming Mode (Program EPROM Byte)
1	0	0	$\overline{PGE}$ and $\overline{PLE}$ Disabled from System
0	1	0	Programming Disabled (Latch Address and Data in EPROM)
1	1	0	$\overline{PGE}$ and $\overline{PLE}$ Disabled from System
0	0	1	Invalid State $\overline{PGE}$ = 0 if $\overline{PLE}$ = 0
1	0	1	Invalid State $\overline{PGE}$ = 0 if $\overline{PLE}$ = 0
0	1	1	"High Voltage" on V_{PP}
1	1	1	$\overline{PGE}$ and $\overline{PLE}$ Disabled from System (Operating Mode)

PROGRAMMING

The MCU bootstrap program can be used to program the MCU EPROM.

A MCM2716 UV EPROM (other industry standard EPROMs may be used) must first be programmed with the same information that is to be transferred to the MCU EPROM.

The MC68705P5 MCU is programmed in the same as the MC68705P3. Refer to application note, *MC68705P3/R3/U3 8-Bit EPROM Microcomputer Programming Module* (AN-856 Rev2) for schematic diagrams and instructions on programming the MC68705P5 MCU EPROM.

NOTE

The MC68705P5 will not execute the bootstrap program when in the secure mode. The on-chip EPROM must be completely erased before programming. To enter the secure mode, bit 3 in the mask option register must be programmed to logic "one" and memory locations $782 and $783 must be programmed with $20 and $FE, respectively. After programming, the only way to change the non-secure mode is by erasing the entire EPROM.

EMULATION

The MC68705P5 emulates the MC6805P2 and MC6805P6 "exactly". The MC6805P2/P6 mask features are implemented in the mask option register (MOR) EPROM byte on the MC68705P5. A few minor exceptions to the exactness of emulation are listed below:

1. The MC68705P2/P6 "future ROM" area is implemented in the MC68705P3 and these 704 bytes must be left unprogrammed to accurately simulate the MC6805P2/P6. The MC6805P2/P6 read all "zeros" from this area.

2. The reserved ROM areas in the MC6805P2/P6 and the MC68705P5 have different data stored in them, and this data is subject to change without notice. The MC6805P2/P6 use the reserved ROM for the self-check feature, and the MC68705P5 uses this area for the bootstrap program.

3. The MC6805P2/P6 read all "ones" in its 48-byte "future RAM" area. This RAM is not implemented in the MC6805P2/P6 mask ROM versions but is implemented in the MC68705P5.

4. The V_{PP} line (pin 6) in the MC68705P5 must be tied to V_{CC} for normal operation. In the MC6805P2/P6, pin 6 is the NUM pin and is grounded in normal operation.

5. The LVI feature is not available in the MC68705P5. Processing differences are not presently compatible with proper design of this feature in the EPROM version.

The operation of all other circuitry has been exactly duplicated or designed to function identically in both devices including interrupts, timer, data ports, and data direction registers (DDRs). A design goal has been to provide the user with a safe, inexpensive way to verify a program and system design before committing to a factory programmed ROM.

INSTRUCTION SET

The MCU has a set of 59 basic instructions which can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following list of instructions.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (2's Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch

instructions are two-byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Higher	BHI
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
(Branch if Higher or Same)	(BHS)
Branch if Carry Set	BCS
(Branch if Lower)	(BLO)
Branch if Not Equal	BNE
Branch if Equal	BEQ
Branch if Half Carry Clear	BHCC
Branch if Half Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Bit is Clear	BMC
Branch if Interrupt Mask Bit is Set	BMS
Branch if Interrupt Line is Low	BIL
Branch if Interrupt Line is High	BIH
Branch to Subroutine	BSR

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the first 256 bytes of the memory space where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear and bit test, and branch functions are all implemented with a single instruction. For test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch if Bit n is Set	BRSET n (n=0 . . . 7)
Branch if Bit n is Clear	BRCLR n (n=0 . . . 7)
Set Bit n	BSET n (n=0 . . . 7)
Clear Bit n	BCLR n (n=0 . . . 7)

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No Operation	NOP

OPCODE MAP SUMMARY

Table 2 is an opcode map for the instructions used on the MCU.

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single-byte instructions, while the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. Two-byte direct-addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following

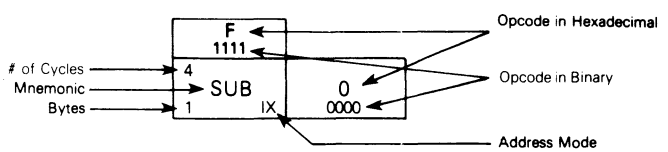
Table 2. Opcode Map

Hi / Low	Bit Manipulation BTB 0 0000	Bit Manipulation BSC 1 0001	Branch REL 2 0010	Read-Modify-Write DIR 3 0011	Read-Modify-Write INH 4 0100	Read-Modify-Write INH 5 0101	Read-Modify-Write IX1 6 0110	Read-Modify-Write IX 7 0111	Control INH 8 1000	Control INH 9 1001	Register/Memory IMM A 1010	Register/Memory DIR B 1011	Register/Memory EXT C 1100	Register/Memory IX2 D 1101	Register/Memory IX1 E 1110	Register/Memory IX F 1111	Hi / Low
0 0000	10 BRSET0 3 BTB	7 BSET0 2 BSC	4 BRA 2 REL	6 NEG 2 DIR	4 NEG 1 INH	4 NEG 1 INH	7 NEG 2 IX1	6 NEG 1 IX	9 RTI 1 INH		2 SUB 2 IMM	4 SUB 2 DIR	5 SUB 3 EXT	6 SUB 3 IX2	5 SUB 2 IX1	4 SUB 1 IX	0 0000
1 0001	10 BRCLR0 3 BTB	7 BCLR0 2 BSC	4 BRN 2 REL						6 RTS 1 INH		2 CMP 2 IMM	4 CMP 2 DIR	5 CMP 3 EXT	6 CMP 3 IX2	5 CMP 2 IX1	4 CMP 1 IX	1 0001
2 0010	10 BRSET1 3 BTB	7 BSET1 2 BSC	4 BHI 2 REL								2 SBC 2 IMM	4 SBC 2 DIR	5 SBC 3 EXT	6 SBC 3 IX2	5 SBC 2 IX1	4 SBC 1 IX	2 0010
3 0011	10 BRCLR1 3 BTB	7 BCLR1 2 BSC	4 BLS 2 REL	6 COM 2 DIR	4 COMA 1 INH	4 COMX 1 INH	7 COM 2 IX1	6 COM 1 IX	11 SWI 1 INH		2 CPX 2 IMM	4 CPX 2 DIR	5 CPX 3 EXT	6 CPX 3 IX2	5 CPX 2 IX1	4 CPX 1 IX	3 0011
4 0100	10 BRSET2 3 BTB	7 BSET2 2 BSC	4 BCC 2 REL	6 LSR 2 DIR	4 LSRA 1 INH	4 LSRX 1 INH	7 LSR 2 IX1	6 LSR 1 IX			2 AND 2 IMM	4 AND 2 DIR	5 AND 3 EXT	6 AND 3 IX2	5 AND 2 IX1	4 AND 1 IX	4 0100
5 0101	10 BRCLR2 3 BTB	7 BCLR2 2 BSC	4 BCS 2 REL								2 BIT 2 IMM	4 BIT 2 DIR	5 BIT 3 EXT	6 BIT 3 IX2	5 BIT 2 IX1	4 BIT 1 IX	5 0101
6 0110	10 BRSET3 3 BTB	7 BSET3 2 BSC	4 BNE 2 REL	6 ROR 2 DIR	4 RORA 1 INH	4 RORX 1 INH	7 ROR 2 IX1	6 ROR 1 IX			2 LDA 2 IMM	4 LDA 2 DIR	5 LDA 3 EXT	6 LDA 3 IX2	5 LDA 2 IX1	4 LDA 1 IX	6 0110
7 0111	10 BRCLR3 3 BTB	7 BCLR3 2 BSC	4 BEQ 2 REL	6 ASR 2 DIR	4 ASRA 1 INH	4 ASRX 1 INH	7 ASR 2 IX1	6 ASR 1 IX		2 TAX 1 INH		5 STA 2 DIR	6 STA 3 EXT	7 STA 3 IX2	6 STA 2 IX1	5 STA 1 IX	7 0111
8 1000	10 BRSET4 3 BTB	7 BSET4 2 BSC	4 BHCC 2 REL	6 LSL 2 DIR	4 LSLA 1 INH	4 LSLX 1 INH	7 LSL 2 IX1	6 LSL 1 IX		2 CLC 1 INH	2 EOR 2 IMM	4 EOR 2 DIR	5 EOR 3 EXT	6 EOR 3 IX2	5 EOR 2 IX1	4 EOR 1 IX	8 1000
9 1001	10 BRCLR4 3 BTB	7 BCLR4 2 BSC	4 BHCS 2 REL	6 ROL 2 DIR	4 ROLA 1 INH	4 ROLX 1 INH	7 ROL 2 IX1	6 ROL 1 IX		2 SEC 1 INH	2 ADC 2 IMM	4 ADC 2 DIR	5 ADC 3 EXT	6 ADC 3 IX2	5 ADC 2 IX1	4 ADC 1 IX	9 1001
A 1010	10 BRSET5 3 BTB	7 BSET5 2 BSC	4 BPL 2 REL	6 DEC 2 DIR	4 DECA 1 INH	4 DECX 1 INH	7 DEC 2 IX1	6 DEC 1 IX		2 CLI 1 INH	2 ORA 2 IMM	4 ORA 2 DIR	5 ORA 3 EXT	6 ORA 3 IX2	5 ORA 2 IX1	4 ORA 1 IX	A 1010
B 1011	10 BRCLR5 3 BTB	7 BCLR5 2 BSC	4 BMI 2 REL							2 SEI 1 INH	2 ADD 2 IMM	4 ADD 2 DIR	5 ADD 3 EXT	6 ADD 3 IX2	5 ADD 2 IX1	4 ADD 1 IX	B 1011
C 1100	10 BRSET6 3 BTB	7 BSET6 2 BSC	4 BMC 2 REL	6 INC 2 DIR	4 INCA 1 INH	4 INCX 1 INH	7 INC 2 IX1	6 INC 1 IX		2 RSP 1 INH		3 JMP 2 DIR	4 JMP 3 EXT	5 JMP 3 IX2	4 JMP 2 IX1	3 JMP 1 IX	C 1100
D 1101	10 BRCLR6 3 BTB	7 BCLR6 2 BSC	4 BMS 2 REL	6 TST 2 DIR	4 TSTA 1 INH	4 TSTX 1 INH	7 TST 2 IX1	6 TST 1 IX		2 NOP 1 INH	8 BSR 2 REL	7 JSR 2 DIR	8 JSR 3 EXT	9 JSR 3 IX2	8 JSR 2 IX1	7 JSR 1 IX	D 1101
E 1110	10 BRSET7 3 BTB	7 BSET7 2 BSC	4 BIL 2 REL								2 LDX 2 IMM	4 LDX 2 DIR	5 LDX 3 EXT	6 LDX 3 IX2	5 LDX 2 IX1	4 LDX 1 IX	E 1110
F 1111	10 BRCLR7 3 BTB	7 BCLR7 2 BSC	4 BIH 2 REL	6 CLR 2 DIR	4 CLRA 1 INH	4 CLRX 1 INH	7 CLR 2 IX1	6 CLR 1 IX		2 TXA 1 INH		5 STX 2 DIR	6 STX 3 EXT	7 STX 3 IX2	6 STX 2 IX1	5 STX 1 IX	F 1111

Abbreviations for Address Modes

INH	Inherent
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction.

RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −126 to +129 from the opcode address.

INDEX, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. This addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this two-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This addressing mode can be used in a manner similar to indexed, 8-bit offset except that this three-byte instruction allows tables to be anywhere in memory.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte to which the specified bit is to be set or cleared. Any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single two-byte instruction.

CAUTION

The corresponding DDRs for ports A, B, and C are write only registers (registers at $004, $005, and $006). A read operation on these registers always returns a "one". Since BSET and BCLR are read-modify-write functions, they cannot be used to set or clear a DDR bit (all "unaffected" bits would be set). It is recommended that all DDR bits in a port be written using a single-store instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit to be tested, and its condition (set or clear), is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltages EPROM Programming Voltage (V_{PP} Pin) TIMER Pin Normal Mode Bootstrap Programming Mode All Others	 V_{PP} V_{in} V_{in} V_{in}	 −0.3 to +22.0 −0.3 to +7.0 −0.3 to +15.0 −0.3 to +7.0	V
Operating Temperature Range	T_A	0 to +70	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature	T_J	+150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation it is recommended the V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{CC}$. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Cerdip	 θ_{JA}	 60	 °C/W

3

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
$P_D = P_{INT} + P_{I/O}$
$P_{INT} = I_{CC} \times V_{CC}$, Watts – Chip Internal Power
$P_{I/O}$ = Power Dissipation on Input and Output Pins – User Determined

For most applications $P_{I/O} < P_{INT}$ and can be neglected.

The following is an approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected):

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

PROGRAMMING OPERATION ELECTRICAL CHARACTERISTICS

(V_{CC} = 5.25 Vdc ±0.5, V_{SS} = 0 Vdc, T_A = 20° to 30°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Programming Voltage (V_{PP} Pin)	V_{PP}	20.0	21.0	22.0	V
V_{PP} Supply Current V_{PP} = 5.25 V V_{PP} = 21.0 V	I_{PP}	 — —	— —	 8 30	mA
Programming Oscillator Frequency	f_{oscp}	0.9	1.0	1.1	MHz
Bootstrap Programming Mode Voltage (TIMER Pin) I_{in} = 100 μA Max	V_{IHTP}	9.0	12.0	15.0	V

SWITCHING CHARACTERISTICS (V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0° to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Oscillator Frequency Normal	f_{osc}	0.4	—	4.2	MHz
Instruction Cycle Time ($4/f_{osc}$)	t_{cyc}	0.950	—	10	μs
$\overline{INT}$ or Timer Pulse Width (see Interrupt section)	t_{WL}, t_{WH}	t_{cyc} + 250	—	—	ns
$\overline{RESET}$ Pulse Width	t_{RWL}	t_{cyc} + 250	—	—	ns
$\overline{RESET}$ Delay Time (External Cap = 1.0 μF)	t_{RHL}	100	—	—	ms
$\overline{INT}$ Zero Crossing Detection Input Frequency	f_{INT}	0.03	—	1.0	kHz
External Clock Duty Cycle (EXTAL)	—	40	50	60	%

ELECTRICAL CHARACTERISTICS (V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0° to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage	V_{IH}				V
$\overline{RESET}$ (4.75 ≤ V_{CC} ≤ 5.75)		4.0	—	V_{CC}	
(V_{CC} < 4.75)		V_{CC} −0.5	—	V_{CC}	
$\overline{INT}$ (4.75 ≤ V_{CC} ≤ 5.75)		4.0	**	V_{CC}	
(V_{CC} < 4.75)		V_{CC} −0.5	**	V_{CC}	
All Other		2.0	—	V_{CC}	
Input High Voltage (TIMER Pin)	V_{IH}				V
Timer Mode		2.0	—	V_{CC}	
Bootstrap Programming Mode		9.0	12.0	15.0	
Input Low Voltage	V_{IL}				V
$\overline{RESET}$		−0.3	—	0.8	
$\overline{INT}$		−0.3	**	1.5	
All Other		−0.3	—	0.8	
Internal Power Dissipation (No Port Loading, V_{CC} = 5.25 V, T_A = 0°C)	P_{INT}	—	450	TBD	mW
Input Capacitance	C_{in}				pF
XTAL		—	25	—	
All Other		—	10	—	
$\overline{INT}$ Zero-Crossing Voltage, through a Capacitor	V_{INT}	2.0	—	4.0	V_{acp-p}
$\overline{RESET}$ Hysteresis Voltage			—	4.0	V
Out of Reset Voltage	V_{IRES+}	2.1	—	2.0	
Into Reset Voltage	V_{IRES-}	0.8			
Programming Voltage (V_{PP} Pin)	V_{PP}*		21.0		V
Programming EPROM		20.0	V_{CC}	22.0	
Operating Mode		4.0		5.75	
Input Current	I_{in}				μA
TIMER (V_{in}) = 0.4 V)		—	—	20	
$\overline{INT}$ (V_{in} = 0.4 V)		—	20	50	
EXTAL (V_{in}2.4 V to V_{CC} Crystal Option)		—	—	10	
(V_{in} = 0.4 V Crystal Option)		—	—	−1600	
$\overline{RESET}$ (V_{in} = 0.8 V)		−4.0	—	−40	
(External Capacitor Changing Current)					

*V_{PP} is pin 6 on the MC68705P5 and is connected to V_{CC} in the normal operating mode. In the MC6805P2, pin 6 is NUM and is connected to V_{SS} in the normal operating mode. The user must allow for this difference when emulating the MC6805P2 ROM-based MCU.

**Due to internal biasing, this input (when not used) floats to approximately 2.0 V.

PORT ELECTRICAL CHARACTERISTICS (V_{CC} = +5.25 Vdc, ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0° to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Port A					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 µA	V_{OH}	2.4	—	—	V
Output High Voltage, I_{Load} = −10 µA	V_{OH}	V_{CC} − 1.0	—	—	V
Input High Voltage, I_{Load} = −300 µA (Max)	V_{IH}	2.0	—	V_{CC} + 0.7	V
Input Low Voltage, I_{Load} = −500 µA (Max)	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current (V_{in} = 2.0 V to V_{CC})	I_{IH}	—	—	300	µA
Hi-Z State Input Current (V_{in} = 0.4 V)	I_{IL}	—	—	500	µA
Port B					
Output Low Voltage, I_{Load} = 3.2 mA	(V_{OL}	—	—	0.4	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.0	V
Output High Voltage, I_{Load} = −200 µA	V_{OH}	2.4	—	—	V
Darlington Current Drive (Source), V_O = 1.5 V	I_{OH}	−1.0	—	10	mA
Input High Voltage	V_{IH}	2.0	—	V_{CC} + 0.7	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	2	20	µA
Port C					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 µA	V_{OH}	2.4	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC} + 0.7	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	2	20	µA

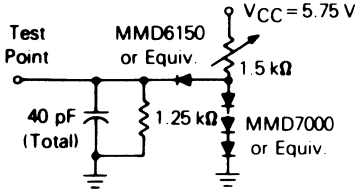


Figure 10. TTL Equivalent Test Load (Port B)

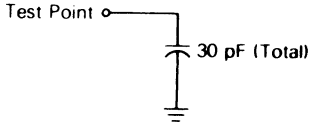


Figure 11. CMOS Equivalent Test Load (Port A)

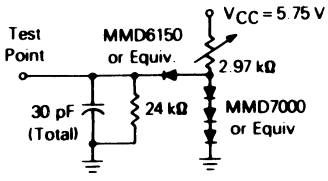


Figure 12. TTL Equivalent Test Load (Ports A and C)

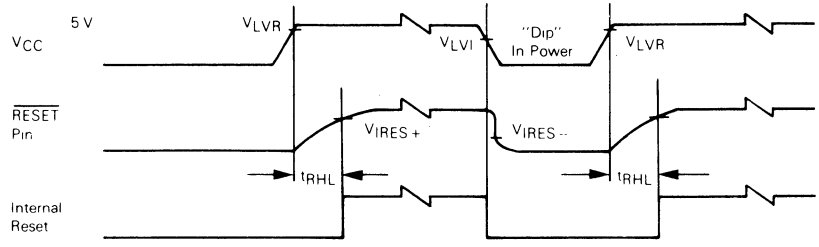


Figure 13. Power and Reset Timing

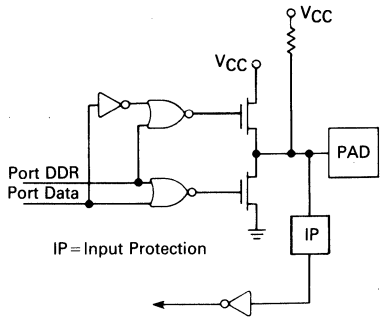


Figure 14. Port A Logic Diagram

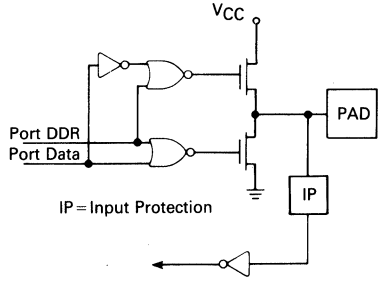


Figure 15. Port B and C Logic Diagram

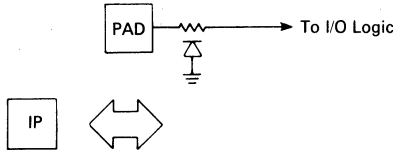


Figure 16. Typical Input Protection

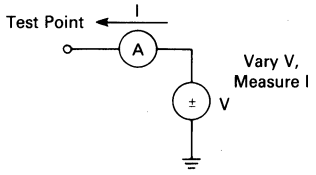


Figure 17. I/O Characteristic Measurement Circuit

ORDERING INFORMATION

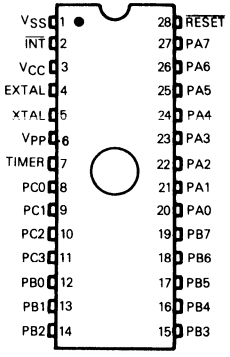
The following table provides generic information pertaining to the package type, temperature, and MC order numbers for the MC68705P5.

Table 3. Generic Information

Package Type	Internal Clock Frequency (MHz)	Temperature	Order Number
Cerdip (S Suffix)	1.0	0° to 70°C	MC68705P5S
Cerdip (S Suffix)	1.0	−40° to 85°C	MC68705P5CS

MECHANICAL DATA

PIN ASSIGNMENTS



3

MC68705R3

Technical Summary
8-Bit EPROM Microcontroller Unit

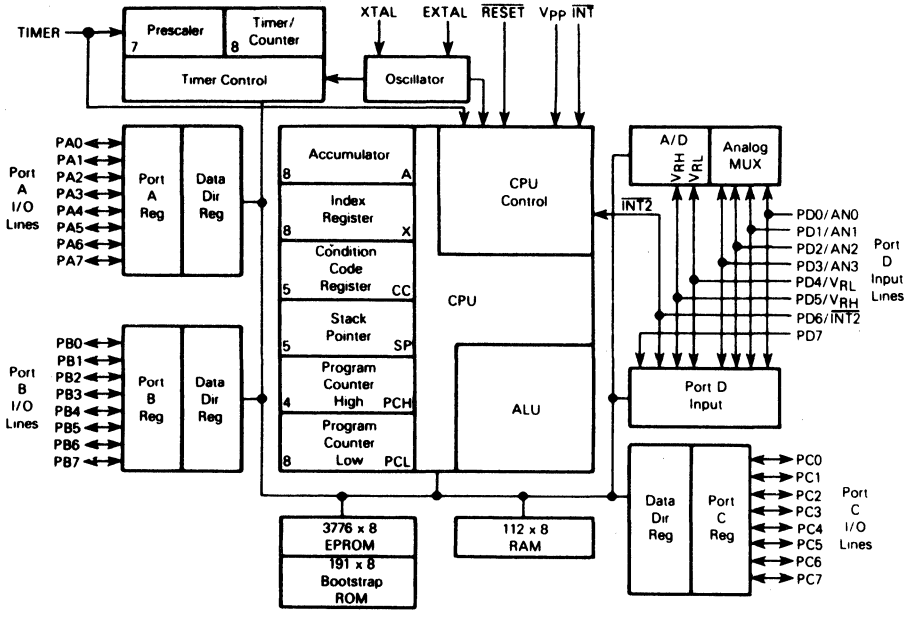
The MC68705R3 (HMOS) Microcontroller Unit (MCU) is an EPROM member of the MC6805 Family of microcontrollers. The user programmable EPROM allows program changes and lower volume applications. This low cost MCU has parallel I/O capability with pins programmable as input or output. This publication contains condensed information on the MCU; for detailed information, refer to *M6805 HMOS, M146805 CMOS Family User's Manual* (M6805UM(AD2)) or contact your local Motorola sales office.

Refer to the block diagram for the hardware features and to the list below for additional features available on the MCU.

- Internal 8-Bit Timer with 7-Bit Programmable Prescaler
- On-chip Oscillator
- Memory Mapped I/O
- Versatile Interrupt Handling
- Bit Manipulation
- Bit Test and Branch Instruction
- Vectored Interrupts
- Bootstrap Program in ROM
- 112 Bytes of RAM
- 3776 Bytes of Eprom
- 24 I/O Pins
- 4-Channel Analog-to-Digital Converter

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

V_{CC} AND V_{SS}

Power is supplied to the microcontroller using these two pins. V_{CC} is +5.25 volts (±0.5Δ) power, and V_{SS} is ground.

V_{PP}

This pin is used when programming the EPROM. In normal operation, this pin is connected to V_{CC}.

$\overline{INT}$

This pin provides the capability for asynchronously applying an external interrupt to the MCU. Refer to **INTERRUPTS** for more detailed information.

EXTAL, XTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a resistor/capacitor combination, or an external signal (depending on mask option register setting) is connected to these pins to provide a system clock.

RC Oscillator

With this option, a resistor is connected to the oscillator pins as shown in Figure 1. The relationship between R and f_{osc} is shown in Figure 2.

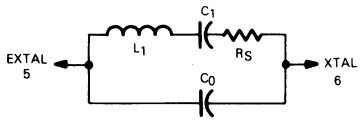
Crystal

The circuit shown in Figure 1 is recommended when using a crystal. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time. Refer to **ELECTRICAL SPECIFICATIONS** for V_{CC} specifications.

External Clock

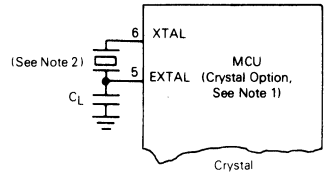
An external clock should be applied to the EXTAL input with the XTAL input connected to V_{SS}, as shown in Figure 1. This option may only be used with the crystal oscillator option selected in the mask option register.

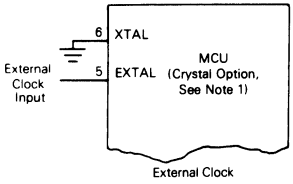
3

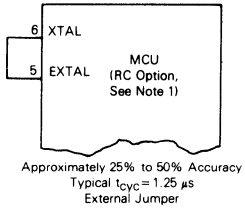


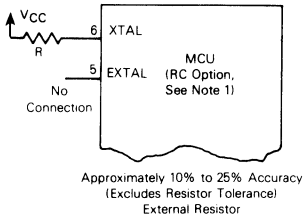
AT – Cut Parallel Resonance Crystal
C_0 = 7 pF Max
Freq. = 4.0 MHz @ C_L = 24 pF
R_S = 50 ohms Max.

Piezoelectric ceramic resonators which have the equivalent specifications may be used instead of crystal oscillators. Follow ceramic resonator manufacturer's suggestions for C_0, C_1, and R_S values.









NOTES:
1. For the MC68705R3 MOR b7 = 0 for the crystal option and MOR b7 = 1 for the RC option. When the TIMER input pin is in the V_{IHTP} range (in the bootstrap EPROM programming mode), the crystal option is forced. When the TIMER input is at or below V_{CC}, the clock generator option is determined by bit 7 of the mask option register (CLK).
2. The recommended C_L value with a 4.0 MHz crystal is 27 pF maximum, including system distributed capacitance. There is an internal capacitance of approximately 25 pF on the XTAL pin. For crystal frequencies other than 4 MHz, the total capacitance on each pin should be scaled as the inverse of the frequency ratio. For example, with a 2 MHz crystal, use approximately 50 pF on EXTAL and approximately 25 pF on XTAL. The exact value depends on the motional-arm parameters of the crystal used.

Figure 1. Oscillator Connections

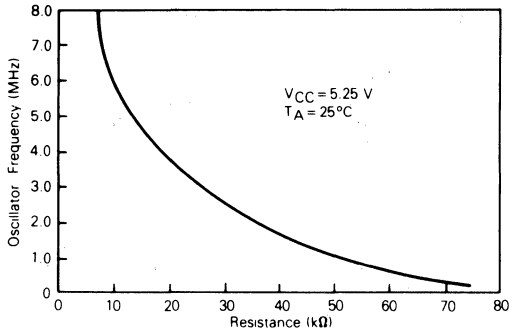


Figure 2. Typical Frequency vs Resistance for RC Oscillator Option Only

3

TIMER

This pin is used as an external input to control the internal timer/counter circuitry. This pin also detects a higher voltage level used to initiate the bootstrap program.

$\overline{\text{RESET}}$

This pin has a Schmitt trigger input and an on-chip pullup. The MCU can be reset by pulling $\overline{\text{RESET}}$ low.

INPUT/OUTPUT LINES (PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD7)

These 32 lines are arranged into four 8-bit ports (A, B, C, and D). Ports A, B, and C are programmable as either inputs or outputs under software control of the data direction registers. Port D is a fixed input port. It has up to four analog inputs, plus two voltage reference inputs when the analog-to-digital converter is used (PD5/V_{RH}, PD4/V_{RL}), and an $\overline{\text{INT2}}$ input. Port D lines can be read directly and used as binary inputs. If an analog input is used, then the voltage reference pins must be used in the analog mode. Refer to **PROGRAMMING** for additional information.

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

Ports A, B, and C are programmable as either input or output under software control of the corresponding data direction register (DDR). Port D lines are input only. The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic one for output and a logic zero for input. On reset, all the DDRs are initialized to a logic zero state to put the ports in the input mode. The port output registers are not initialized on reset and should be written to before setting the DDR bits.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This port write may be used to initialize the data registers and avoid undefined outputs. Care must be exercised when using read-modify-write instructions since the data read corresponds to the pin level if the DDR is an input (zero) and, also, to the latched output when the DDR is an output (one). Refer to Table 1 for I/O functions and to Figure 3 for typical port circuitry.

Table 1. I/O Pin Functions

Data Direction Register Bit	Latched Output Data Bit	Output State	Input To MCU
1	0	0	0
1	1	1	1
0	X	Hi-Z**	Pin

**Port B and C are three-state ports. Port A has an internal pullup devices to provide CMOS data drive capability.

Port D provides reference voltage and multiplexed analog inputs. The V_{RL} and V_{RH} lines are internally connected to the A/D resistor. Port D can always be used as

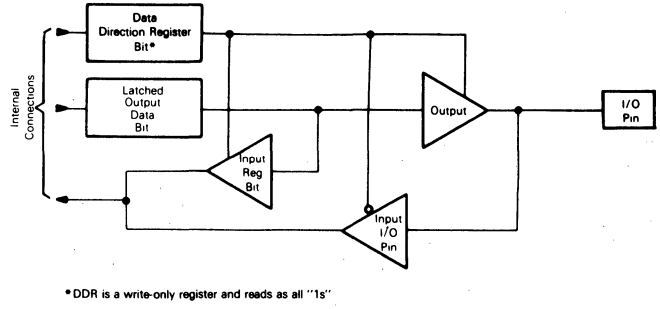


Figure 3. Typical Port I/O Circuitry and Register Configuration

digital inputs, but for analog inputs, V_{RH} and V_{RL} must be connected to the appropriate reference voltage.

NOTE

Read-modify-write instructions should be not used when writing to DDRs always read as 'one'.

MEMORY

The MCU is capable of addressing 4096 bytes of memory and I/O registers. The memory map is shown in Figure 4. The locations consists of user EPROM, bootstrap ROM, user RAM, a mask option register (MOR), a program control register, miscellaneous register, A/D control registers, and I/O. The interrupt vectors are located from $FF8 to $FFF. The bootstrap is a mask-programmed ROM that allows the MCU to program its own EPROM.

The stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer decrements during pushes and increments during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

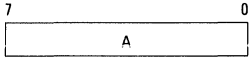
Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

REGISTERS

The MCU contains the registers described in the following paragraphs.

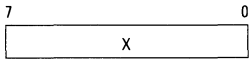
ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



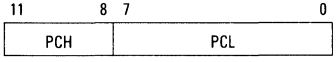
INDEX REGISTER (X)

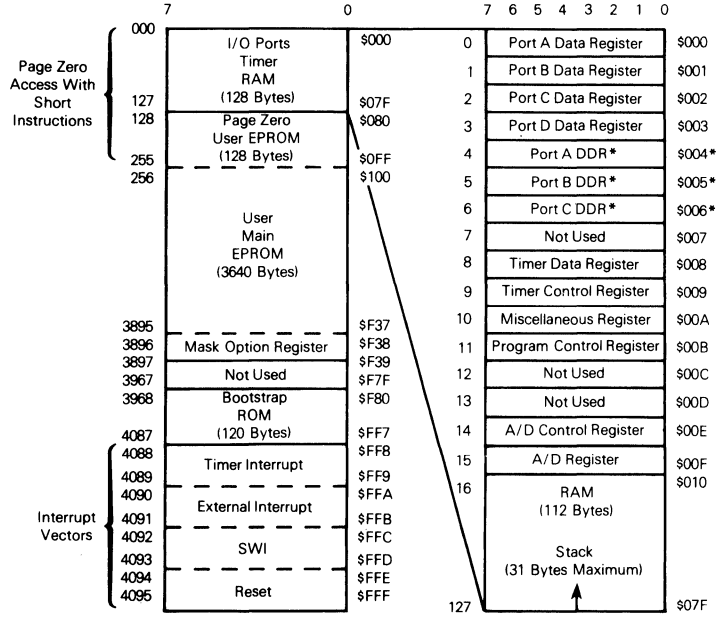
The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16- bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



PROGRAM COUNTER (PC)

The program counter is a 12-bit register that contains the address of the next byte to be fetched.





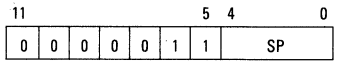
*Caution: Data direction registers (DDRs) are write-only; they read as $FF.

Figure 4. Memory Map

STACK POINTER (SP)

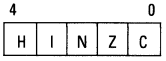
The stack pointer is a 12-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set at location $07F. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

The seven most-significant bits of the stack pointer are permanently set at 0000011. Subroutines and interrupts may be nested down to location $061 (31 bytes maximum), which allows the programmer to use up to 15 levels of subroutine calls (less if interrupts are allowed).



CONDITION CODE REGISTER (CC)

The condition code register is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program, and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.



Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, the timer and external interrupt is masked (disabled). If an external interrupt occurs while this bit is set, the interrupt is latched and is processed as soon as the interrupt bit is cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic 1).

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/Borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occured during the last arithmetic operation. This bit is also affected during bit test and branch instructions, and during shifts and rotates.

RESETS

The MCU can be reset two ways: by initial power-up and by the external reset input ($\overline{\text{RESET}}$). The $\overline{\text{RESET}}$ input consists mainly of a Schmitt trigger that senses the line logic level.

POWER-ON-RESET (POR)

An internal reset is generated on power-up that allows the internal clock generator to stabilize. The power-on reset is used strictly for power turn-on conditions and should not be used to detect any drop in the power supply voltatge. A delay of t_{RHL} milliseconds is required before allowing $\overline{\text{RESET}}$ input to go high. Connecting a capacitor to the $\overline{\text{RESET}}$ input (Figure 5) typically provides sufficient delay.

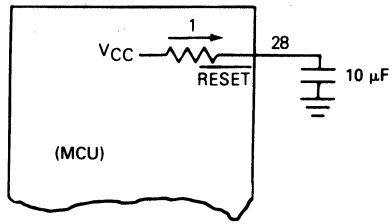


Figure 5. Power-Up $\overline{\text{RESET}}$ Delay Circuit

EXTERNAL $\overline{\text{RESET}}$ INPUT

The MCU is reset when a logic zero is applied to the $\overline{\text{RESET}}$ input for a period longer than one machine cycle (t_{cyc}). Under this type of reset, the Schmitt trigger switches off at V_{IRES-} to provide an internal reset voltage.

INTERRUPTS

The MCU can be interrupted four different ways: (1) through the external interrupt $\overline{\text{INT}}$ input pin, (2) with the internal timer interrupt request, (3) using the software interrupt instruction (SWI), or (4) the external Port D ($\overline{\text{INT2}}$) input pin.

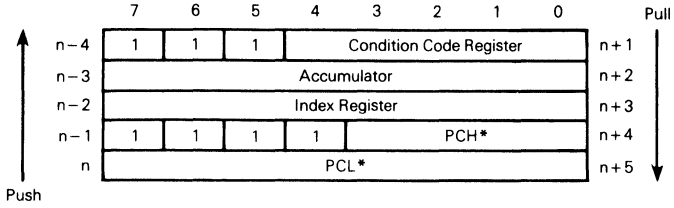
Interrupts cause the processor registers to be saved on the stack and the interrupt mask (I bit) set to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack after which normal processing resumes. The stacking order is shown in Figure 6.

Unlike $\overline{\text{RESET}}$, hardware interrupts do not cause the current instruction execution to be halted but are considered pending until the current instruction is complete.

NOTE

The current instruction is considered to be the one already fetched and being operated on.

When the current instruction is complete, the processor checks all pending hardward interrupts and, if unmasked (I bit clear), proceeds with interrupt processing; otherwise, the next instruction if fetched and executed. Masked interrupts are latched for later interrupt service. If the timer interrupt status bit is cleared before unmasking the interrupt, then the interrupt is not latched.



*For subroutine calls, only PCH and PCL are stacked.

Figure 6. Interrupt Stacking Order

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction regardless of the setting of the I bit. Refer to Figure 7 for the reset and interrupt instruction processing sequence.

TIMER INTERRUPT

If the time mask bit (TCR6) is cleared, then, each time the timer decrements to zero (transitions from $01 to $00), an interrupt request is generated. The actual processor interrupt is generated only if the interrupt mask bit of the condition code register (CCR) is also cleared. When the interrupt is recognized, the current state of the machine is pushed onto the stack and the I bit in the CCR is set, masking further interrupts until the present one is serviced. The contents of the timer interrupt vector, containing the location of the timer interrupt service routine, is then loaded into the program counter. At the end of the timer interrupt service routine, the software normally executes an RTI instruction which restores the machine state and starts executing the interrupted program. The timer interrupt status bit can only be cleared by software.

EXTERNAL INTERRUPT

The external interrupt is internally synchronized and then latched on the falling edge of INT and INT2. Clearing the I bit enables the external interrupt. The INT2 interrupt has an interrupt request bit (bit 7) and a mask bit (bit 6) in the miscellaneous register (MR). The INT2 interrupt is inhibited when the mask bit is set. The INT2 is always read as a digital input on port D. The INT2 and timer interrupt request bits, if set, cause the MCU to process an interrupt when the condition code I bit is clear. The following paragraphs describe two typical external interrupt circuits.

Zero-Crossing Interrupt

A sinusoidal input signal (f_{INT} maximum) can be used to generate an external interrupt (see Figure 8a) for use as a zero-crossing detector (for negative transitions of the ac sinusoid). This type of circuit allows applications such as servicing time-of-day routines and engaging/disengaging ac power control devices. Off-chip, full-wave rectification provides an interrupt at every zero crossing of the ac signal and, thereby, provides a 2f clock.

Digital-Signal Interrupt

With this type of circuit (Figure 8b), the $\overline{INT}$ pin can be driven by a digital signal. The maximum frequency of a signal that can be recognized by the TIMER or $\overline{INT}$ pin logic is dependent on the parameter labeled t_{WL}, t_{WH}. Refer to **TIMER** for additional information.

SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. The SWI execution is similar to the hardware interrupts.

MODES OF OPERATION

The MCU has two modes of operations. These modes are the normal and bootstrap. The following paragraphs describe the modes.

NORMAL MODE

This mode is a single-chip mode and is entered if the following conditions are met: (1) the $\overline{RESET}$ line is low, (2) the PC0 pin is within its normal operational range, and (3) the V_{PP} pin is connected to V_{CC}. The next rising edge of the $\overline{RESET}$ pin then causes the part to enter the normal mode.

BOOTSTRAP

The bootstrap mode is entered if the TIMER pin = +12 V. Refer to application note, *MC68705P3/R3/U3 8-Bit EPROM Microcomputer Programming Module* (AN-857 Rev.2).

TIMER

The MCU consists of an 8-bit software programmable counter driven by a 7-bit software programmable prescaler. The various timer sources are made via the timer control register (TCR) and/or the mask option register (MOR). The 8-bit counter may be loaded under program control and is decremented toward zero. When the timer reaches zero, the timer interrupt request bit (bit 7) in the timer control register (TCR) is set. Refer to Figure 9 for timer block diagram.

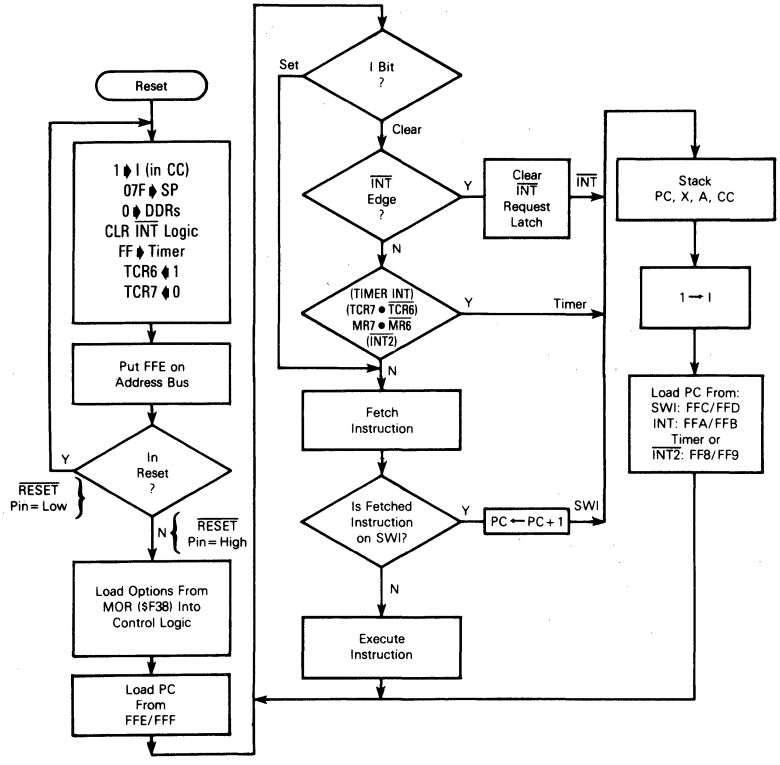


Figure 7. Reset and Interrupt Processing Flowchart

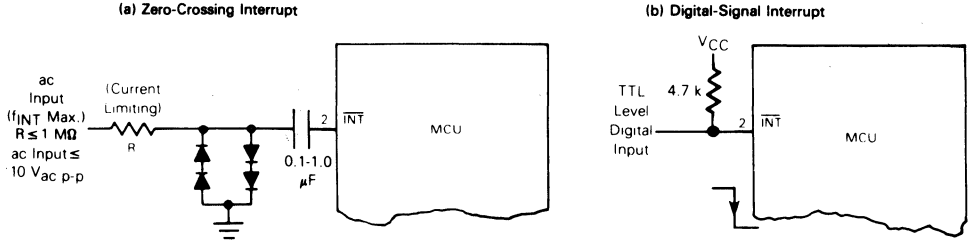
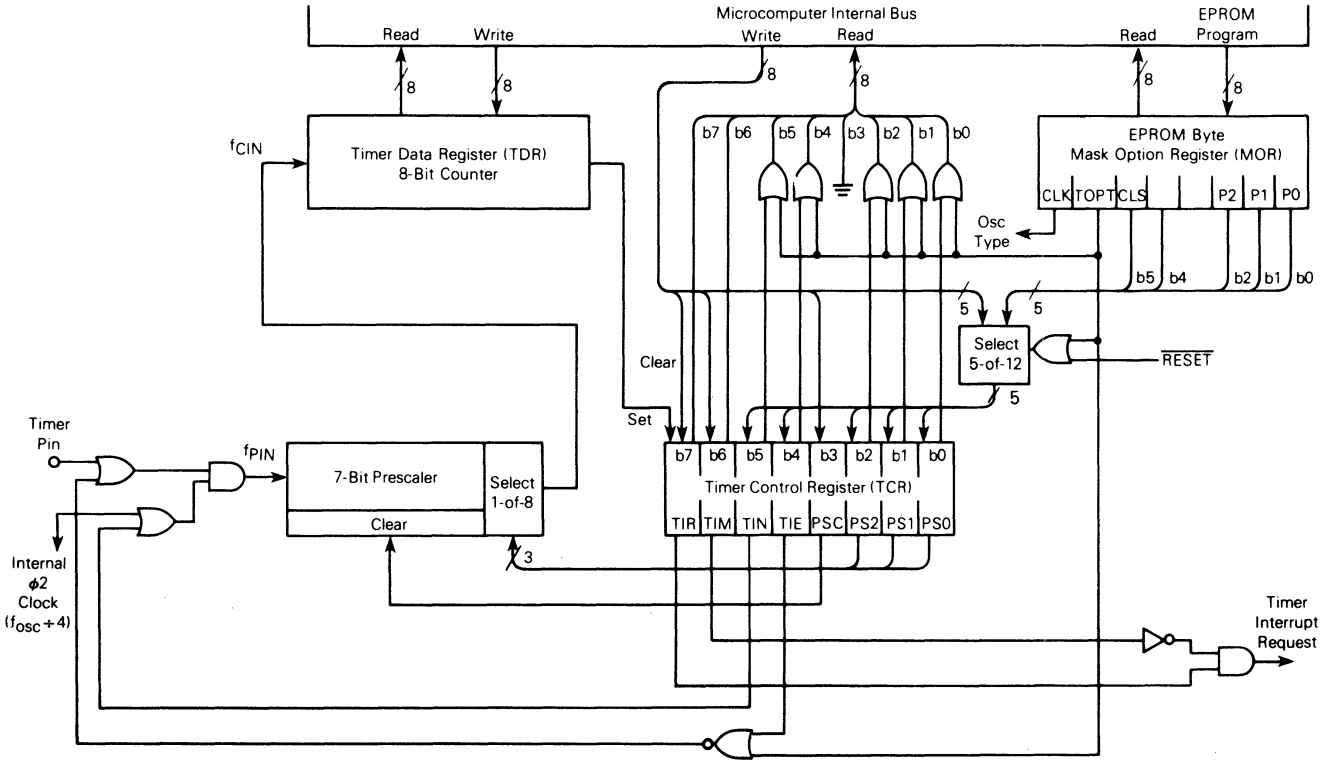


Figure 8. Typical Interrupt Circuits

The timer interrupt can be masked (disabled) by setting the timer interrupt mask bit (bit 6) in the TCR. When the I bit in the condition code register is cleared and the TCR bit 6 is cleared, the processor receives the interrupt. The MCU responds to this interrupt by (1) saving the present CPU state on the stack, (2) fetching the timer interrupt vector, and (3) executing the interrupt routine. The timer interrupt request bit must be cleared by software. Refer to **RESETS** and **INTERRUPTS** for additional information.

The prescaler is a 7-bit divider which is used to extend the maximum length of the timer. To avoid truncation errors, the prescaler is cleared when TCR bit 3 is set to a logic one; however, the TCR bit 3 always reads as a logic zero to ensure proper operation with read-modify-write instructions.



NOTE: The TOPT bit in the mask option register selects whether the timer is software programmable via the timer control register or emulates the mask programmable parts via the MOR EPROM byte.

Figure 9. Timer Block Diagram

The timer continues to count past zero, falling from $00 through $FF, and continues the countdown. The counter can be read at any time by reading the timer data register (TDR). This allows a program to determine the length of time since a timer interrupt has occurred without disturbing the counting process. The TDR is unaffected by reset.

SOFTWARE CONTROLLED MODE

This mode is selected when TOPT (bit 6) in the MOR is programmed to zero. The timer prescaler input can be configured for three different operating modes plus a disable mode, depending on the value written to TCR control bits 4 and 5 (TIE and TIN). The following paragraphs describe the different modes.

Timer Input Mode 1

When TIE and TIN are both programmed to zero, the timer input is from the internal clock (phase two) and the timer input pin is disabled. The internal clock mode can be used for periodic interrupt generation as well as a reference for frequency and event measurement.

Timer Input Mode 2

When TIE = 1 and TIN = 0, the internal clock and the timer input signals are ANDed to form the timer input. This mode can be used to measure external pulse widths. The active high, external pulse gates in the internal clock for the duration of the external pulse. The accuracy of the count is ±1.

Timer Input Mode 3

When TIE = 0 and TIN = 1, no prescaler input frequency is applied to the prescaler and the timer is disabled.

Timer Input Mode 4

When TIE and TIN are both one, the timer input is from the external clock. The external clock can be used to count external events as well as to provide an external frequency for generating periodic interrupts. Frequency of external input must be $\leq f_{osc}/8$.

MOR CONTROLLED MODE

This mode is selected when TOPT (bit 6) in the MOR is programmed to logic one. The timer circuits are the same as described in **SOFTWARE CONTROLLED MODE**. The logic levels of TCR bits 0, 1, 2, and 5 are determined during EPROM programming by the same bits in the MOR. Therefore, bits 0, 1, 2, and 5 in the MOR control the prescaler division and the timer clock selection. TIE (bit 4) and PSC (bit 3) in the TCR are set to a logic one when in the MOR controlled mode. TIM (bit 6) and TIR (bit 7) are controlled by the counter and software.

TIMER CONTROL REGISTER (TCR) $009

This is an 8-bit register that controls various functions such as configuring operation mode, setting ratio of the prescaler, and generating timer interrupt request signal. All bits are read/write except bit 3. The configuration of the TCR is determined by the TOPT (bit 6) in the MOR.

When TOPT = 1, the TCR emulates the MC6805R2; when TOPT = 0, the TCR is controlled by software.

TCR with MOR TOPT = 1

7	6	5	4	3	2	1	0
TIR	TIM	*	1	PSC	*	*	*

TCR with MOR TOPT = 0

7	6	5	4	3	2	1	0
TIR	TIM	TIN	TIE	PSC	PS2	PS1	PS0
RESET: 0	1	U	U	U	U	U	U

*The value of corresponding bits in MOR is written during $\overline{\text{RESET}}$ rising edge. These bits always read "one".

TIR — Timer Interrupt Request
Used to indicate the timer interrupt when it is logic one
1 = Set when the timer data register changes to all zeros
0 = Cleared by external reset, power-on reset, or under program control

TIM — Timer Interrupt Mask
Used to inhibit the timer interrupt
1 = Interrupt inhibited
0 = Interrupt enabled

TIN — External or Internal
Selects input clock source
1 = External clock selected
0 = Internal clock selected ($f_{osc}/4$)

TIE — TIMER External Enable
Used to enable external TIMER pin. When TOPT = 1, TIE is always a logical "one".
1 = Enables external timer pin
0 = Disables external timer pin

PSC — Prescaler Clear
Write only bit. Writing a one to this bit resets the prescaler to zero. A read of this location always indicates a zero when TOPT = 0. When TOPT = 1, this bit will read a logical "one" and has no effect on the prescaler.

PS2, PS1, PS0 — Prescaler Clear
Decoded to select one of eight outputs of the prescaler

Prescaler

PS2	PS1	PS0	Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

NOTES

When changing the PS bits in software, the PSC bit should be written to a "one" in the same write cycle to clear the prescaler. Changing the PS bits without clearing the prescaler may cause prescaler truncation.

MASK OPTION REGISTER (MOR) $F38

The MOR is implemented in EPROM. This register contains all zeros prior to programming and is not affected by reset. The MOR bits are described in the following paragraphs.

7	6	5	4	3	2	1	0
CLK	TOPT	CLS			P2	P1	P0

CLK — Clock (oscillator type)
1 = Resistor Capacitor (RC)
0 = Crystal

TOPT — Timer Option
1 = MC6805R2 type timer/prescaler. All bits except 6 and 7, of the TCR are invisible to the user. Bits 5, 2, 1, and 0 of the MOR determine the equivalent MC6805R2 mask options.
0 = All TCR bits are implemented as a software programmable timer. The state of MOR bits 5, 4, 2, 1, and 0 sets the initial values of their respective TCR bits.

CLS — Timer/Prescaler Clock Source
1 = External TIMER pin
0 = Internal clock

Bit 4
Not used if TOPT = 1. Sets the initial value of TIE in the TCR if TOPT = 0.
1 = Not used
0 = Sets initial value of TIE in the TCR

Bit 3
Not used

P2, P1, P0
The logical levels of these bits, when decoded, select one of eight outputs on the timer prescaler.

Prescaler

P2	P1	P0	Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

PROGRAMMING CONTROL REGISTER (PCR) $00B

The PCR is an 8-bit register which provides the necessary control bits to program the EPROM. The bootstrap program manipulates the PCR when programming so the user need not be concerned with PCR in most applications.

7	6	5	4	3	2	1	0
1	1	1	1	1	$\overline{\text{VPON}}$	$\overline{\text{PGE}}$	$\overline{\text{PLE}}$
RESET: U	U	U	U	U	U	1	1

$\overline{\text{PLE}}$ — Programming Latch Enable
Controls address and data being latched into the EPROM. Set during reset, but may be cleared anytime.
1 = Read EPROM
0 = Latch address and data on EPROM

PGE — Program Enable
Enables programming of EPROM. Must be set when changing the address and data. Set during reset.
1 = Inhibit EPROM programming
0 = Enable EPROM programming (if $\overline{\text{PLE}}$ is low)

$\overline{\text{VPON}}$ — V_{PP} On
A read-only bit that indicates high voltage at the V_{PP} pin. When set to "one", disconnects PGE and PLE from the chip.
1 = No high voltage on V_{PP} pin
0 = High voltage on V_{PP} pin

NOTE

$\overline{\text{VPON}}$ being "zero" does not indicate that the V_{PP} level is correct for programming. It is used as a safety interlock for the user in the normal operating mode.

$\overline{\text{VPON}}$	$\overline{\text{PGE}}$	$\overline{\text{PLE}}$	Programming Conditions
0	0	0	Programming mode (program EPROM byte)
1	0	0	$\overline{\text{PGE}}$ and $\overline{\text{PLE}}$ disabled from system
0	1	0	Programming disabled (latch address and data in EPROM)
1	1	0	$\overline{\text{PGE}}$ and $\overline{\text{PLE}}$ disabled from system
0	0	1	Invalid state; $\overline{\text{PGE}} = 0$ if $\overline{\text{PLE}} = 0$
1	0	1	Invalid state; $\overline{\text{PGE}} = 0$ if $\overline{\text{PLE}} = 0$
0	1	1	"High voltage" on V_{PP}
1	1	1	$\overline{\text{PGE}}$ and $\overline{\text{PLE}}$ disabled from system (operating mode)

EPROM PROGRAMMING

ERASING THE EPROM

The EPROM can be erased by exposure to high-intensity ultraviolet (UV) light with a wavelength of 2537 angstroms. The recommended integrated dose (UV intensity × exposure time) is 25Ws/cm^2. The lamps should be used without software filters, and the MCU should be positioned about one inch from the UV tubes. Ultraviolet erasure clears all bits of the MCU EPROM to the "zero" state. Data then can be entered by programming "ones" into the desired bit locations.

PROGRAMMING

The MCU bootstrap program can be used to program the MCU EPROM. The alternate vectoring used to implement the self-check is used to start execution of the bootstrap program.

A MCM2532 UV EPROM (other industry standard EPROMs may be used) must first be programmed with the same information that is to be transferred to the MCU EPROM. Refer to application note, *MC68705P3/R3/U3 8-bit EPROM Microcomputer Programming Module* (AN-857

3

Rev.2) for schematic diagrams and instructions on programming the MCU EPROM.

ANALOG-TO-DIGITAL CONVERTER

The chip resident 8-bit analog-to-digital (A/D) converter uses a successive approximation technique as show in Figure 10. Four external analog inputs can be connected to the A/D via Port D. Four internal analog channels ($V_{RH}-V_{RL}$, $V_{RH}-V_{RL}/2$, $V_{RH}-V_{RL}/4$, and V_{RL}) may be selected for calibration. The accuracy of these internal channels may not meet the accuracy specifications of the external channels.

Multiplexer selection is controlled by the A/D control register (ACR) bits 0, 1, and 2. Refer to Table 2 for multiplexer selection. The ACR is shown in Figure 10. The converter uses 30 machine cycles to complete a conversion of a sampled analog input. When the conversion is complete, the digital value is placed in the A/D result register (ARR); the conversion flag is set; selected input is sampled again; and a new conversion begins. When ACR7 is cleared, the conversion in progress is aborted and the selected input, which is held internally, is sampled for five machine cycles.

The converter uses V_{RH} and V_{RL} as reference voltages. An input voltage equal to or greater than V_{RH} converts to $FF. An input voltage equal to or less than V_{RL}, but greater than V_{SS}, converts to $00. Maximum and minimum ratings must not be exceeded. Each analog input source should use V_{RH} as the supply voltage and should be referenced to V_{RL} for the ratiometric conversion. To maintain full accuracy of the A/D, three requirements should be followed: (1) V_{RH} should be equal to or less than V_{CC}, (2) V_{RL} should be equal to or greater than V_{SS} but less than maximum specifications, and (3) $V_{RH}-V_{RL}$ should be equal to or greater than 4 volts.

The A/D has a built-in 1/2 LSB offset intended to reduce the magnitude of the quantizing error to ± 1/2 LSB, rather than +0, −1 LSB with no offset. This implies that, ignoring errors, the transition point from $00 to $01 occurs at 1/2 LSB above V_{RL}. Similarly, the transition from $FE to $FF occurs 1-1/2 LSB below V_{RH}, ideally.

3

Table 2. A/D Input MUX Selection

A/D Control Register			Input Selected	A/D Output (Hex)		
ACR2	ACR1	ACR0		Min	Typ	Max
0	0	0	AN0			
0	0	1	AN1			
0	1	0	AN2			
0	1	1	AN3			
1	0	0	V_{RH}*	FE	FF	FF
1	0	1	V_{RL}*	00	00	01
1	1	0	$V_{RH/4}$*	3F	40	41
1	1	1	$V_{RH/2}$*	7F	80	81

*Internal (calibration) levels

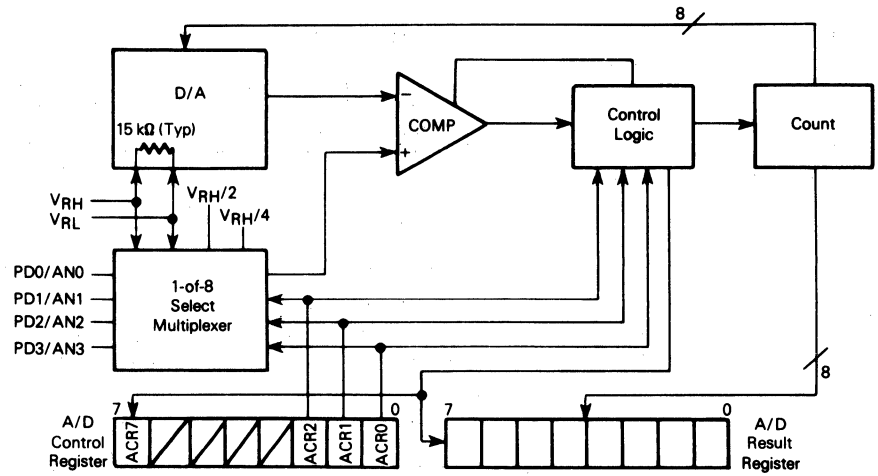


Figure 10. A/D Block Diagram

INSTRUCTION SET

The MCU has a set of 59 basic instructions which can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following instruction list.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the first 256 bytes of the memory space where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear and bit test, and branch functions are all implemented with a single instruction. For test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch if Bit n is Set	BRSET n (n=0 . . . 7)
Branch if Bit n is Clear	BRCLR n (n=0 . . . 7)
Set Bit n	BSET n (n=0 . . . 7)
Clear Bit n	BCLR n (n=0 . . . 7)

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (2's Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Higher	BHI
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
(Branch if Higher or Same)	(BHS)
Branch if Carry Set	BCS
(Branch if Lower)	(BLO)
Branch if Not Equal	BNE
Branch if Equal	BEQ
Branch if Half Carry Clear	BHCC
Branch if Half Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Bit is Clear	BMC
Branch if Interrupt Mask Bit is Set	BMS
Branch if Interrupt Line is Low	BIL
Branch if Interrupt Line is High	BIH
Branch to Subroutine	BSR

3

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No-Operation	NOP

OPCODE MAP SUMMARY

Table 3 is an opcode map for the instructions used on the MCU.

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single byte instructions, while the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. Two byte direct-addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction.

RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −126 to +129 from the opcode address.

INDEXED, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. This addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this two-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This addressing mode can be used in a manner similar to indexed, 8-bit offset except that this three-byte instruction allows tables to be anywhere in memory.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte to which the specified bit is to be set or cleared. Any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single two-byte instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit to be tested, and its condition (set or clear), is included in the opcode. The address of the byte to be tested is in

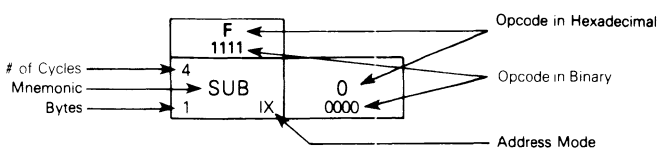
Table 3. Opcode Map

Hi / Low	Bit Manipulation BTB 0 0000	Bit Manipulation BSC 1 0001	Branch REL 2 0010	Read-Modify-Write DIR 3 0011	Read-Modify-Write INH 4 0100	Read-Modify-Write INH 5 0101	Read-Modify-Write IX1 6 0110	Read-Modify-Write IX 7 0111	Control INH 8 1000	Control INH 9 1001	Register/Memory IMM A 1010	Register/Memory DIR B 1011	Register/Memory EXT C 1100	Register/Memory IX2 D 1101	Register/Memory IX1 E 1110	Register/Memory IX F 1111	Hi / Low
0 0000	10 BRSET0 3 BTB	7 BSET0 2 BSC	4 BRA 2 REL	6 NEG 2 DIR	4 NEG 1 INH	4 NEG 1 INH	7 NEG 2 IX1	6 NEG 1 IX	9 RTI 1 INH		2 SUB 2 IMM	4 SUB 2 DIR	5 SUB 3 EXT	6 SUB 3 IX2	5 SUB 2 IX1	4 SUB 1 IX	0 0000
1 0001	10 BRCLR0 3 BTB	7 BCLR0 2 BSC	4 BRN 2 REL						6 RTS 1 INH		2 CMP 2 IMM	4 CMP 2 DIR	5 CMP 3 EXT	6 CMP 3 IX2	5 CMP 2 IX1	4 CMP 1 IX	1 0001
2 0010	10 BRSET1 3 BTB	7 BSET1 2 BSC	4 BHI 2 REL								2 SBC 2 IMM	4 SBC 2 DIR	5 SBC 3 EXT	6 SBC 3 IX2	5 SBC 2 IX1	4 SBC 1 IX	2 0010
3 0011	10 BRCLR1 3 BTB	7 BCLR1 2 BSC	4 BLS 2 REL	6 COM 2 DIR	4 COMA 1 INH	4 COMX 1 INH	7 COM 2 IX1	6 COM 1 IX	11 SWI 1 INH		2 CPX 2 IMM	4 CPX 2 DIR	5 CPX 3 EXT	6 CPX 3 IX2	5 CPX 2 IX1	4 CPX 1 IX	3 0011
4 0100	10 BRSET2 3 BTB	7 BSET2 2 BSC	4 BCC 2 REL	6 LSR 2 DIR	4 LSRA 1 INH	4 LSRX 1 INH	7 LSR 2 IX1	6 LSR 1 IX			2 AND 2 IMM	4 AND 2 DIR	5 AND 3 EXT	6 AND 3 IX2	5 AND 2 IX1	4 AND 1 IX	4 0100
5 0101	10 BRCLR2 3 BTB	7 BCLR2 2 BSC	4 BCS 2 REL								2 BIT 2 IMM	4 BIT 2 DIR	5 BIT 3 EXT	6 BIT 3 IX2	5 BIT 2 IX1	4 BIT 1 IX	5 0101
6 0110	10 BRSET3 3 BTB	7 BSET3 2 BSC	4 BNE 2 REL	6 ROR 2 DIR	4 RORA 1 INH	4 RORX 1 INH	7 ROR 2 IX1	6 ROR 1 IX			2 LDA 2 IMM	4 LDA 2 DIR	5 LDA 3 EXT	6 LDA 3 IX2	5 LDA 2 IX1	4 LDA 1 IX	6 0110
7 0111	10 BRCLR3 3 BTB	7 BCLR3 2 BSC	4 BEQ 2 REL	6 ASR 2 DIR	4 ASRA 1 INH	4 ASRX 1 INH	7 ASR 2 IX1	6 ASR 1 IX		2 TAX 1 INH		5 STA 2 DIR	6 STA 3 EXT	7 STA 3 IX2	6 STA 2 IX1	5 STA 1 IX	7 0111
8 1000	10 BRSET4 3 BTB	7 BSET4 2 BSC	4 BHCC 2 REL	6 LSL 2 DIR	4 LSLA 1 INH	4 LSLX 1 INH	7 LSL 2 IX1	6 LSL 1 IX		2 CLC 1 INH	2 EOR 2 IMM	4 EOR 2 DIR	5 EOR 3 EXT	6 EOR 3 IX2	5 EOR 2 IX1	4 EOR 1 IX	8 1000
9 1001	10 BRCLR4 3 BTB	7 BCLR4 2 BSC	4 BHCS 2 REL	6 ROL 2 DIR	4 ROLA 1 INH	4 ROLX 1 INH	7 ROL 2 IX1	6 ROL 1 IX		2 SEC 1 INH	2 ADC 2 IMM	4 ADC 2 DIR	5 ADC 3 EXT	6 ADC 3 IX2	5 ADC 2 IX1	4 ADC 1 IX	9 1001
A 1010	10 BRSET5 3 BTB	7 BSET5 2 BSC	4 BPL 2 REL	6 DEC 2 DIR	4 DECA 1 INH	4 DECX 1 INH	7 DEC 2 IX1	6 DEC 1 IX		2 CLI 1 INH	2 ORA 2 IMM	4 ORA 2 DIR	5 ORA 3 EXT	6 ORA 3 IX2	5 ORA 2 IX1	4 ORA 1 IX	A 1010
B 1011	10 BRCLR5 3 BTB	7 BCLR5 2 BSC	4 BMI 2 REL							2 SEI 1 INH	2 ADD 2 IMM	4 ADD 2 DIR	5 ADD 3 EXT	6 ADD 3 IX2	5 ADD 2 IX1	4 ADD 1 IX	B 1011
C 1100	10 BRSET6 3 BTB	7 BSET6 2 BSC	4 BMC 2 REL	6 INC 2 DIR	4 INCA 1 INH	4 INCX 1 INH	7 INC 2 IX1	6 INC 1 IX		2 RSP 1 INH		3 JMP 2 DIR	4 JMP 3 EXT	5 JMP 3 IX2	4 JMP 2 IX1	3 JMP 1 IX	C 1100
D 1101	10 BRCLR6 3 BTB	7 BCLR6 2 BSC	4 BMS 2 REL	6 TST 2 DIR	4 TSTA 1 INH	4 TSTX 1 INH	7 TST 2 IX1	6 TST 1 IX		2 NOP 1 INH	8 BSR 2 REL	7 JSR 2 DIR	8 JSR 3 EXT	9 JSR 3 IX2	8 JSR 2 IX1	7 JSR 1 IX	D 1101
E 1110	10 BRSET7 3 BTB	7 BSET7 2 BSC	4 BIL 2 REL								2 LDX 2 IMM	4 LDX 2 DIR	5 LDX 3 EXT	6 LDX 3 IX2	5 LDX 2 IX1	4 LDX 1 IX	E 1110
F 1111	10 BRCLR7 3 BTB	7 BCLR7 2 BSC	4 BIH 2 REL	6 CLR 2 DIR	4 CLRA 1 INH	4 CLRX 1 INH	7 CLR 2 IX1	6 CLR 1 IX		2 TXA 1 INH		5 STX 2 DIR	6 STX 3 EXT	7 STX 3 IX2	6 STX 2 IX1	5 STX 1 IX	F 1111

Abbreviations for Address Modes

INH	Inherent
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage EPROM Programming Voltage (VPP Pin) TIMER Pin — Normal Mode TIMER Pin — Bootstrap Programming Mode All Others	 V_{PP} V_{in} V_{in} V_{in}	 −0.3 to +22.0 −0.3 to +7.0 −0.3 to +15.0 −0.3 to +7.0	V
Operating Temperature Range MC68705R3 MC68705R3C	T_A	T_L to T_H 0 to +70 −40 to 85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature Plastic Cerdip	T_J	 150 175	°C/W

These devices contain circuity to protect the inputs against damage due to high static voltages or electrical fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high-impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ and $V_{out}) \leq V_{CC}$. Reliability of operation is enhanced if unused inputs except EXTAL are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic (P Suffix) Plastic (FN Suffix) Cerdip (S Suffix)	θ_{JA}	 50 100 60	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
$P_D = P_{INT} + P_{PORT}$
$P_{INT} = I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A

PROGRAMMING OPERATION ELECTRICAL CHARACTERISTICS

(V_{CC} = 5.25 Vdc ±0.5%, V_{SS} = 0, T_A = 20 to 30°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Programming Voltage	V_{PP}	20.0	21.0	22.0	V
V_{PP} Supply Current V_{PP} = 5.25 V V_{PP} = 21.0 V	I_{PP}	 — —	 — —	 8 30	mA
Oscillator Frequency	$f_{osc(p)}$	0.9	1.0	1.1	MHz
Bootstrap Programming Mode Voltage (TIMER Pin) @ I_{IHTP} = 100 µA Maximum	V_{IHTP}	9.0	12.0	15.0	V

ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0° to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage $\overline{RESET}$ (4.75 ≤ V_{CC} ≤ 5.75) (V_{CC} < 4.75) $\overline{INT}$ 4.75 ≤ V_{CC} < 5.75) (V_{CC} < 4.75) All Other	V_{IH}	 4.0 V_{CC} − 0.5 4.0 V_{CC} − 0.5 2.0	 — — ** ** —	 V_{CC} V_{CC} V_{CC} V_{CC} V_{CC}	V
Input High Voltage (TIMER Pin) Timer Mode Bootstrap Programming Mode	V_{IH}	 2.0 9.0	 — 12.0	 V_{CC} + 1.0 15.0	V
Input Low Voltage $\overline{RESET}$ $\overline{INT}$ All Other	V_{IL}	 V_{SS} V_{SS} V_{SS}	 — ** —	 0.8 1.5 0.8	V
$\overline{INT}$ Zero-Crossing Input Voltage — Through a Capacitor	V_{INT}	2.0	—	4.0	$V_{ac\ p\text{-}p}$
Internal Power Dissipation (No Port Loading, V_{CC} = 5.25 V for Steady-State Operation) T_A = 0°C T_A = 40°C	P_{INT}	— —	520 580	740 800	mW
Input Capacitance EXTAL All Other (See Note)	C_{in}	 — —	 25 10	 — —	pF
$\overline{RESET}$ Hysteresis Voltage Out of Reset Voltage Into Reset Voltage	 V_{IRES+} V_{IRES-}	 2.1 0.8	 — —	 4.0 2.0	V
Programming Voltage (V_{PP} Pin) Programming EPROM Operating Voltage	V_{PP}*	 20.0 4.75	 21.0 V_{CC}	 22.0 5.75	V
Input Current TIMER (V_{in} = 0.4 V) $\overline{INT}$ (V_{in} = 0.4 V) EXTAL (V_{in} = 2.4 V to V_{CC}) (V_{in} = 0.4 V) $\overline{RESET}$ (V_{in} = 0.8 V) (External Capacitor Changing Current)	 I_{in} I_{RES}	 — — — — −4.0	 — 20 — — —	 20 50 10 −1600 −40	µA

*V_{PP} (pin 7) is connected to V_{CC} in the normal operating mode.
**Due to internal biasing, this input (when not used) floats to approximately 2.0 V.

NOTE: Port D analog inputs, when selected, C_{in} = 25 pF for the first 5 out of 30 cycles.

SWITCHING CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Oscillator Frequency Normal	f_{osc}	 0.4	 —	 4.2	MHz
Instruction Cycle Time ($4/f_{osc}$)	t_{cyc}	0.950	—	10	μs
$\overline{INT}$, $\overline{INT2}$, or Timer Pulse Width	t_{WL}, t_{WH}	t_{cyc} + 250	—	—	ns
$\overline{RESET}$ Pulse Width	t_{RWL}	t_{cyc} + 250	—	—	ns
$\overline{RESET}$ Delay Time (External Cap = 1.0 μF)	t_{RHL}	—	100	—	ms
$\overline{INT}$ Zero Crossing Detection Input Frequency	f_{INT}	0.03	—	1.0	kHz
External Clock Duty Cycle (EXTAL)	—	40	50	60	%
Crystal Oscillator Start-Up Time	—	—	—	100	ms

A/D CONVERTER CHARACTERISTICS

(V_{CC} = +5.25 V ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Min	Typ	Max	Unit	Comments
Resolution	8	8	8	Bits	
Non-Linearity	—	—	±1/2	LSB	For V_{RH} = 4.0 to 5.0 V and V_{RL} = 0 V.
Quantizing Error	—	—	±1/2	LSB	
Conversion Range	V_{RL}	—	V_{RH}	V	
V_{RH} V_{RL}	— V_{SS}	— —	V_{CC} 0.2	V V	A/D accuracy may decrease proportionately as V_{RH} is reduced be below 4.0 V. The sum of V_{RH} and V_{RL} must not exceed V_{CC}.
Conversion Time	30	30	30	t_{cyc}	Includes sampling time
Monotonicity	Inherent (within total error)				
Zero Input Reading	00	00	01	hexadecimal	V_{in} = 0
Ratiometric Reading	FE	FF	FF	hexadecimal	V_{in} = V_{RH}
Sample Time	5	5	5	t_{cyc}	
Sample/Hold Capacitance, Input	—	—	25	pF	
Analog Input Voltage	V_{RL}	—	V_{RH}	V	Negative transients on any analog lines (pins 19-24) are not allowed at any time during conversion.

PORT ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0° to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Port A					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Output High Voltage, I_{Load} = −10 μA	V_{OH}	V_{CC} − 1.0	—	—	V
Input High Voltage, I_{Load} = −300 μA (Max)	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage, I_{Load} = −500 μA (Max)	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current (V_{in} = 2.0 V to V_{CC})	I_{IH}	—	—	−300	μA
Hi-Z State Input Current (V_{in} = 0.4 V)	I_{IL}	—	—	−500	μA
Port B					
Output Low Voltage, I_{Load} = 3.2 mA	V_{OL}	—	—	0.4	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.0	V
Output High Voltage, I_{Load} = −200 μA	V_{OH}	2.4	—	—	V
Darlington Current Drive (Source), V_O = 1.5 V	I_{OH}	−1.0	—	−10	mA
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port C					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port D (Input Only)					
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Input Current	I_{in}	—	<1	5	μA

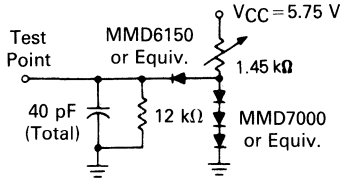


Figure 11. TTL Equivalent Test Load (Port B)

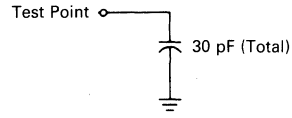


Figure 12. CMOS Equivalent Test Load (Port A)

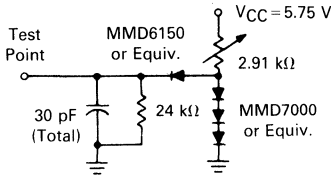


Figure 13. TTL Equivalent Test Load (Ports A and C)

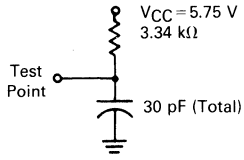


Figure 14. Open-Drain Equivalent Test Load (Port C)

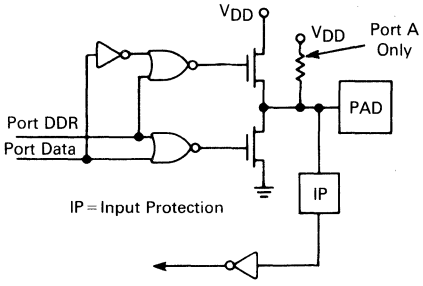


Figure 15. Ports A and C Logic Diagram

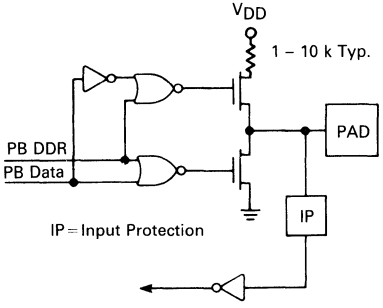
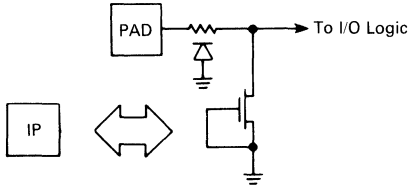


Figure 16. Port B Logic Diagram



Port 17. Typical Input Protection

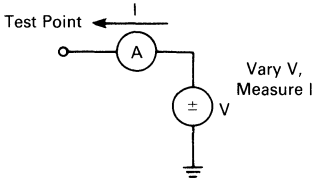


Figure 18. I/O Characteristic Measurement Circuit

ORDERING INFORMATION

The following table provides generic information pertaining to the package type, temperature, and MC order numbers for the MC68705R3.

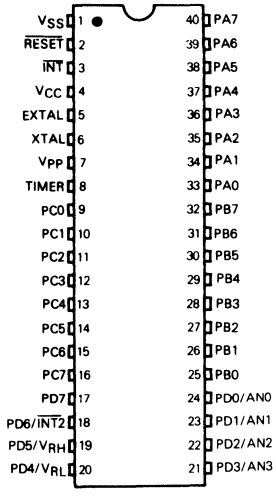
Table 4. Generic Information

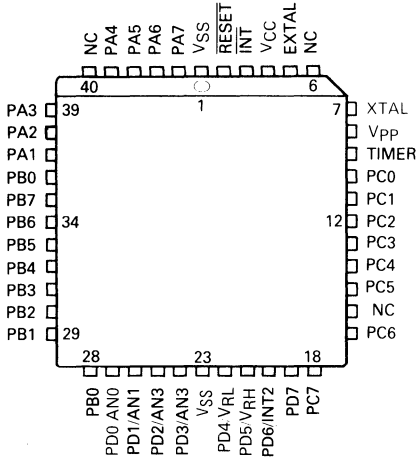
Package Type	Temperature	Order Number
Cerdip S Suffix	0°C to 70°C −40° to +85°C	MC68705R3S MC68705R3CS
Plastic P Suffix	0°C to 70°C −40°C to 85°C	MC68705R3P MC68705R3CP
PLCC FN Suffix	−40°C to +85°C	MC68705R3CFN

MECHANICAL DATA

3

PIN ASSIGNMENTS





MC68705R5

Technical Summary

8-Bit EPROM Microcontroller Unit

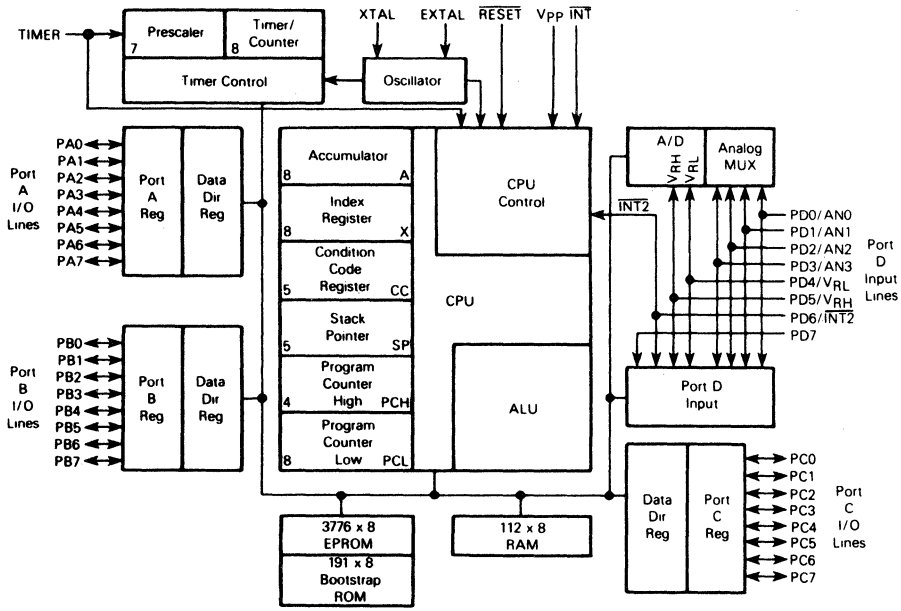
The MC68705R5 (HMOS) Microcontroller Unit (MCU) is an EPROM member of the MC6805 Family of microcontrollers. The user programmable EPROM allows program changes and lower volume applications. This low cost MCU has parallel I/O capability with pins programmable as input or output. This publication contains condensed information on the MCU; for detailed information, refer to *M6805 HMOS, M146805 CMOS Family User's Manual* (M6805UM(AD2)) or contact your local Motorola sales office.

Refer to the block diagram for the hardware features and to the list below for additional features available on the MCU.

- Internal 8-Bit Timer with 7-Bit Programmable Prescaler
- On-chip Oscillator
- Memory Mapped I/O
- Versatile Interrupt Handling
- Bit Manipulation
- Bit Test and Branch Instruction
- Vectored Interrupts
- Bootstrap program in ROM
- 112 Bytes of RAM
- 3776 Bytes of Eprom
- 24 I/O Pins
- 4-Channel Analog-to-Digital Converter
- EPROM Security Feature

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

V_{CC} AND V_{SS}

Power is supplied to the microcontroller using these two pins. V_{CC} is +5.25 volts (±0.5Δ) power, and V_{SS} is ground.

V_{PP}

This pin is used when programming the EPROM. In normal operation, this pin is connected to V_{CC}.

$\overline{INT}$

This pin provides the capability for asynchronously applying an external interrupt to the MCU. Refer to **INTERRUPTS** for more detailed information.

EXTAL, XTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a resistor/capacitor combination, or an external signal (depending on mask option register setting) is connected to these pins to provide a system clock.

RC Oscillator

With this option, a resistor is connected to the oscillator pins as shown in Figure 1. The relationship between R and f_{osc} is shown in Figure 2.

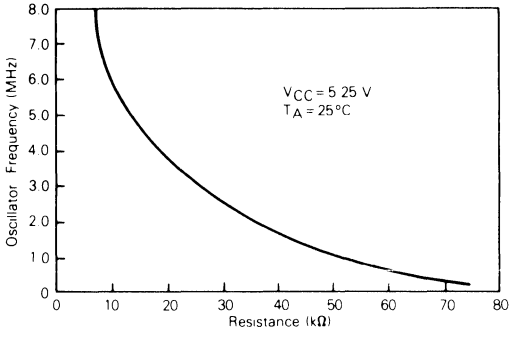


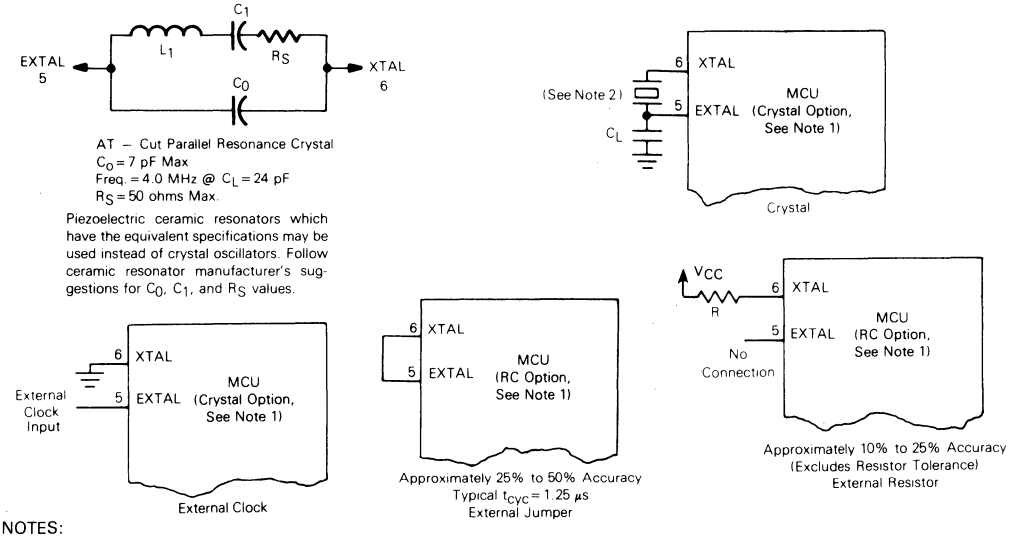
Figure 2. Typical Frequency vs Resistance for RC Oscillator Option Only

Crystal

The circuit shown in Figure 1 is recommended when using a crystal. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time. Refer to **ELECTRICAL SPECIFICATIONS** for V_{CC} specifications.

External Clock

An external clock should be applied to the EXTAL input with the XTAL input connected to V_{SS}, as shown in Figure 1. This option may only be used with the crystal oscillator option selected in the mask option register.



NOTES:

1. For the MC68705R5 MOR b7 = 0 for the crystal option and MOR b7 = 1 for the RC option. When the TIMER input pin is in the V_{IHTP} range (in the bootstrap EPROM programming mode), the crystal option is forced. When the TIMER input is at or below V_{CC}, the clock generator option is determined by bit 7 of the Mask Option Register (CLK).
2. The recommended C_L value with a 4.0 MHz crystal is 27 pF maximum, including system distributed capacitance. There is an internal capacitance of approximately 25 pF on the XTAL pin. For crystal frequencies other than 4 MHz, the total capacitance on each pin should be scaled as the inverse of the frequency ratio. For example, with a 2 MHz crystal, use approximately 50 pF on EXTAL and approximately 25 pF on XTAL. The exact value depends on the Motional-Arm parameters of the crystal used.

Figure 1. Oscillator Connections

TIMER

This pin is used as an external input to control the internal timer/counter circuitry. This pin also detects a higher voltage level used to initiate the bootstrap program.

$\overline{\text{RESET}}$

This pin has a Schmitt trigger input and an on-chip pullup. The MCU can be reset by pulling $\overline{\text{RESET}}$ low.

INPUT/OUTPUT LINES (PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD7)

These 32 lines are arranged into four 8-bit ports (A, B, C, and D). Ports A, B, and C are programmable as either inputs or outputs under software control of the data direction registers. Port D is a fixed input port. It has up to four analog inputs, plus two voltage reference inputs when the analog-to-digital converter is used (PD5/V_{RH}, PD4/V_{RL}), and an $\overline{\text{INT2}}$ input. Port D lines can be read directly and used as binary inputs. If an analog input is used, then the voltage reference pins must be used in the analog mode. Refer to **PROGRAMMING** for additional information.

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

Ports A, B, and C are programmable as either input or output under software control of the corresponding data direction register (DDR). Port D lines are input only. The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic one for output and a logic zero for input. On reset, all the DDRs are initialized to a logic zero state to put the ports in the input mode. The port output registers are not initialized on reset and should be written to before setting the DDR bits.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This port write may be used to initialize the data registers and avoid undefined outputs. Care must be exercised when using read-modify-write instructions since the data read corresponds to the pin level if the DDR is an input (zero) and, also, to the latched output when the DDR is an output (one). See Table 1 for I/O functions and to Figure 3 for typical port circuitry.

Port D provides reference voltage and multiplexed analog inputs. The V_{RL} and V_{RH} lines are internally connected to the A/D resistor. Port D can always be used as digital inputs, but for analog inputs, V_{RH} and V_{RL} must be connected to the appropriate reference voltage.

NOTE

Read-modify-write instructions should not be used when writing to the DDR because DDRs always read as 'one'.

Table 1. I/O Pin Functions

Data Direction Register Bit	Latched Output Data Bit	Output State	Input To MCU
1	0	0	0
1	1	1	1
0	X	Hi-Z**	Pin

**Port B and C are three-state ports. Port A has an internal pullup devices to provide CMOS data drive capability.

MEMORY

The MCU is capable of addressing 4096 bytes of memory and I/O registers. The memory map is shown in Figure 4. The location consist of user EPROM, bootstrap ROM, user RAM, a mask option register (MOR), a program control register, miscellaneous register, A/D control registers, and I/O. The interrupt vectors are located from $FF8 to $FFF. The bootstrap is a mask-programmed ROM that allows the MCU to program its own EPROM.

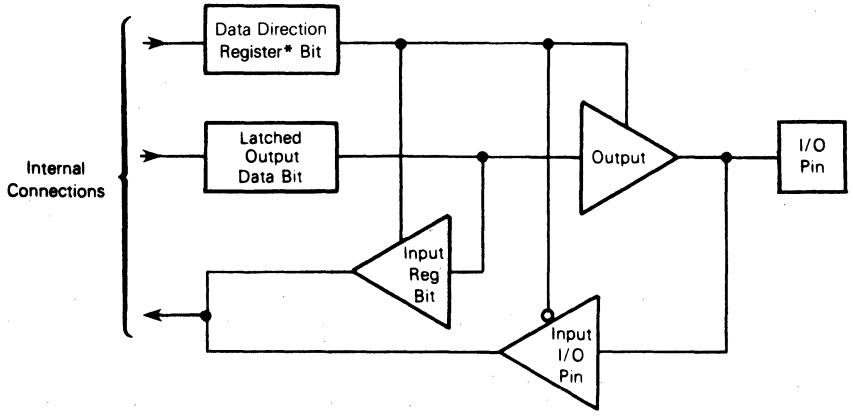
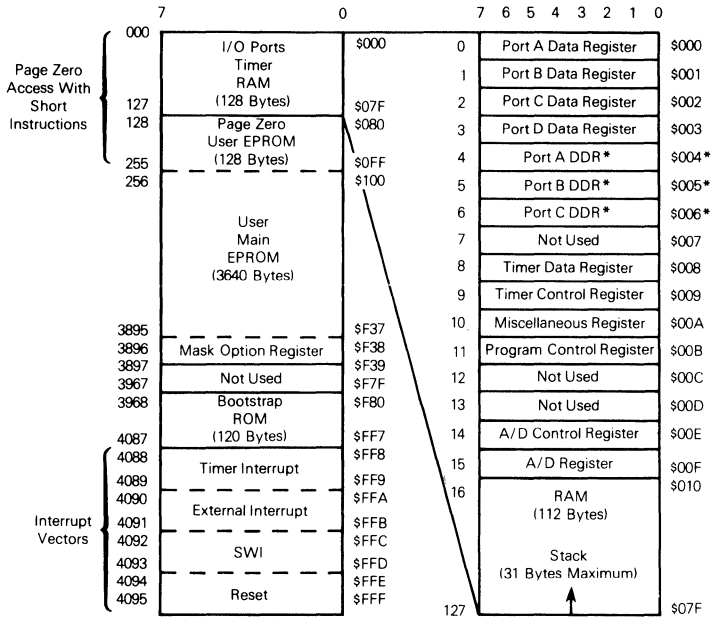


Figure 3. Typical Port I/O Circuitry and Register Configuration



*Caution: Data direction registers (DDRs) are write-only; they read as $FF.

Figure 4. Memory Map

The stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer decrements during pushes and increments during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

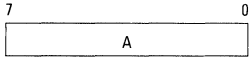
Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

REGISTERS

The MCU contains the registers described in the following paragraphs.

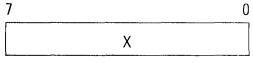
ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



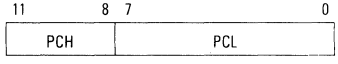
INDEX REGISTER (X)

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16- bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



PROGRAM COUNTER (PC)

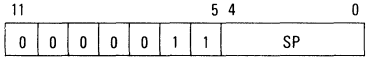
The program counter is an 12-bit register that contains the address of the next byte to be fetched.



STACK POINTER (SP)

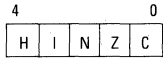
The stack pointer is an 12-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set at location $07F. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

The seven most-significant bits of the stack pointer are permanently set at 0000011. Subroutines and interrupts may be nested down to location $061 (31 bytes maximum), which allows the programmer to use up to 15 levels of subroutine calls (less if interrupts are allowed).



CONDITION CODE REGISTER (CC)

The condition code register is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program, and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.



Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, the timer and external interrupt is masked (disabled). If an external interrupt occurs while this bit is set, the interrupt is latched and is processed as soon as the interrupt bit is cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic 1).

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/Borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occured during the last arithmetic operation. This bit is also affected during bit test and branch instructions, and during shifts and rotates.

RESETS

The MCU can be reset two ways: by initial power-up and by the external reset input ($\overline{\text{RESET}}$). The $\overline{\text{RESET}}$ input consists mainly of a Schmitt trigger that senses the reset line logic level.

POWER-ON-RESET (POR)

An internal reset is generated on power-up that allows the internal clock generator to stabilize. The power-on reset is used strictly for power turn-on conditions and should not be used to detect any drop in the power supply voltage. A delay of t_{RHL} milliseconds is required before allowing $\overline{\text{RESET}}$ input to go high. Connecting a capacitor to the $\overline{\text{RESET}}$ input (Figure 5) typically provides sufficient delay.

EXTERNAL $\overline{\text{RESET}}$ INPUT

The MCU is reset when a logic zero is applied to the $\overline{\text{RESET}}$ input for a period longer than one machine cycle (t_{cyc}). Under this type of reset, the Schmitt trigger switches off at V_{IRES-} to provide an internal reset voltage.

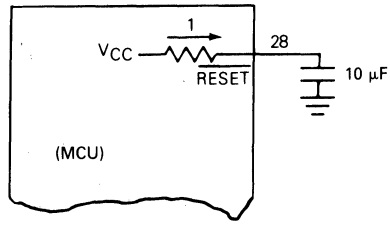


Figure 5. Power-Up $\overline{\text{RESET}}$ Delay Circuit

INTERRUPTS

The MCU can be interrupted four different ways: (1) through the external interrupt $\overline{\text{INT}}$ input pin, (2) with the internal timer interrupt request, (3) using the software interrupt instruction (SWI), or (4) the external Port D ($\overline{\text{INT2}}$) input pin.

Interrupts cause the processor registers to be saved on the stack and the interrupt mask (I bit) set to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack after which normal processing resumes. The stacking order is shown in Figure 6.

Unlike $\overline{\text{RESET}}$, hardware interrupts do not cause the current instruction execution to be halted but are considered pending until the current instruction is complete.

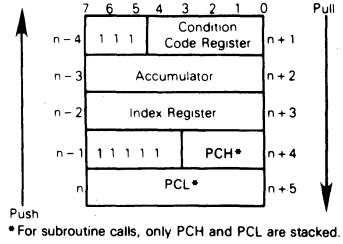


Figure 6. Interrupt Stacking Order

NOTE

The current instruction is considered to be the one already fetched and being operated on.

When the current instruction is complete, the processor checks all pending hardward interrupts and, if unmasked (I bit clear), proceeds with interrupt processing; otherwise, the next instruction if fetched and executed. Masked interrupts are latched for later interrupt service. If the timer interrupt status bit is cleared before unmasking the interrupt, then the interrupt is not latched.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction regardless of the setting of the I bit. Refer to Figure 7 for the reset and interrupt instruction processing sequence.

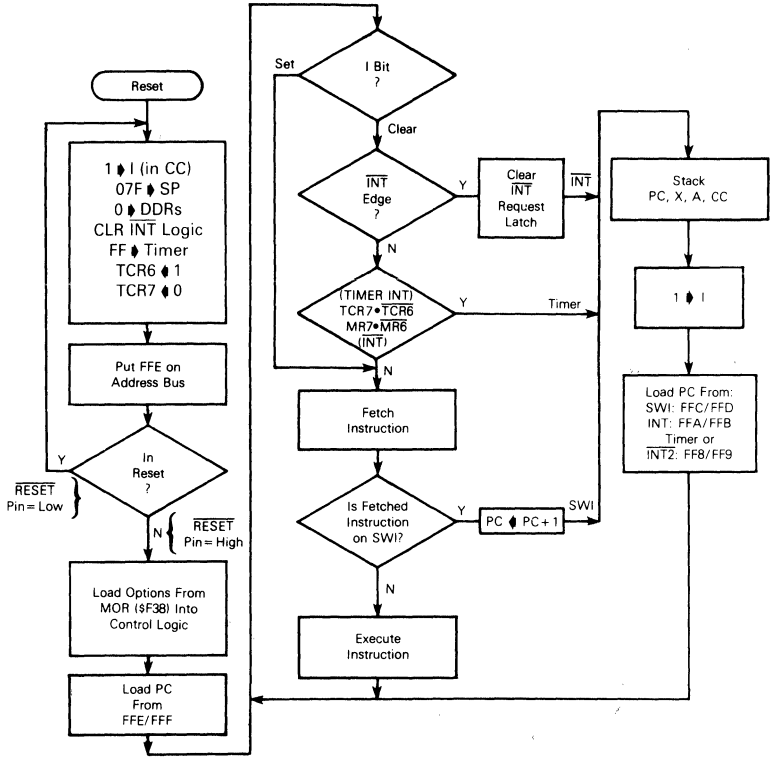


Figure 7. Reset and Interrupt Processing Flowchart

3

TIMER INTERRUPT

If the time mask bit (TCR6) is cleared, then, each time the timer decrements to zero (transitions from $01 to $00), an interrupt request is generated. The actual processor interrupt is generated only if the interrupt mask bit of the condition code register (CCR) is also cleared. When the interrupt is recognized, the current state of the machine is pushed onto the stack and the I bit in the CCR is set, masking further interrupts until the present one is serviced. The contents of the timer interrupt vector, containing the location of the timer interrupt service routine, is then loaded into the program counter. At the end of the timer interrupt service routine, the software normally executes an RTI instruction which restores the machine state and starts executing the interrupted program. The timer interrupt status bit can only be cleared by software.

EXTERNAL INTERRUPT

The external interrupt is internally synchronized and then latched on the falling edge of $\overline{INT}$ and $\overline{INT2}$. Clearing the I bit enables the external interrupt. The $\overline{INT2}$ interrupt has an interrupt request bit (bit 7) and a mask bit (bit 6) in the miscellaneous register (MR). The $\overline{INT2}$ interrupt is inhibited when the mask bit is set. The $\overline{INT2}$ is always read as a digital input on port D. The $\overline{INT2}$ and timer interrupt request bits, if set, cause the MCU to process an interrupt when the condition code I bit is clear. The following paragraphs describe two typical external interrupt circuits.

Zero-Crossing Interrupt

A sinusoidal input signal ($f_{\overline{INT}}$ maximum) can be used to generate an external interrupt (see Figure 8a) for use as a zero-crossing detector (for negative transitions of the ac sinusoid). This type of circuit allows applications such as servicing time-of-day routines and engaging disengaging ac power control devices. Off-chip, full-wave rectification provides an interrupt at every zero crossing of the ac signal and, thereby, provides a 2f clock.

Digital-Signal Interrupt

With this type of circuit (Figure 8b), the $\overline{INT}$ pin can be driven by a digital signal. The maximum frequency of a signal that can be recognized by the TIMER or $\overline{INT}$ pin logic is dependent on the parameter labeled t_{WL}, t_{WH}. Refer to **TIMER** for additional information.

SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. The SWI execution is similar to the hardware interrupts.

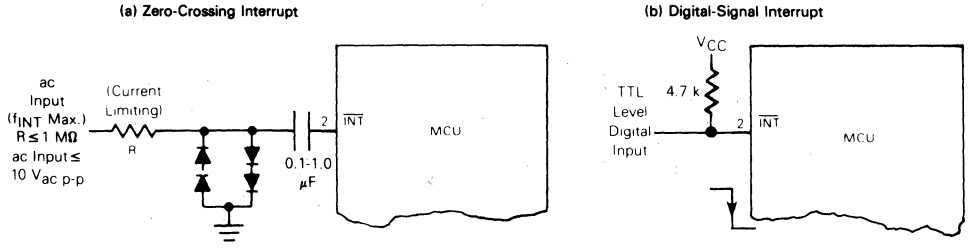


Figure 8. Typical Interrupt Circuits

MODES OF OPERATION

The MCU has two modes of operations. These modes are the normal and bootstrap. The following paragraphs describe the modes.

NORMAL MODE

This mode is a single-chip mode and is entered if the following conditions are met: (1) the $\overline{\text{RESET}}$ line is low, (2) the PC0 pin is within its normal operational range, and (3) the V_{PP} pin is connected to V_{CC}. The next rising edge of the $\overline{\text{RESET}}$ pin then causes the part to enter the normal mode.

BOOTSTRAP

The bootstrap mode is entered if the TIMER pin equals 12 V. Refer to application note, *MC68705P3/R3/U3 8-Bit EPROM Microcontroller Programming Module*, (AN-857 Rev.2).

TIMER

The MCU consists of an 8-bit software programmable counter driven by a 7-bit software programmable prescaler. The various timer sources are made via the timer control register (TCR) and/or the mask option register (MOR). The 8-bit counter may be loaded under program control and is decremented toward zero. When the timer reaches zero, the timer interrupt request bit (bit 7) in the timer control register (TCR) is set. Refer to Figure 9 for timer block diagram.

The timer interrupt can be masked (disabled) by setting the timer interrupt mask bit (bit 6) in the TCR. When the

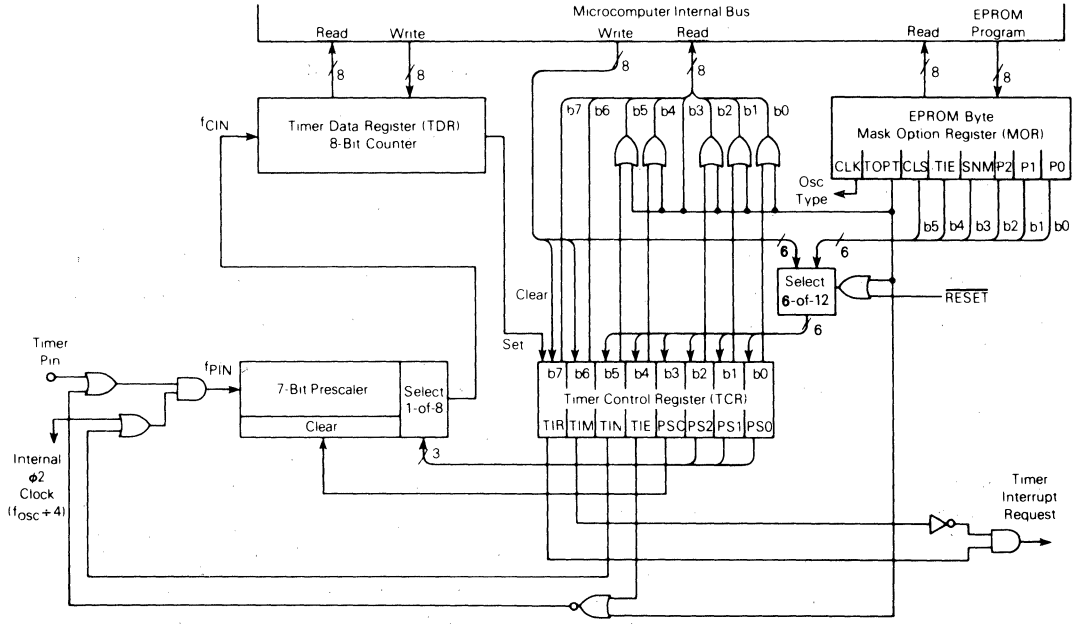


Figure 9. Timer Block Diagram

I bit in the condition code register is cleared and the TCR bit 6 is cleared, the processor receives the interrupt. The MCU responds to this interrupt by (1) saving the present CPU state on the stack, (2) fetching the timer interrupt vector, and (3) executing the interrupt routine. The timer interrupt request bit must be cleared by software. Refer to **RESETS** and **INTERRUPTS** for additional information.

The prescaler is a 7-bit divider which is used to extend the maximum length of the timer. To avoid truncation errors, the prescaler is cleared when TCR bit 3 is set to a logic 1; however, the TCR bit 3 always reads as a logic 0 to ensure proper operation with read-modify write instructions.

The timer continues to count past zero, falling from $00 through $FF, and continues the countdown. The counter can be read at any time by reading the timer data register (TDR). This allows a program to determine the length of time since a timer interrupt has occurred without disturbing the counting process. The TDR is unaffected by reset.

SOFTWARE CONTROLLED MODE

This mode is selected when TOPT (bit 6) in the MOR is programmed to zero. The timer prescaler input can be configured for three different operating modes plus a disable mode, depending on the value written to TCR control bits 4 and 5 (TIE and TIN). The following paragraphs describe the different modes.

Timer Input Mode 1

When TIE and TIN are both programmed to zero, the timer input is from the internal clock (phase two) and the timer input pin is disabled. The internal clock mode can be used for periodic interrupt generation as well as a reference for frequency and event measurement.

Timer Input Mode 2

When TIE = 1 and TIN = 0, the internal clock and the timer input signals are ANDed to form the timer input. This mode can be used to measure external pulse widths. The active high, external pulse gates in the internal clock for the duration of the external pulse. The accuracy of the count is ±1.

Timer Input Mode 3

When TIE = 0 and TIN = 1, no prescaler input frequency is applied to the prescaler and the timer is disabled.

Timer Input Mode 4

When TIE and TIN are both one, the timer input is from the external clock. The external clock can be used to count external events as well as to provide an external frequency for generating periodic interrupts. Frequency of external input must be $\leq f_{osc}/8$.

MOR CONTROLLED MODE

This mode is selected when TOPT (bit 6) in the MOR is programmed to logic one. The timer circuits are the same as described in **SOFTWARE CONTROLLED MODE**. The logic levels of TCR bits 0, 1, 2, and 5 are determined during EPROM programming by the same bits in the MOR. Therefore, bits 0, 1, 2, and 5 in the MOR control the prescaler division and the timer clock selection. TIE (bit 4) and PSC (bit 3) in the TCR are set to a logic one when in the MOR controlled mode. TIM (bit 6) and TIR (bit 7) are controlled by the counter and software.

TIMER CONTROL REGISTER (TCR) $009

This is an 8-bit register that controls various functions such as configuring operation mode, setting ratio of the prescaler, and generating timer interrupt request signal. All bits are read/write except bit 3. The configuration of the TCR is determined by the TOPT (bit 6) in the MOR. When TOPT = 1, the TCR emulates the MC6805R2; when TOPT = 0, the TCR is controlled by software.

TCR with MOR TOPT = 1

7	6	5	4	3	2	1	0
TIR	TIM	*	*	PSC	*	*	*

TCR with MOR TOPT = 0

7	6	5	4	3	2	1	0
TIR	TIM	TIN	TIE	PSC	PS2	PS1	PS0

RESET:							
0	1	U	U	U	U	U	U

*The value of corresponding bits in MOR is written during $\overline{\text{RESET}}$ rising edge. These bits always read "one".

TIR — Timer Interrupt Request
Used to indicate the timer interrupt when it is logic one
1 = Set when the timer data register changes to all zeros
0 = Cleared by external reset, power-on reset, or under program control

TIM — Timer Interrupt Mask
Used to inhibit the timer interrupt
1 = Interrupt inhibited
0 = Interrupt enabled

TIN — External or Internal
Selects input clock source
1 = External clock selected
0 = Internal clock selected (f_{osc} 4)

TIE — TIMER External Enable
Used to enable external TIMER pin. When TOPT 1, TIE is always a logical "one".
1 = Enables external timer pin
0 = Disables external timer pin

PSC — Prescaler Clear
Write only bit. Writing a one to this bit resets the prescaler to zero. A read of this location always indicates a zero when TOPT = 0. When TOPT = 1, this bit will read a logical "one1" and has no effect on the prescaler.

PS2, PS1, PS0 — Prescaler Select Bits
Decoded to select one of eight outputs of the prescaler

NOTES

When changing the PS bits in software, the PSC bit should be written to a "one" in the same write cycle to clear the prescaler. Changing the PS bits without clearing the prescaler may cause prescaler truncation.

PS2	PS1	PS0	Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

MASK OPTION REGISTER (MOR) $F38

The MOR is implemented in EPROM. This register contains all zeros prior to programming and is not affected by reset. The MOR bits are described in the following paragraphs.

7	6	5	4	3	2	1	0
CLK	TOPT	CLS		SNM	P2	P1	P0

CLK — Clock (oscillator type)
- 1 = Resistor Capacitor (RC)
- 0 = Crystal

TOPT — Timer Option
- 1 = MC6805R2 type timer/prescaler. All bits except 6 and 7, of the TCR are invisible to the user. Bits 5, 2, 1, and 0 of the MOR determine the equivalent MC6805R2 mask options.
- 0 = All TCR bits are implemented as a software programmable timer. The state of MOR bits 5, 4, 2, 1, and 0 sets the initial values of their respective TCR bits.

CLS — Timer/Prescaler Clock Source
- 1 = External TIMER pin
- 0 = Internal clock

Bit 4

Not used if TOPT = 1. Sets the initial value of TIE in the TCR if TOPT = 0.
- 1 = Not used
- 0 = Sets initial value of TIE in the TCR

SNM — Secure Mode
- 1 = EPROM contents cannot be access externally
- 0 = EPROM not programmed

P2, P1, P0 — The logical levels of these bits, when decoded, select one of eight outputs on the timer prescaler.

P2	P1	P0	Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

PROGRAMMING CONTROL REGISTER (PCR) $00B

The PCR is an 8-bit register which provides the necessary control bits to program the EPROM. Because the bootstrap program manipulates the PCR when programming, the user need not be concerned with PCR in most applications.

7	6	5	4	3	2	1	0
1	1	1	1	1	$\overline{VPON}$	$\overline{PGE}$	$\overline{PLE}$
RESET:							
U	U	U	U	U	U	1	1

$\overline{PLE}$ — Programming Latch Enable

Controls address and data being latched into the EPROM. Set during reset, but may be cleared anytime.
- 1 = Read EPROM
- 0 = Latch address and data on EPROM

$\overline{PGE}$ — Program Enable

Enables programming of EPROM. Must be set when changing the address and data. Set during reset.
- 1 = Inhibit EPROM programming
- 0 = Enable EPROM programming (if $\overline{PLE}$ is low)

$\overline{VPON}$ — V_{PP} On

A read-only bit that indicates high voltage at the V_{PP} pin. When set to "one", disconnects $\overline{PGE}$ and $\overline{PLE}$ from the chip.
- 1 = No high voltage on V_{PP} pin
- 0 = High voltage on V_{PP} pin

NOTE

$\overline{VPON}$ being "zero" does not indicate that the V_{PP} level is correct for programming. It is used as a safety interlock for the user in the normal operating mode.

$\overline{VPON}$	$\overline{PGE}$	$\overline{PLE}$	Programming Conditions
0	0	0	Programming mode (program EPROM byte)
1	0	0	$\overline{PGE}$ and $\overline{PLE}$ disabled from system
0	1	0	Programming disabled (latch address and data in EPROM)
1	1	0	$\overline{PGE}$ and $\overline{PLE}$ disabled from system
0	0	1	Invalid state; $\overline{PGE}$ = 0 if $\overline{PLE}$ = 0
1	0	1	Invalid state; $\overline{PGE}$ = 0 if $\overline{PLE}$ = 0
0	1	1	"High voltage" on V_{PP}
1	1	1	$\overline{PGE}$ and $\overline{PLE}$ disabled from system (operating mode)

EPROM PROGRAMMING

ERASING THE EPROM

The EPROM can be erased by exposure to high-intensity ultraviolet (UV) light with a wavelength of 2537 angstroms. The recommended integrated dose (UV intensity × exposure time) is 25Ws/cm^2. The lamps should be used without software filters, and the MCU should be positioned about one inch from the UV tubes. Ultraviolet erasure clears all bits of the MCU EPROM to the "0" state. Data then can be entered by programming "1s" into the desired bit locations.

PROGRAMMING

The MCU bootstrap program can be used to program the MCU EPROM. The alternate vectoring used to implement the self-check is used to start execution of the bootstrap program.

A MCM2532 UV EPROM (other industry standard EPROMs may be used) must first be programmed with the same information that is to be transferred to the MCU EPROM. The MC68705R5 is programmed the same as the MC68705R3. Refer to application note, *MC68705P3/R3/U3 8-Bit EPROM Microcontroller Programming Module* (AN-857 Rev.2) for schematic diagrams and instructions on programming the MCU EPROM.

EMULATION

The MC68705R5 emulates the MC6805R2 and MC6805R3 "exactly". The MC6805R2 and MC6805R3 mask features are implemented in the mask option register EPROM byte. The following list identifies a few minor exceptions to the exactness of the emulation.

1. The MC6805R2 "future ROM" areas are implemented in the MC68705R5, and these 1728 bytes must be left unprogrammed to accurately simulate the MC6805R2.
2. The reserved ROM areas have different data stored in them. In the MC6805R2, this area is used for self-check, and in the MC68705R5 this area is used for the bootstrap program.
3. The MC6805R2 reads all ones in the 48 byte "future RAM" area. This area is not implemented on the MC6805R2 mask ROM version but is implemented on the MC68705R5.
4. The MC68705R5 V_{PP} (pin 7) line is tied to V_{CC} during normal operations. On MC6805R2, this pin is grounded during normal operation; on MC6805R3, this pin is not connected.

ANALOG-TO-DIGITAL CONVERTER

The chip resident 8-bit analog-to-digital (A/D) converter uses a successive approximation technique as show in Figure 10. Four external analog inputs can be connected to the A/D via port D. Four internal analog channels (V_{RH} - V_{RL}, $V_{RH} - V_{RL}/2$, $V_{RH} - V_{RL}/4$, and V_{RL}) may be selected for calibration. The accuracy of these internal channels may not meet the accuracy specifications of the external channels.

Multiplexer selection is controlled by the A D control register (ACR) bits 0, 1, and 2. Refer to Table 2 for multiplexer selection. The ACR is shown in Figure 10. The converter uses 30 machine cycles to complete a conversion of a sampled analog input. When the conversion is complete, the digital value is placed in the A D result register (ARR); the conversion flag is set; selected input is sampled again; and a new conversion begins. When ACR7 is cleared, the conversion in progress is aborted and the selected input, which is held internally, is sampled for five machine cycles.

The converter uses V_{RH} and V_{RL} as reference voltages. An input voltage equal to or greater than V_{RH} converts to $FF. An input voltage equal to or less than V_{RL}, but greater than V_{SS}, converts to $00. Maximum and minimum ratings must not be exceeded. Each analog input source should use V_{RH} as the supply voltage and should be referenced to V_{RL} for the ratiometric conversion. To maintain full accuracy of the A D, three requirements should be followed: (1) V_{RH} should be equal to or less than V_{CC}, (2) V_{RL} should be equal to or greater than V_{SS} but less than maximum specifications, and (3) V_{RH} V_{RL} should be equal to or greater than 4 volts.

The A D has a built-in 1 2 LSB offset intended to reduce the magnitude of the quantizing error to · 1 2 LSB rather than +0, - 1 LSB with no offset. This implies that, ignoring errors, the transition point from $00 to $01 occurs

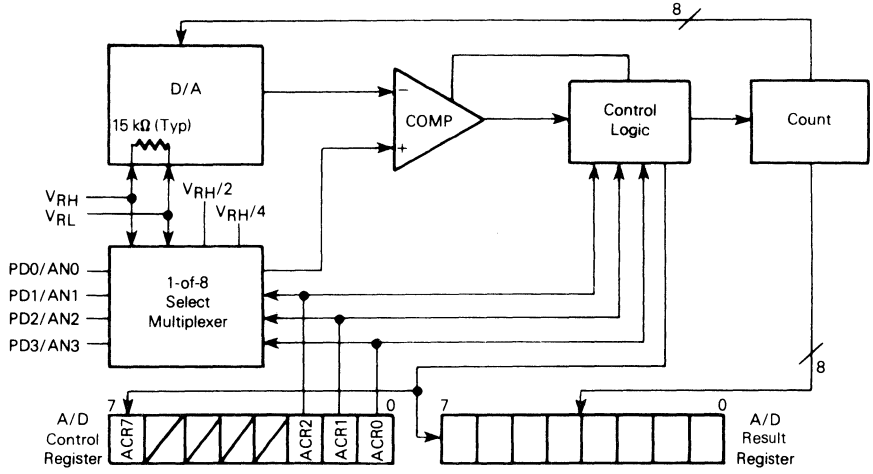


Figure 10. A/D Block Diagram

Table 2. A/D Input MUX Selection

A/D Control Register			Input Selected	A/D Output (Hex)		
ACR2	ACR1	ACR0		Min	Typ	Max
0	0	0	AN0			
0	0	1	AN1			
0	1	0	AN2			
0	1	1	AN3			
1	0	0	V_{RH}*	FE	FF	FF
1	0	1	V_{RL}*	00	00	01
1	1	0	$V_{RH/4}$*	3F	40	41
1	1	1	$V_{RH/2}$*	7F	80	81

*Internal (calibration) levels

at 1 2 LSB above V_{RL}. Similarly, the transition from $FE to $FF occurs 1-1/2 LSB below V_{RH}, ideally.

INSTRUCTION SET

3

The MCU has a set of 59 basic instructions which can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following instruction list.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (2's Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Higher	BHI
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
(Branch if Higher or Same)	(BHS)
Branch if Carry Set	BCS
(Branch if Lower)	(BLO)
Branch if Not Equal	BNE
Branch if Equal	BEQ
Branch if Half Carry Clear	BHCC
Branch if Half Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Bit is Clear	BMC
Branch if Interrupt Mask Bit is Set	BMS
Branch if Interrupt Line is Low	BIL
Branch if Interrupt Line is High	BIH
Branch to Subroutine	BSR

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the first 256 bytes of the memory space where

all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear and bit test, and branch functions are all implemented with a single instruction. For test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch if Bit n is Set	BRSET n (n = 0 . . . 7)
Branch if Bit n is Clear	BRCLR n (n = 0 . . . 7)
Set Bit n	BSET n (n = 0 . . . 7)
Clear Bit n	BCLR n (n = 0 . . . 7)

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No-Operation	NOP

OPCODE MAP SUMMARY

Table 3 is an opcode map for the instructions used on the MCU.

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single byte instructions, while the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. Two-byte direct addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction.

RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from − 126 to + 129 from the opcode address.

INDEXED, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. This addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this two-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

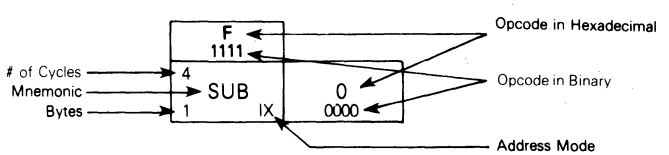
Table 3. Opcode Map

Hi / Low	Bit Manipulation BTB 0 0000	Bit Manipulation BSC 1 0001	Branch REL 2 0010	Read-Modify-Write DIR 3 0011	Read-Modify-Write INH 4 0100	Read-Modify-Write INH 5 0101	Read-Modify-Write IX1 6 0110	Read-Modify-Write IX 7 0111	Control INH 8 1000	Control INH 9 1001	Register/Memory IMM A 1010	Register/Memory DIR B 1011	Register/Memory EXT C 1100	Register/Memory IX2 D 1101	Register/Memory IX1 E 1110	Register/Memory IX F 1111	Hi / Low
0 0000	10 BRSET0 3 BTB	7 BSET0 2 BSC	4 BRA 2 REL	6 NEG 2 DIR	4 NEG 1 INH	4 NEG 1 INH	7 NEG 2 IX1	6 NEG 1 IX	9 RTI 1 INH		2 SUB 2 IMM	4 SUB 2 DIR	5 SUB 3 EXT	6 SUB 3 IX2	5 SUB 2 IX1	4 SUB 1 IX	0 0000
1 0001	10 BRCLR0 3 BTB	7 BCLR0 2 BSC	4 BRN 2 REL						6 RTS 1 INH		2 CMP 2 IMM	4 CMP 2 DIR	5 CMP 3 EXT	6 CMP 3 IX2	5 CMP 2 IX1	4 CMP 1 IX	1 0001
2 0010	10 BRSET1 3 BTB	7 BSET1 2 BSC	4 BHI 2 REL								2 SBC 2 IMM	4 SBC 2 DIR	5 SBC 3 EXT	6 SBC 3 IX2	5 SBC 2 IX1	4 SBC 1 IX	2 0010
3 0011	10 BRCLR1 3 BTB	7 BCLR1 2 BSC	4 BLS 2 REL	6 COM 2 DIR	4 COMA 1 INH	4 COMX 1 INH	7 COM 2 IX1	6 COM 1 IX	11 SWI 1 INH		2 CPX 2 IMM	4 CPX 2 DIR	5 CPX 3 EXT	6 CPX 3 IX2	5 CPX 2 IX1	4 CPX 1 IX	3 0011
4 0100	10 BRSET2 3 BTB	7 BSET2 2 BSC	4 BCC 2 REL	6 LSR 2 DIR	4 LSRA 1 INH	4 LSRX 1 INH	7 LSR 2 IX1	6 LSR 1 IX			2 AND 2 IMM	4 AND 2 DIR	5 AND 3 EXT	6 AND 3 IX2	5 AND 2 IX1	4 AND 1 IX	4 0100
5 0101	10 BRCLR2 3 BTB	7 BCLR2 2 BSC	4 BCS 2 REL								2 BIT 2 IMM	4 BIT 2 DIR	5 BIT 3 EXT	6 BIT 3 IX2	5 BIT 2 IX1	4 BIT 1 IX	5 0101
6 0110	10 BRSET3 3 BTB	7 BSET3 2 BSC	4 BNE 2 REL	6 ROR 2 DIR	4 RORA 1 INH	4 RORX 1 INH	7 ROR 2 IX1	6 ROR 1 IX			2 LDA 2 IMM	4 LDA 2 DIR	5 LDA 3 EXT	6 LDA 3 IX2	5 LDA 2 IX1	4 LDA 1 IX	6 0110
7 0111	10 BRCLR3 3 BTB	7 BCLR3 2 BSC	4 BEQ 2 REL	6 ASR 2 DIR	4 ASRA 1 INH	4 ASRX 1 INH	7 ASR 2 IX1	6 ASR 1 IX		2 TAX 1 INH		5 STA 2 DIR	6 STA 3 EXT	7 STA 3 IX2	6 STA 2 IX1	5 STA 1 IX	7 0111
8 1000	10 BRSET4 3 BTB	7 BSET4 2 BSC	4 BHCC 2 REL	6 LSL 2 DIR	4 LSLA 1 INH	4 LSLX 1 INH	7 LSL 2 IX1	6 LSL 1 IX		2 CLC 1 INH	2 EOR 2 IMM	4 EOR 2 DIR	5 EOR 3 EXT	6 EOR 3 IX2	5 EOR 2 IX1	4 EOR 1 IX	8 1000
9 1001	10 BRCLR4 3 BTB	7 BCLR4 2 BSC	4 BHCS 2 REL	6 ROL 2 DIR	4 ROLA 1 INH	4 ROLX 1 INH	7 ROL 2 IX1	6 ROL 1 IX		2 SEC 1 INH	2 ADC 2 IMM	4 ADC 2 DIR	5 ADC 3 EXT	6 ADC 3 IX2	5 ADC 2 IX1	4 ADC 1 IX	9 1001
A 1010	10 BRSET5 3 BTB	7 BSET5 2 BSC	4 BPL 2 REL	6 DEC 2 DIR	4 DECA 1 INH	4 DECX 1 INH	7 DEC 2 IX1	6 DEC 1 IX		2 CLI 1 INH	2 ORA 2 IMM	4 ORA 2 DIR	5 ORA 3 EXT	6 ORA 3 IX2	5 ORA 2 IX1	4 ORA 1 IX	A 1010
B 1011	10 BRCLR5 3 BTB	7 BCLR5 2 BSC	4 BMI 2 REL							2 SEI 1 INH	2 ADD 2 IMM	4 ADD 2 DIR	5 ADD 3 EXT	6 ADD 3 IX2	5 ADD 2 IX1	4 ADD 1 IX	B 1011
C 1100	10 BRSET6 3 BTB	7 BSET6 2 BSC	4 BMC 2 REL	6 INC 2 DIR	4 INCA 1 INH	4 INCX 1 INH	7 INC 2 IX1	6 INC 1 IX		2 RSP 1 INH		3 JMP 2 DIR	4 JMP 3 EXT	5 JMP 3 IX2	4 JMP 2 IX1	3 JMP 1 IX	C 1100
D 1101	10 BRCLR6 3 BTB	7 BCLR6 2 BSC	4 BMS 2 REL	6 TST 2 DIR	4 TSTA 1 INH	4 TSTX 1 INH	7 TST 2 IX1	6 TST 1 IX		2 NOP 1 INH	8 BSR 2 REL	7 JSR 2 DIR	8 JSR 3 EXT	9 JSR 3 IX2	8 JSR 2 IX1	7 JSR 1 IX	D 1101
E 1110	10 BRSET7 3 BTB	7 BSET7 2 BSC	4 BIL 2 REL								2 LDX 2 IMM	4 LDX 2 DIR	5 LDX 3 EXT	6 LDX 3 IX2	5 LDX 2 IX1	4 LDX 1 IX	E 1110
F 1111	10 BRCLR7 3 BTB	7 BCLR7 2 BSC	4 BIH 2 REL	6 CLR 2 DIR	4 CLRA 1 INH	4 CLRX 1 INH	7 CLR 2 IX1	6 CLR 1 IX		2 TXA 1 INH		5 STX 2 DIR	6 STX 3 EXT	7 STX 3 IX2	6 STX 2 IX1	5 STX 1 IX	F 1111

Abbreviations for Address Modes

INH	Inherent
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This addressing mode can be used in a manner similar to indexed, 8-bit offset except that this three-byte instruction allows tables to be anywhere in memory.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte to which the specified bit is to be set or cleared. Any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single two-byte instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit to be tested, and its condition (set or clear), is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from − 125 to + 130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

3

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	− 0.3 to + 7.0	V
Input Voltage EPROM Programming Voltage (V_{PP} Pin) TIMER Pin — Normal Mode TIMER Pin — Bootstrap Programming Mode All Others	 V_{PP} V_{in} V_{in} V_{in}	 − 0.3 to + 22.0 − 0.3 to + 7.0 − 0.3 to + 15.0 − 0.3 to + 7.0	V
Operating Temperature Range MC68705R5 MC68705R5C	T_A	T_L to T_H 0 to + 70 − 40 to + 85	C
Storage Temperature Range	T_{stg}	− 55 to + 150	C
Junction Temperature Plastic Cerdip	T_J	 150 175	C W

These devices contain circuitry to protect the inputs against damage due to high static voltages or electrical fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high-impedance circuit. For proper operation, it is recommended that V_{in} and V_{out} be constrained to the range V_{SS} ≤ (V_{in} and V_{out}) ≤ V_{CC}. Reliability of operation is enhanced if unused inputs except EXTAL are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic (P Suffix) Plastic (FN Suffix) Cerdip (S Suffix)	θ_{JA}	 50 100 60	C W

POWER CONSIDERATION

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A

ELECTRICAL CHARACTERISTICS

(V_{CC} = 5.25 Vdc ± 0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage	V_{IH}				V
RESET (4.75 < V_{CC} < 5.75)		4.0	—	V_{CC}	
(V_{CC} < 4.75)		V_{CC} − 0.5	—	V_{CC}	
INT 4.75 < V_{CC} < 5.75)		4.0	**	V_{CC}	
(V_{CC} < 4.75)		V_{CC} − 0.5	**	V_{CC}	
All Other		2.0	—	V_{CC}	
Input High Voltage (TIMER Pin)	V_{IH}				V
Timer Mode		2.0	—	V_{CC} + 1.0	
Bootstrap Programming Mode		9.0	12.0	15.0	
Input Low Voltage	V_{IL}				V
RESET		V_{SS}	—	0.8	
INT		V_{SS}	**	1.5	
All Other		V_{SS}	—	0.8	
INT Zero-Crossing Input Voltage — Through a Capacitor	V_{INT}	2.0	—	4.0	$V_{ac\ p\text{-}p}$
Internal Power Dissipation (No Port Loading, V_{CC} = 5.25 V T_A = 0°C	P_{INT}	—	520	740	mW
for Steady-State Operation) T_A = 40°C		—	580	800	
Input Capacitance	C_{in}				pF
EXTAL		—	25	—	
All Other (See Note)		—	10	—	
RESET Hysteresis Voltage					V
Out of Reset Voltage	V_{IRES+}	2.1	—	4.0	
Into Reset Voltage	V_{IRES-}	0.8	—	2.0	
Programming Voltage (V_{PP} Pin)	V_{PP}*				V
Programming EPROM		20.0	21.0	22.0	
Operating Voltage		4.75	V_{CC}	5.75	
Input Current					μA
TIMER (V_{in} = 0.4 V)	I_{in}	—	—	20	
INT (V_{in} = 0.4 V)		—	20	50	
EXTAL (V_{in} = 2.4 V to V_{CC})		—	—	10	
(V_{in} = 0.4 V)		—	—	−1600	
RESET (V_{in} = 0.8 V)	I_{RES}	−4.0	—	−40	
(External Capacitor Changing Current)					

*V_{PP} (pin 7) is connected to V_{CC} in the normal operating mode.
**Due to internal biasing, this input (when not used) floats to approximately 2.0 V.

NOTE: Port D analog inputs, when selected, C_{in} = 25 pF for the first 5 out of 30 cycles.

PROGRAMMING OPERATION ELECTRICAL CHARACTERISTICS

(V_{CC} = 5.25 Vdc + .05%, V_{SS} = 0, T_A = 20°C to 30°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Programming Voltage	V_{PP}	20.0	21.0	22.0	V
V_{PP} Supply Current V_{PP} 5.25 V V_{PP} 21.0 V	I_{PP}	 — —	 — —	 8 30	mA
Oscillator Frequency	$f_{osc(p)}$	0.9	1.0	1.1	MHz
Bootstrap Programming Mode Voltage (TIMER Pin) (@ I_{IHTP} 100 µA Maximum	V_{IHTP}	9.0	12.0	15.0	V

SWITCHING CHARACTERISTICS

(V_{CC} 5.25 Vdc 0.5 Vdc, V_{SS} 0 Vdc, T_A 0 C to 70 C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Oscillator Frequency Normal	f_{osc}	 0.4	 —	 4.2	MHz
Instruction Cycle Time (4 f_{osc})	t_{cyc}	0.950	—	10	µs
$\overline{INT}$, $\overline{INT2}$, or Timer Pulse Width	t_{WL}, t_{WH}	t_{cyc} + 250	—	—	ns
$\overline{RESET}$ Pulse Width	t_{RWL}	t_{cyc} + 250	—	—	ns
$\overline{RESET}$ Delay Time (External Cap 1.0 µF)	t_{RHL}	—	100	—	ms
$\overline{INT}$ Zero Crossing Detection Input Frequency	f_{INT}	0.03	—	1.0	kHz
External Clock Duty Cycle (EXTAL)	—	40	50	60	%
Crystal Oscillator Start-Up Time	—	—	—	100	ms

A/D CONVERTER CHARACTERISTICS

(V_{CC} = 5.25 V 0.5 Vdc, V_{SS} 0 Vdc, T_A 0 C to 70 C, unless otherwise noted)

Characteristic	Min	Typ	Max	Unit	Comments
Resolution	8	8	8	Bits	
Non-Linearity	—	—	1 2	LSB	For V_{RH} = 4.0 to 5.0 V and V_{RL} = 0 V.
Quantizing Error	—	—	1 2	LSB	
Conversion Range	V_{RL}	—	V_{RH}	V	
V_{RH}	—	—	V_{CC}	V	A D accuracy may decrease proportionately as
V_{RL}	V_{SS}	—	0.2	V	V_{RH} is reduced be below 4.0 V. The sum of V_{RH} and V_{RL} must not exceed V_{CC}.
Conversion Time	30	30	30	t_{cyc}	Includes sampling time
Monotonicity	Inherent (within total error)				
Zero Input Reading	00	00	01	hexadecimal	V_{in} = 0
Ratiometric Reading	FE	FF	FF	hexadecimal	V_{in} = V_{RH}
Sample Time	5	5	5	t_{cyc}	
Sample/Hold Capacitance, Input	—	—	25	pF	
Analog Input Voltage	V_{RL}	—	V_{RH}	V	Negative transients on any analog lines (pins 19-24) are not allowed at any time during conversion.

3

PORT ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0° to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Port A					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Output High Voltage, I_{Load} = −10 μA	V_{OH}	V_{CC} − 1.0	—	—	V
Input High Voltage, I_{Load} = −300 μA (Max)	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage, I_{Load} = −500 μA (Max)	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current (V_{in} = 2.0 V to V_{CC})	I_{IH}	—	—	−300	μA
Hi-Z State Input Current (V_{in} = 0.4 V)	I_{IL}	—	—	−500	μA
Port B					
Output Low Voltage, I_{Load} = 3.2 mA	V_{OL}	—	—	0.4	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.0	V
Output High Voltage, I_{Load} = −200 μA	V_{OH}	2.4	—	—	V
Darlington Current Drive (Source), V_O = 1.5 V	I_{OH}	−1.0	—	−10	mA
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port C					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port D (Input Only)					
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Input Current	I_{in}	—	<1	5	μA

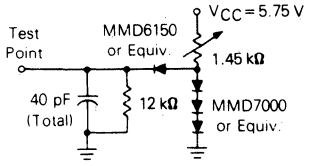


Figure 11. TTL Equivalent Test Load (Port B)

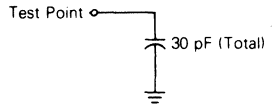


Figure 12. CMOS Equivalent Test Load (Port A)

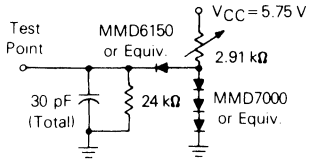


Figure 13. TTL Equivalent Test Load
(Ports A and C)

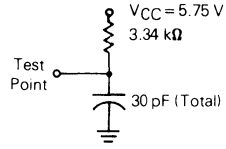


Figure 14. Open-Drain Equivalent Test Load
(Port C)

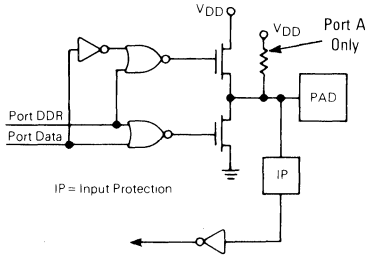


Figure 15. Ports A and C Logic Diagram

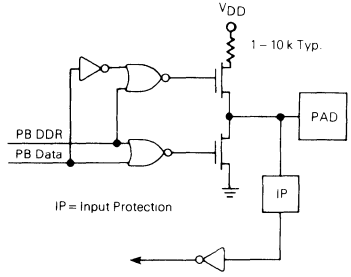


Figure 16. Port B Logic Diagram

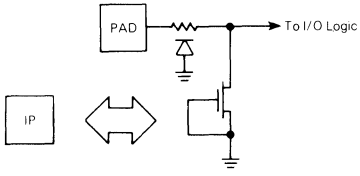


Figure 17. Typical Input Protection

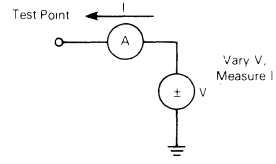


Figure18. I/O Characteristic
Measurement Circuit

MECHANICAL DATA

ORDERING INFORMATION

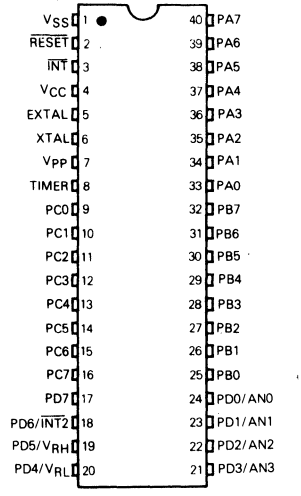
The following table provides generic information pertaining to the package type, temperature, and MC order numbers for the MC68705R5.

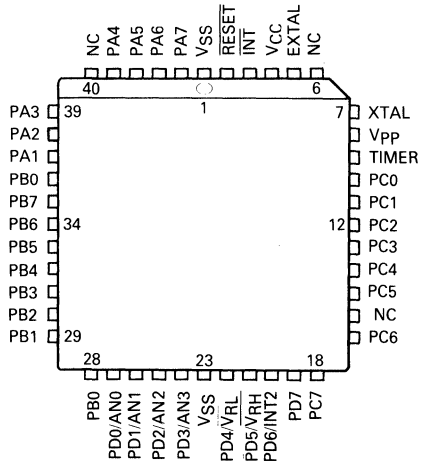
Table 4. Generic Information

Package Type	Temperature	Order Number
Cerdip S Suffix	0°C to 70°C −40°C to +85°C	MC68705R5S MC68705R5CS
Plastic P Suffix	0°C to 70°C −40°C to 85°C	MC68705R5P MC68705R5CP
PLCC FN Suffix	−40°C to 85°C	MC68705R5CFN

3

PIN ASSIGNMENTS





MC68705S3

Technical Summary
8-Bit EPROM Microcontroller Unit

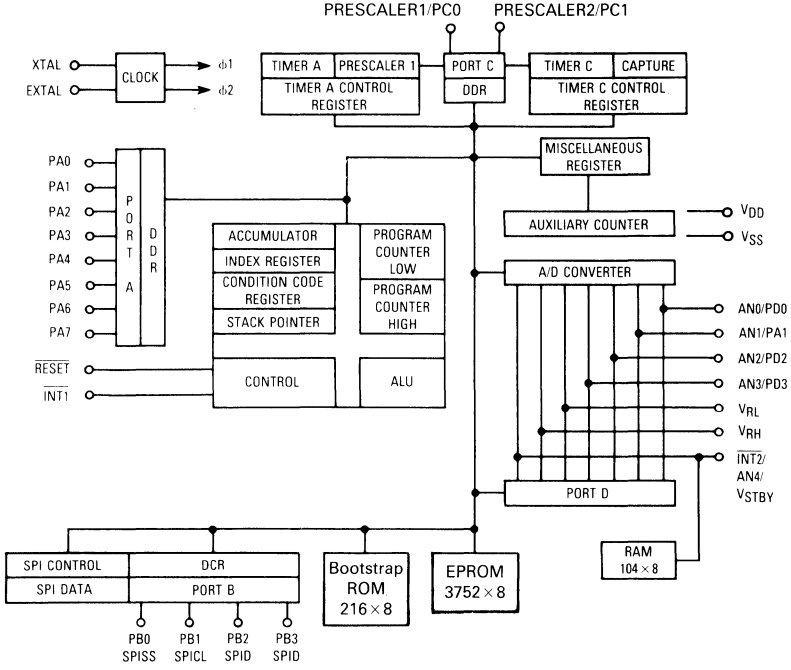
The MC68705 (HMOS) Microcontroller Unit (MCU)is an EPROM member of the MC6805 Family of microcontrollers. The user programmable EPROM allows program changes and lower volume applications. This high performance MCU has parallel I/O capability with pins programmable as input or output. This publication contains condensed information on the MCU; for detailed information, refer to *Advance Information 8-Bit Microcontroller* (ADI997R1) or contact your local Motorola sales office.

Refer to the block diagram for the hardware features and to the list below for additional features available on the MCU.

- One 7-Bit and One 15-Bit Software Programmable Prescaler
- On-chip Oscillator
- Memory Mapped I/O
- Versatile Interrupt Handling
- Bit Manipulation
- Bit Test and Branch Instruction
- Vectored Interrupts
- Bootstrap Program in ROM
- 3752 Bytes of EPROM
- 104 Bytes of RAM
- Serial Peripheral Interface
- Two 8-Bit and One 16-Bit Timers
- A/D Converter
- EPROM Read Inhibit Security Bit

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

V_{CC} and V_{SS}

Power is supplied to the microcontroller using these two pins. V_{CC} is +5.25 volts (±0.5Δ) power, and V_{SS} is ground.

NUM

This pin is for factory use only. It should be connected to V_{SS}.

$\overline{INT1}$, $\overline{INT2}$

These pins provide the capability for asynchronously applying an external interrupt to the MCU. Refer to **INTERRUPTS** for more detailed information.

XTAL, EXTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a ceramic resonator, a resistor/capacitor combination, or an external signal (depending on setting of the Mask Option Register) is connected to these pins to provide a system clock.

RC Oscillator

With this option, a resistor/capacitor combination is connected to the oscillator pins as shown in Figure 1(c). Refer to Figure 2 for the relationship between R and f_{osc}.

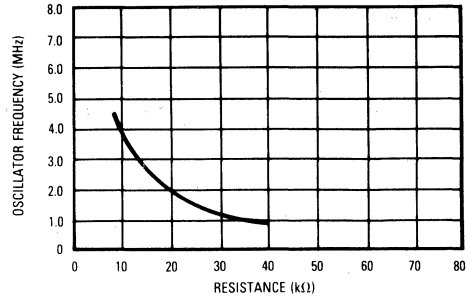


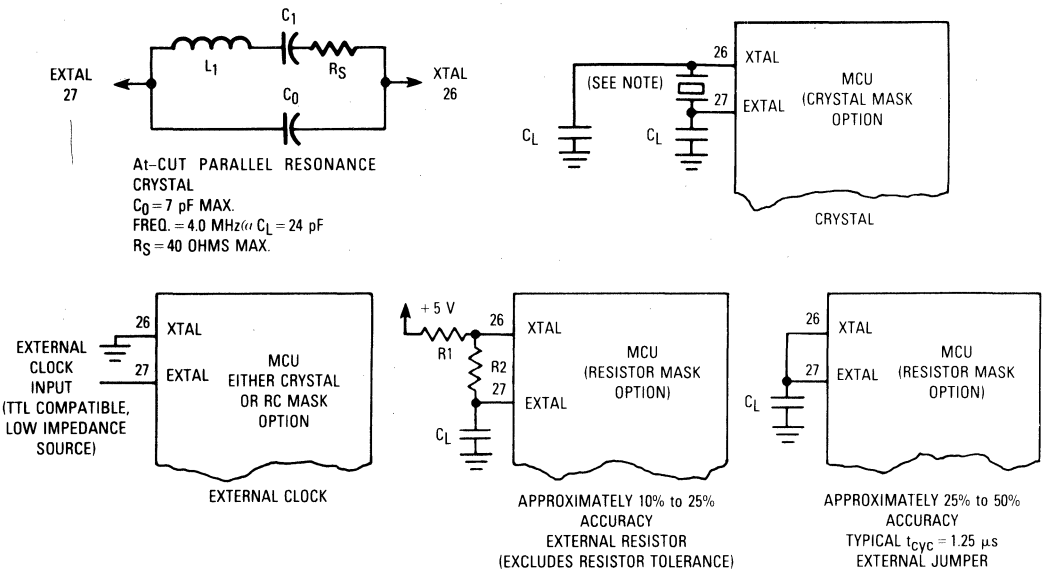
Figure 2. Typical Frequency vs Resistance for RC Oscillator Option Only

Crystal

The circuit shown in Figure 1(b) is recommended when using a crystal. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time.

External Clock

An external clock should be applied to the EXTAL input with the XTAL input grounded, as shown in Figure 1(d).



NOTE: The recommended C_L value with a 4.0 MHz crystal is 27 pF, maximum, including system distributed capacitance. For crystal frequencies other than 4 MHz, the total capacitance on each pin should be scaled as the inverse of the frequency ratio. For example, with 2 MHz crystal, use approximately 50 pF on EXTAL and approximately 50 pF on XTAL. The exact value depends on the motional-arm parametes of the crystal used.

Figure 1. Oscillator Connections

This option may only be used with the crystal oscillator option selected in the mask option register.

PC0, PC1

This pins allow an external input to decrement the internal timer/counter circuitry. Refer to **TIMERS** for additional information.

$\overline{\text{RESET}}$/V_{PP}

This pin has a Schmitt trigger input. The MCU can be reset by pulling $\overline{\text{RESET}}$ low. The V_{PP} input is used to input the programming voltage to the MCU EPROM. A 1K ohm pullup resistor should be used to allow proper operation of the reset and watchdog timer operations.

INPUT/OUTPUT LINES (PA0-PA7, PB0-PB3, PC0-PC1, PD0-PD6)

Ports A, B, and C are programmable as either inputs or outputs under software control of the data direction registers. Port D is a fixed input port and not controlled by any data direction register. Port D has up to five analog inputs, plus two voltage reference inputs when the analog-to-digital (A/D) converter is used (PD5/V_{RH}, PD4/V_{RL}) and an $\overline{\text{INT2}}$ input. If the analog input is used, the voltage reference pins (PD5/V_{RH} and PD4/V_{RL}) must be used in the analog mode. Refer to **INPUT/OUTPUT PORTS** for additional information.

INPUT/OUTPUT PORTS

INPUT/OUTPUT PROGRAMMING

Ports A, B, and C are programmable as either input or output under software control of the corresponding data direction register (DDR). The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic one for output and a logic zero for input. On reset, all DDRs are initialized to a logic zero state to put the ports in the input mode. The port output registers are not initialized on reset and should be written to before setting the DDR bits.

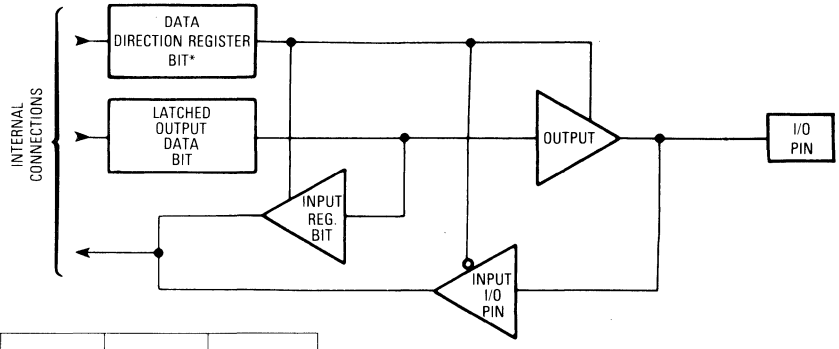
Port D provides the multiplexed analog inputs, reference voltages, and $\overline{\text{INT2}}$. These lines are shared with the port D digital inputs. PD0–PD3 may always be used as digital or analog inputs. The V_{RL} and V_{RH} lines are internally connected to the A/D resistor. Analog inputs may be prescaled to obtain the V_{RL} and V_{RH} recommended input voltage range.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This port write may be used to initialize the data registers and avoid undefined outputs. Care must be exercised when using read-modify-write instructions since the data read corresponds to the pin level if the DDR is an input (zero) and, also, to the latched output when the DDR is an output (one). Refer to Table 1 for I/O functions and to Figure 3 for typical port circuitry.

PORT B TOGGLE CAPABILITY

Port B0 and B1 registers have toggle capability at the timer underflow times. Under the control of the timer output cross-couple bit in the miscellaneous register (MR0), the overflow pulses from timer A, B, and C are directed to port B0 and B1 data registers. See Figure 4 for port B configuration flow chart.

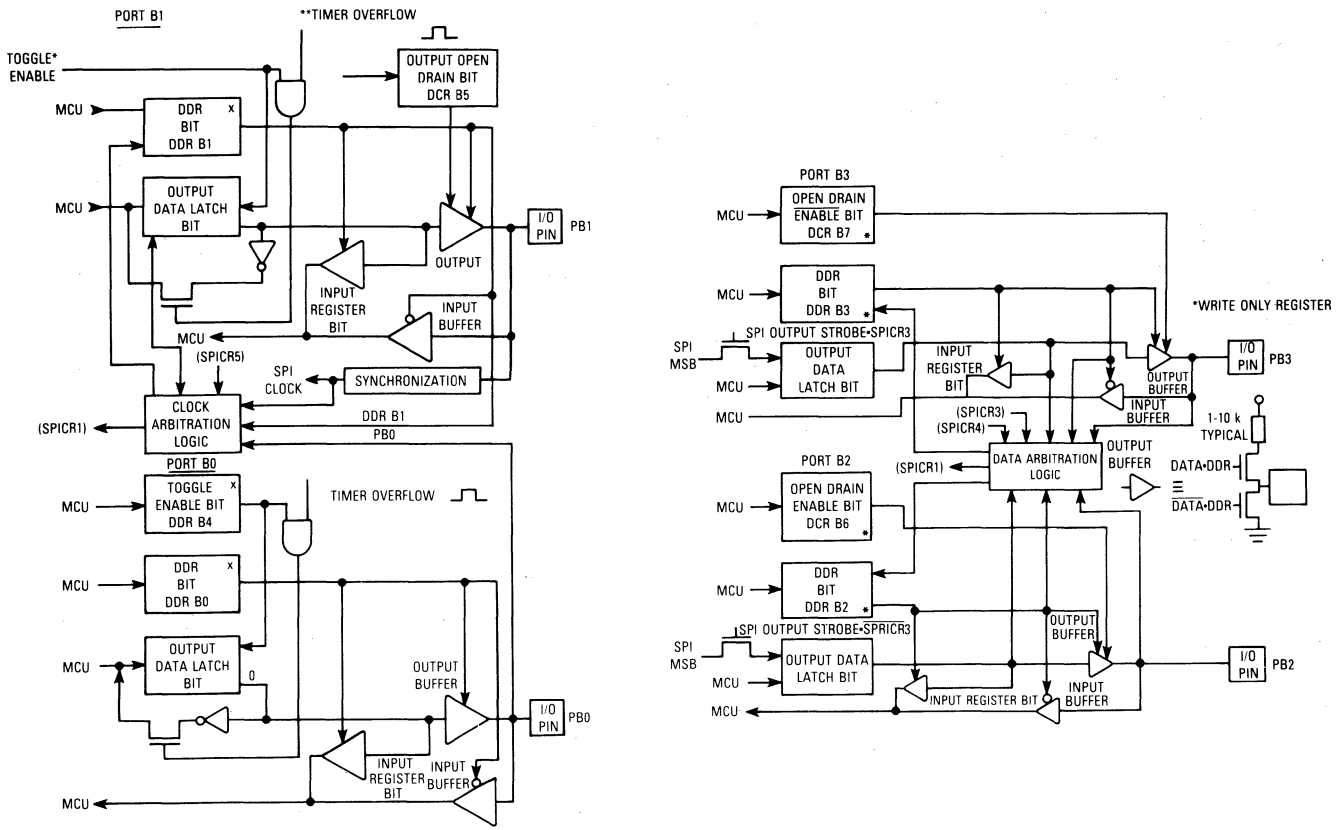
An incoming toggle pulse on port B0 is allowed to toggle the data register if port B DCR bit 4 (DCR4) is



Data Direction Register Bit	Output Data Bit	Output State	Input To MCU
1	0	0	0
1	1	1	1
0	X	High-Z**	Pin

*DDR is a write-only register and reads as all "1s".
**Ports A, B, and C are three-state ports. Port A has optional internal pullup devices to provide CMOS drive capability. See Electrical Characteristics tables for complete information.

Figure 3. Typical Port I/O Circuitry and Register Configuration



*Toggle Enable B1 = $(\overline{SPICR7}\cdot SPICR4\cdot(\overline{PB0} + DDRB0))\cdot SPICR2\cdot\overline{SPICR4}\cdot\overline{CLAQ}$
**A or B or C Depends on (MR0) and MOR5
*Write Only Register

Figure 4. Port B Configuration

cleared. This bit is set on reset. An incoming toggle pulse on port B1 is allowed to toggle the port B1 data register under the following conditions governed by control bits in SPI control register and SPI clock arbitration flip-flop status.

PB1 toggle enable = $(\overline{SPICR7})$•SPICR4• $(\overline{PB0} + DDRB0)$ + SPICR2• $\overline{SPICR4}$•$\overline{CLAQ}$

where: SPICR7 = SPI interrupt request bit
SPICR4 = SPI operation enable bit
SPICR2 = port B1 toggle enable/start bit
CLAQ = clock arbitration flip-flop output

When PB1 toggle enable is asserted, the MCU write to PB1 data register is inhibited. When SPI is not used, SPICR4 and CLAQ are reset. Therefore, SPICR2 can directly control the port B1 toggle capability. Port toggle capability allows action on port B0 or B1 or both as a result of timer overflows. This method speeds up timer overflow to port service. A write to port B0 or B1 data registers is inhibited while the individual port toggle enable is asserted.

The port B DCR consists of four status bits (DCR4-DCR7) and four data direction bits (DCR0-DCR3). DCR4 is a toggle enable control bit for port B0. When cleared, the timer overflow pulse causes the data register on port B0 to toggle. Port A has an 8-bit and port C has a 2-bit wide data direction register.

MEMORY

The MCU is capable of addressing 4096 bytes of memory and I/O registers. The memory map is shown in Figure 5. The locations consist of user EPROM, bootstrap ROM, user RAM, eight timer registers, a mask option register (MOR), a miscellaneous register, a program control register, two A/D registers, two SPI registers, and four I/O port registers. The interrupt vectors are located from $FF8 to $FFF. The bootstrap is a mask-programmed ROM that allows the MCU to program its own EPROM.

The stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer decrements during pushes and increments during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

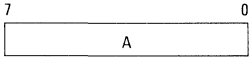
Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

REGISTERS

The MCU contains the registers described in the following paragraphs.

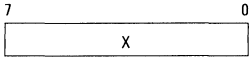
ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



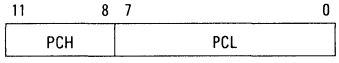
INDEX REGISTER (X)

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16-bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



PROGRAM COUNTER (PC)

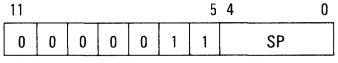
The program counter is a 12-bit register that contains the address of the next byte to be fetched.



STACK POINTER (SP)

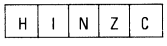
The stack pointer is a 12-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set at location $07F. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

The seven most-significant bits of the stack pointer are permanently set at 0000011. Subroutines and interrupts may be nested down to location $061 (31 bytes maximum), which allows the programmer to use up to 15 levels of subroutine calls (less if interrupts are allowed).



CONDITION CODE REGISTER (CC)

The condition code register is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.



Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, the timer (A, B, and C), the external ($\overline{INT1}$ and $\overline{INT2}$) interrupts, and the SPI interrupt are masked (disabled). If an interrupt occurs while this bit is set, the interrupt is latched and is processed as soon as the interrupt bit is cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic one).

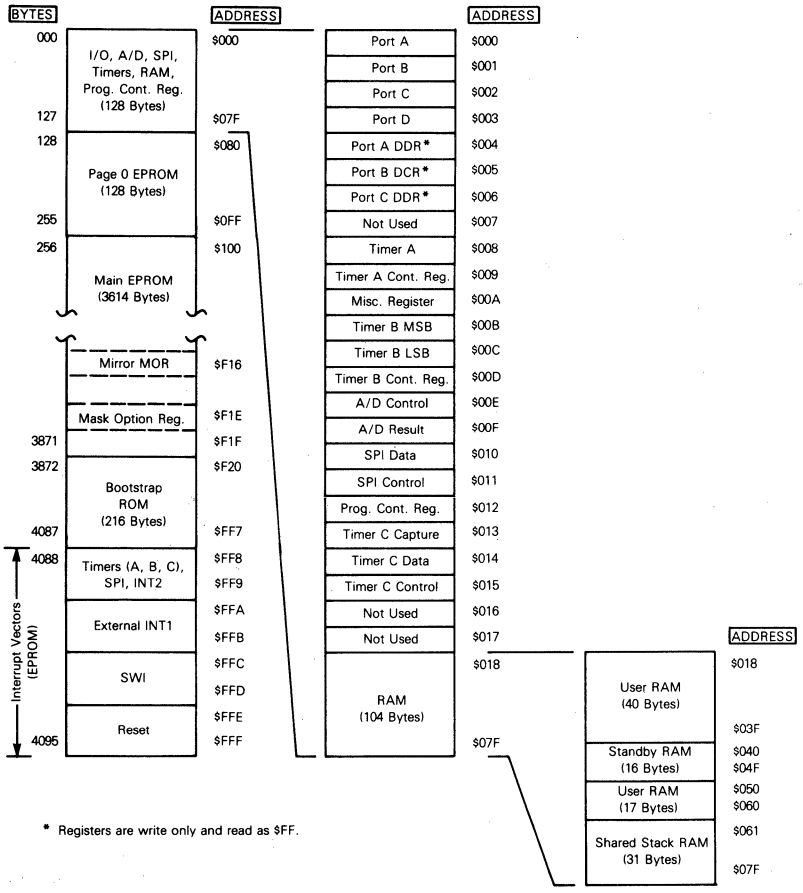


Figure 5. Memory Map

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/Borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions, and during shifts and rotates.

MISCELLANEOUS REGISTERS (MR) $0A

This register contains control and status information related to $\overline{INT2}$, auxiliary counter, prescalers 1 and 2, and timer overflow.

MR7	MR6	MR5	MR4	MR3	MR2	MR1	MR0
RESET:							
0	1	0	1			0	0

MR7 — $\overline{INT2}$ Interrupt Request Bit
If not masked by MR6, it causes an interrupt to the MCU; if the I bit in the CCR is clear, the MCU will acknowledge the interrupt.
1 = Interrupt requested
0 = Interrupt not requested

MR6 — $\overline{INT2}$ Interrupt Request Mask
1 = Inhibits $\overline{INT2}$ interrupt request
0 = Does not inhibit $\overline{INT2}$ interrupt request

MR5 — Auxiliary Counter Status/Preset Bit
If not masked by MR4, it will drive a switch to V_{SS} on the $\overline{RESET}$ pin causing the MCU to reset. This bit may

be used as an auxiliary counter preset bit. If MR5 is clear, a write of logic one will preset the auxiliary counter (MR5 will remain zero), and if set, a write of logic zero will preset the auxiliary counter.
1 = Auxiliary counter overflow
0 = Auxiliary counter clear

MR4 — Watchdog Control Bit
This bit cannot be set via software. The watchdog timer can only be disabled by reset.
1 = Watchdog timer disabled
0 = Watchdog timer enabled

MR3 — Prescaler 1 Clear Bit
Presets the contents of prescaler 1 to $7F.
1 = Prescaler 1 preset
0 = Prescaler 1 not preset

MR2 — Prescaler 2 Clear Bit
Presets the contents of prescaler 2 to $7FFF.
1 = Prescaler 2 preset
0 = Prescaler 2 not preset

MR1 — Prescaler Cross-Couple Bit
This bit controls the output of prescalers 1 and 2 and directs them to either timer A or B clock inputs.
1 = Prescaler 1 feeds timer B clock input, and prescaler 2 feeds timer A input
0 = Prescaler 1 output is used as clock input for timer A, and prescaler 2 output is used as clock input for timer B

MR0 — Port B Toggle Cross-Couple Bit
This bit controls the overflow pulses of timers A and B and directs them to either port B0 or B1.
1 = Timer A overflow output is directed to port B0, and timer B or timer C (depending on the status of MOR5) output is directed to port B1
0 = Overflow output pulse of timer A is used as a port B1 data register toggle clock source, and timer B or timer C overflow output pulse is directed to port B0 toggle clock input

RESETS

The MCU can be reset four ways: (1) by initial power-up; (2) by the external reset input ($\overline{\text{RESET}}$); (3) by a forced reset generated by the "watchdog" counter; and (4) by an optional internal low voltage detect circuit. The $\overline{\text{RESET}}$ input consists mainly of a Schmitt trigger that senses the line logic level. Figure 6 shows the MCU reset circuit.

POWER-ON-RESET (POR)

An internal reset is generated on power-up that allows the internal clock generator to stabilize. The power-on reset is used strictly for power turn-on conditions and should not be used to detect any drop in the power supply voltage. A delay of t_{RHL} milliseconds is required before allowing $\overline{\text{RESET}}$ input to go high. Connecting a capacitor to the $\overline{\text{RESET}}$ input (Figure 7) typically provides sufficient delay.

EXTERNAL RESET INPUT

The MCU is reset when a logic zero is applied to the $\overline{\text{RESET}}$ input for a period longer than one machine cycle (t_{CYC}). Under this type of reset, the Schmitt trigger switches off at V_{IRES-} to provide an internal reset voltage.

FORCED RESET

If the auxiliary counter reset mask bit in the miscellaneous counter (MR4) is cleared and the auxiliary counter

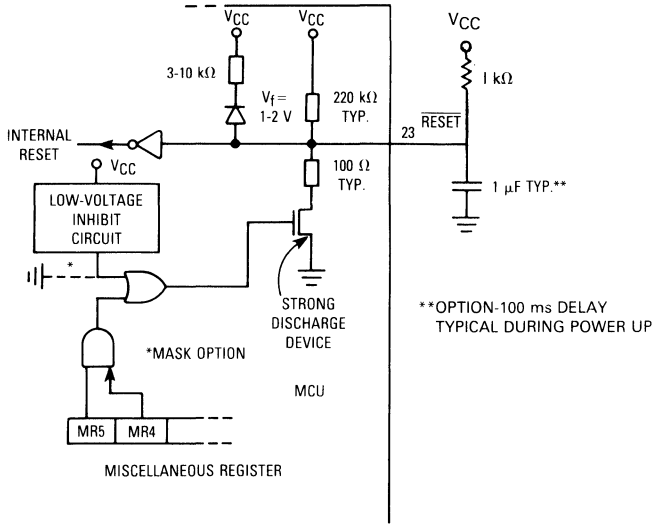


Figure 6. MCU Reset Circuit

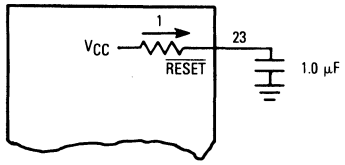


Figure 7. Power-Up Reset Delay Circuit

status bit (MR5) is set, as a result of counter overflow, a switch to V_{SS} is turned on pulling the $\overline{\text{RESET}}$ pin low. A consequent voltage drop below V_{IRES-} on $\overline{\text{RESET}}$ causes a reset, which in turn sets MR4. Switching to V_{SS} when the $\overline{\text{RESET}}$ pin is turned off allows voltage to rise above V_{IRES+}, after which the reset is released. $\overline{\text{RESET}}$ pin voltage variation occurring as a result of forced reset may be amplified externally in order to provide a reset to other peripheral circuits in the system. The reset output from the MCU is not TTL compatible.

3

LOW-VOLTAGE INHIBIT (LVI)

The optional low-voltage detection circuit causes a reset of the MCU if the power supply voltage falls below a certain level (V_{LVI}). The only requirement is that the V_{CC} must remain at or below the V_{LVI} threshold for one t_{cyc} minimum.

In typical applications, the V_{CC} bus filter capacitor will eliminate negative-going voltage glitches of less than one t_{cyc}. The output from the low-voltage detector is connected directly to the internal reset circuitry. It also forces the $\overline{\text{RESET}}$ pin low via a strong discharge device through a resistor. The internal reset is removed once the power supply voltage rises above a recovery level (V_{LVR}) at which time a normal power-on reset occurs.

INTERRUPTS

The MCU can be interrupted eight different ways: through the external interrupt $\overline{\text{INT1}}$ input pin, with the internal timer (either A, B, or C) interrupt request, using the software interrupt instruction (SWI), SPI interrupt request, external port D bit 6 ($\overline{\text{INT2}}$) input pin, or at reset.

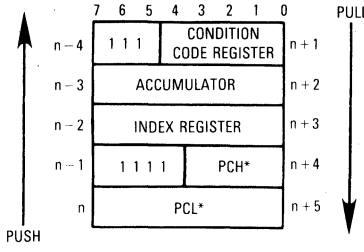
Interrupts cause the processor registers to be saved on the stack and the interrupt mask (I bit) set to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack after which normal processing resumes. The stacking order is shown in Figure 8.

Unlike $\overline{\text{RESET}}$, hardware interrupts do not cause the current instruction execution to be halted, but are considered pending until the current instruction is complete.

NOTE

The current instruction is considered to be the one already fetched and being operated on.

When the current instruction is complete, the processor checks all pending hardware interrupts and, if unmasked



*For subroutine calls, only PCH and PCL are stacked.

Figure 8. Interrupt Stacking Order

(I bit clear), proceeds with interrupt processing; otherwise, the next instruction is fetched and executed. Masked interrupts are latched for later interrupt service. If the timer interrupt status bit is cleared before unmasking the interrupt, then the interrupt is not latched.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction regardless of the setting of the I bit. Refer to Figure 9 for the reset and interrupt instruction processing sequence.

TIMER INTERRUPT

Each interrupt, except $\overline{\text{INT1}}$, has a separate mask bit which must also be cleared, in addition to the I bit, for the MCU to acknowledge the interrupt. The $\overline{\text{INT2}}$, timer A, timer B, timer C, and SPI interrupts each have their own independent mask bits contained in MR6, TACR6, TBCR6, TCOM, TCCM, and SPICR6. The interrupt routine must determine the source of the interrupt by examining the interrupt request bits, TACR7, TBCR7, MR7, TCOF, TCCF, and SPICR7. These bits must be cleared by software. The $\overline{\text{INT1}}$ interrupt has its own vector address. Therefore, the $\overline{\text{INT1}}$ interrupt request is cleared automatically, and then the $\overline{\text{INT1}}$ vector is serviced.

EXTERNAL INTERRUPT

The external interrupt is internally synchronized and then latched on the falling edge of $\overline{\text{INT1}}$ and $\overline{\text{INT2}}$. Clearing the I bit enables the external interrupt. The $\overline{\text{INT2}}$ interrupt has an interrupt request bit (bit 7) and a mask bit (bit 6) in the miscellaneous register (MR). The $\overline{\text{INT2}}$ interrupt is inhibited when the mask bit is set. The $\overline{\text{INT2}}$ is always read as a digital input on port D. The $\overline{\text{INT2}}$ and timer interrupt request bits, if set, cause the MCU to process an interrupt when the condition code I bit is clear. The following paragraphs describe two typical external interrupt circuits.

Zero-Crossing Interrupt

A sinusoidal input signal ($f_{\overline{\text{INT1}}}$ maximum) can be used to generate an external interrupt (see Figure 10a) for use as a zero-crossing detector (for negative transitions of the ac sinusoid). This type of circuit allows applications

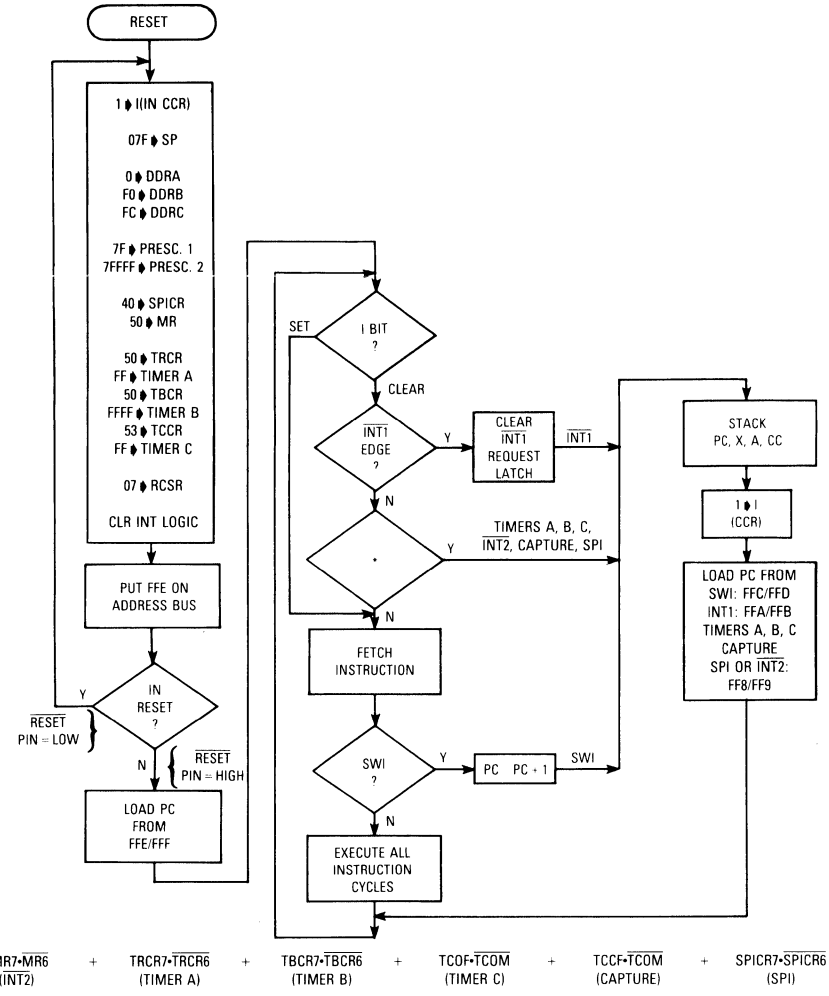


Figure 9. Reset and Interrupt Processing Flowchart

such as servicing time-of-day routines and engaging/disengaging ac power control devices. Off-chip, full-wave rectification provides an interrupt at every zero crossing of the ac signal and, thereby, provides a 2f clock.

Digital-Signal Interrupt

With this type of circuit (Figure 10b), the $\overline{INT1}$ pin can be driven by a digital signal. The maximum frequency of a signal that can be recognized by the TIMER or $\overline{INT1}$ pin logic is dependent on the parameter labeled t_{WL}, t_{WH}. Refer to **TIMER** for additional information.

SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. If the I bit is zero, SWI executes after the other interrupts. The SWI execution is similar to the hardware interrupts.

TIMERS

The MCU has four timers and two programmable prescalers. The timers are identified as timer A, B, C, and the auxiliary counter. Refer to Figure 11 for timers A, B, and C block diagram. The following paragraphs described the different timers.

TIMER A

Timer A is an 8-bit programmable down counter, which can be loaded under program control. Timer A also

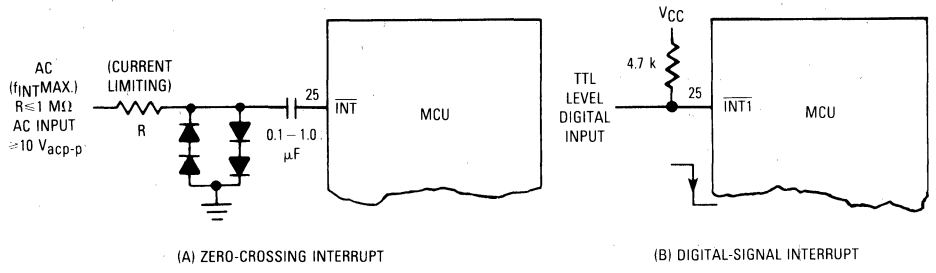


Figure 10. External Interrupt

3

includes a modulus latch which allows the timer to be "auto-reloaded." As clock inputs are received, timer A decrements toward $00. When $00 is reached, bit 7 in the timer A control register is set and the timer is reloaded with the contents of the modulus latch. An underflow condition is also generated when value $00 is reached. This state can be used to toggle bit 0 or bit 1 of port B directly under the control of the miscellaneous register (MR0), the SPI control register, and the port B data direction register. Setting TACR6 or the I bit in the condition control register will prevent timer interrupts from being processed. The timer interrupt request bit *MUST* be cleared by software. There are three ways of loading data from the modulus latch into timer A as described in the following paragrahs.

Direct Loading

When the MCU writes to timer A data register, the data is latched by the modulus latch, and forced into the timer. This operation requires that TACR3 be cleared.

Asynchronous External Event Loading

When TACR3 is a logic one, the contents of the modulus latch are transferred to the timer at the rising edge of $\overline{INT2}$ interrupt request bit (MR7) gated with interrupt request mask bit (MR6). If this loading is used, care must be taken in programming as it will start an interrupt service routine if the I bit in the CCR is clear. Loading $00 to timer A allows a countdown of 256 clocks before the next $00 state is reached.

Auto-Loading

The modulus latch is automatically loaded when the timer reaches $00. This loading is dependent on the setting of TACR3. Auto-loading also occurs in both the previous loading modes. Timer A can be read at any time without affecting the countdown of the timer. The timer and modulus latch are set to $FF on reset.

NOTE

Loading $01 to timer A should be avoided when operating with a divide-by-one prescaler. Doing so will inhibit timer A auto-loading, interrupt generation, and port B toggle mechanisms.

TIMER A CONTROL REGISTER $09

7	6	5	4	3	2	1	0
TACR7	TACR6	TACR5	TACR4	TACR3	TACR2	TACR1	TACR0
RESET:							
0	1	0	0	0	0	0	0

TACR7 — Timer A Interrupt Request Flag
1 = Timer A has transition to $00
0 = Software or reset cleared
TACR6 — Timer A Interrupt Request Mask
1 = Interrupt request inhibited
0 = Interrupt request not inhibited
TACR5 — External or Internal Bit
1 = External clock source for prescaler 1
0 = Internal clock source for prescaler 1
TACR4 — External Enable Bit
Control bit used to enable the external timer pin (PRESCALER1/PC0).

TACR5	TACR4	Prescaler 1 Clock Source
0	0	Internal Clock
0	1	AND of Internal Clock and PRESCALER1/PC0*
1	0	Inputs Disabled
1	1	PRESCALER1/PC0* Low-to-High Transition

*The status of PRESCALER1/PC0 depends upon the data direction status of PRESCALER1/PC0. If PRESCALER1/PC0 is an output, then the clock source is equal to the port data register content, independent of the port electrical loading. If an input, then the clock source is the logic level of PRESCALER1/PC0.

TACR3 — Timer A Load Mode Control
1 = Asynchronous external event loading ($\overline{INT2}$ driven loading is enabled)
0 = Allows direct loading of timer A
TACR2, TACR1, TACR0 — Prescaler 1 Division Ratio Control Bits
When set, these bits select one of eight possible outputs on prescaler 1.

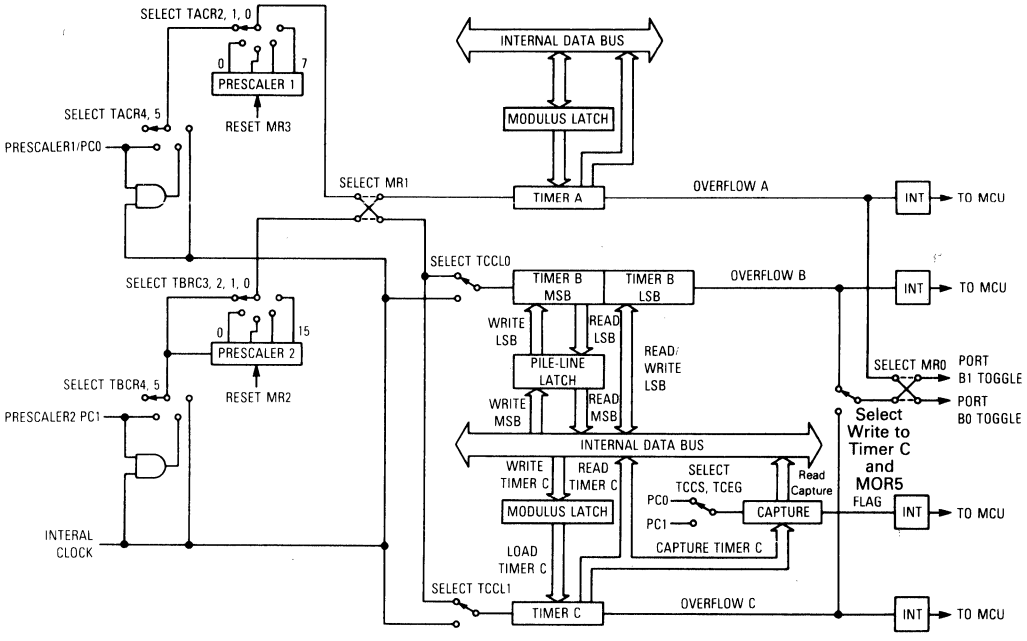


Figure 11. Timers A, B, and C Block Diagram

TACR2	TACR1	TACR0	Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

TIMER B

This is a 16-bit timer which is accessed via two registers ($0B for the most-significant byte (MSB) and $0C for the least-significant byte (LSB)). The MSB has a "pipeline" latch that allows a "snap shot" value of the entire 16 bits to be read. Read/write operations to the LSB are direct. The LSB can be read at anytime without disturbing the count. When the LSB is read, the contents of the MSB are loaded into the pipeline latch so a read of the MSB is actually the contents of the latch.

When writing to the LSB, the contents are immediately entered into the timer. At the same time the pipeline contents are forced into the MSB of the timer. This allows a 16-bit word to be placed into the timer data register during a LSB write operation. An underflow condition is also generated when value $00 is reached. This state can be used to toggle bit 0 or bit 1 of port B directly under the control of the miscellaneous register (MR0), the SPI control register, and the port B data direction register. Setting TBCR6 or the I bit in the condition control register will prevent timer interrupts from being processed. The timer interrupt request bit *MUST* be cleared by software.

TIMER B CONTROL AND STATUS REGISTER $0D

7	6	5	4	3	2	1	0
TBCR7	TBCR6	TBCR5	TBCR4	TBCR3	TBCR2	TBCR1	TBCR0
RESET:							
0	1	0	0	0	0	0	0

TBCR7 — Timer B Interrupt Request Flag
1 = Timer B has transition to $00
0 = Software or reset cleared

TBCR6 — Timer B Interrupt Request Mask
1 = Interrupt request inhibited
0 = Interrupt request not inhibited

TBCR5 — External or Internal Bit
1 = External clock source for prescaler 2
0 = Internal clock source for prescaler 2

TBCR4 — External Enable Bit
Control bit used to enable the external timer pin (PRESCALER2/PC1).

TBCR5	TBCR4	Prescaler 2 Clock Source
0	0	Internal Clock
0	1	AND of Internal Clock and PRESCALER2/PC1*
1	0	Inputs Disabled
1	1	PRESCALER2/PC1* Low-to-High Transition

*The status of PRESCALER2/PC1 depends upon the data direction status of PRESCALER2/PC1. If PRESCALER2/PC1 is an output, then the clock source is equal to the port data register content, independent of the port electrical loading. If an input, then the clock source is the logic level of PRESCALER2/PC1.

TBCR3, TBCR2, TBCR1, TBCR0 — Prescaler 2 Division Ratio Control Bits
When set, these bits select one of eight possible output on prescaler 2.

TBCR3	TBCR2	TBCR1	TBCR0	Divide By
0	0	0	0	1
0	0	0	1	2
0	0	1	0	4
0	0	1	1	8
0	1	0	0	16
0	1	0	1	32
0	1	1	0	64
0	1	1	1	128
1	0	0	0	256
1	0	0	1	512
1	0	1	0	1024
1	0	1	1	2048
1	1	0	0	4096
1	1	0	1	8192
1	1	1	0	16384
1	1	1	1	32768

TIMER C

Timer C is an 8-bit programmable down counter. The timer contains a modulus latch which allows the timer to be auto reloaded. The timer auto reloads with the contents of the modulus latch upon every $01 to $00 transition. Timer C contains a capture register. This read-only register and the contents are refreshed by the contents of the data register during the capture instance. The timer can be written to at any time, and the contents of both the data register and modulus latch are updated immediately. The timer is set to $FF on reset, but the contents of the capture register are not valid until the first capture after reset.

TIMER C CONTROL REGISTER $015

7	6	5	4	3	2	1	0
TCOF	TCOM	TCCF	TCCM	TCEG	TCCS	TCCL1	TCCL0
RESET:							
0	1	0	0	0	0	0	0

TCOF — Timer C Overflow Flag
1 = Timer C has transition to $00
0 = Software or reset cleared

TCOM — Timer C Interrupt Mask
1 = Interrupt request inhibited
0 = Interrupt request not inhibited

TCCF — Timer C Capture Flag
1 = Proper capture occurred on PRESCALER1 or PRESCALER2. No new capture occurs when set
0 = Software or reset cleared

TCCM — Timer C Capture Interrupt Request Mask
- 1 = Inhibits interrupt request generated from TCCF
- 0 = Does not inhibit interrupt request generated from TCCF

TCEG — Timer C Capture Edge Select
- 1 = Selects rising edge of PC0 or PC1 to be capture instance
- 0 = Selects falling edge of PC0 or PC1 to be capture instance

TCCS — Timer C Capture Source Select
- 1 = Select PRESCALER2/PC1 as capture source
- 0 = Select PRESCALER1/PC0 as capture source

TCCL1 and TCCL0 — Timer C Clock Source Select
Clock source selection is defined below.

TCCL1	Timer C Source
0	Internal Clock
0	Internal Clock
1	MR1 Status*
1	MR1 Status*

TCCL0	Timer B Source
0	Internal Clock
1	MR1 Status*
0	Internal Clock
1	MR1 Status*

NOTES:
1. *Denotes prescaler 1 or 2 clock source depending on miscellaneous register bit 1 (MR1) status.
2. MR1 bit cleared (logic zero) at reset:
   Prescaler 1 clock selected to timer A
   Prescaler 2 clock selected to timer B and C
3. MR1 bit set (logic one):
   Prescaler 1 clock selected to timer B and C
   Prescaler 2 clock selected to timer A
4. Prescaler 1 output determined by the status of Timer A control register bits 2, 1, and 0 (TACR2, TACR1, and TACR0).
5. Prescaler 2 output determined by the status of Timer B control register bits 3, 2, 1, and 0 (TBCR3, TBCR2, TBCR1, and TBCR0).

PRESCALER 1

Prescaler 1 is a 7-bit binary down counter whose value is selected by TACR2, TACR1, and TACR0. The selected output is used as the clock input to either timer A or B, depending upon the status of the prescaler cross-couple bit (MR1). The type of clock source to prescaler 1 may be selected by TACR5 and TACR4. Prescaler 1 is set to $7F at reset or under program control when a one is written to prescaler 1 clear bit (MR3).

PRESCALER 2

Prescaler 2 is a 15-bit down counter; its value is selected by TBCR3, TBCR2, TBCR1, and TBCR0. The selected output is used as the clock input to either timer A or B, depending upon the status of the prescaler cross-couple bit (MR1). The type of clock source to prescaler 2 may be selected by TBCR5 and TBCR4. Prescaler 2 is set to $7FFF at reset or under program control when a one is written to prescaler 2 clear bit (MR2).

AUXILIARY COUNTER

This register is a fixed counter which is clocked by the internal clock (f_{OSC} divided by four). Total count period is 4095 cycles. The MCU communicates with this counter via the miscellaneous register (MR5 and MR4). Countdown may be aborted at any time under program control, which also resets the counter to 4095 and clears MR5. When MR4 is clear and MR5 is set as a result of counter time out, the reset pin is internally pulled to ground. If the MCU loses control of the program, the "watchdog" timer will bring the MCU back to reset. Refer to Figure 12 for counter operation diagram.

EPROM PROGRAMMING

ERASING THE EPROM

The EPROM can be erased by exposure to high-intensity ultraviolet (UV) light with a wavelength of 2537A. The recommended integrated dose (UV intensity × exposure time) is 25Ws/cm^2. The lamps should be used without software filters, and the MCU should be positioned about one inch from the UV tubes. Ultraviolet erasure clears all bits of the EPROM to the "zero" state. Data can then be entered by progamming "ones" into the desired bit locations.

CAUTION

Be sure that the EPROM window is shielded from light except when erasing. This protects both the EPROM and light-sensitive nodes.

MASK OPTION REGISTER (MOR) $F1E

The MOR is implemented in EPROM and contains all zeros prior to programming. The MOR bits are described in the following paragraphs. This register is not affected by reset.

7	6	5	4	3	2	1	0
CLK	TOPT	PBTS	LVI	*	*	*	SEC

CLK — Clock (oscillator type)
- 1 = Resistor Capacitor (RC)
- 0 = Crystal

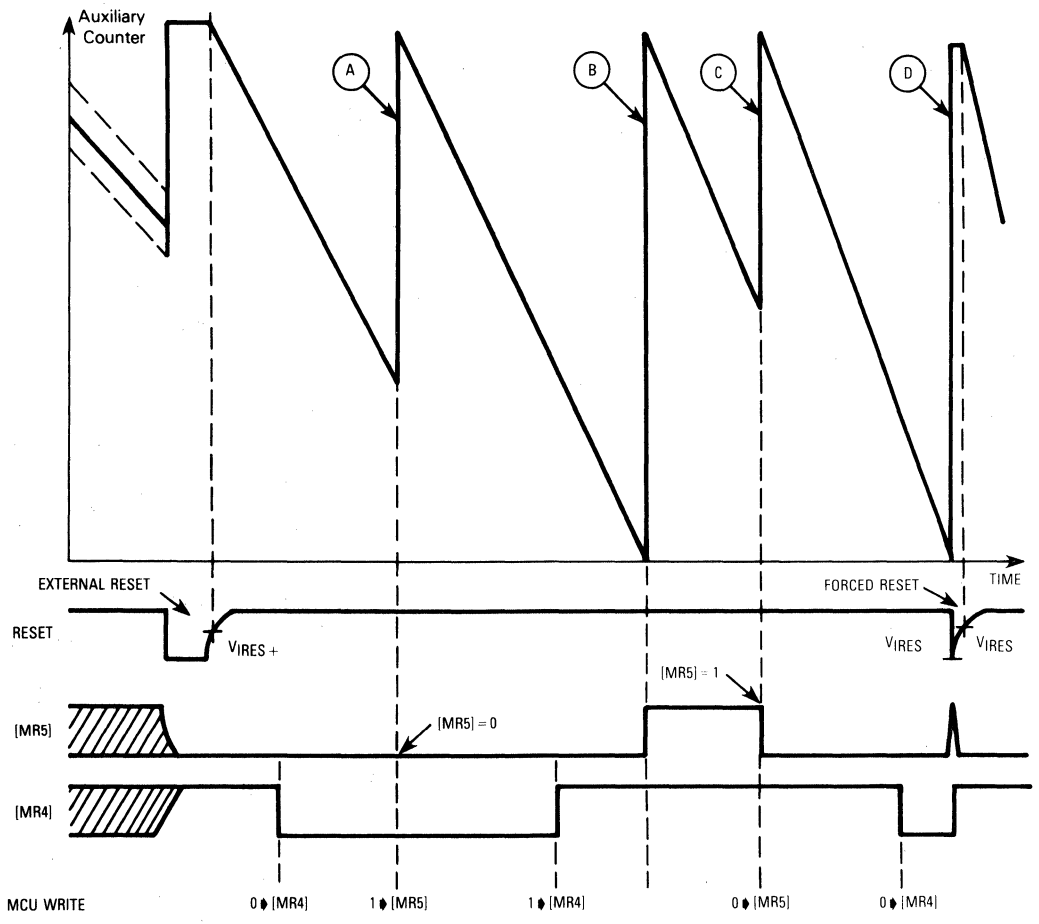
TOPT — Timer Option
- 1 = Enables timer C
- 0 = Disables timer C

PBTS — Port B Toggle Source
This bit is not used on the TJ6 mask set. When cleared the operation is the same as the TJ6 mask set operation.
- 1 = Port B toggle source will come from the timer B overflow even if a write operation is performed on timer C
- 0 = After the first write operation to timer C, the toggle source coming from the timer B overflow is replaced by the timer C overflow. If no write operation is performed on timer C, then timer B is the port B toggle source.

LVI — Low Voltage Inhibit
- 1 = Enables low-voltage detection circuitry
- 0 = Disables low-voltage detection circuitry

Bits 1-3
User available register bits during normal mode of operation

SEC — Security
For full security, this bit must be set in the MOR and mirror MOR ($F16).
- 1 = Enables EPROM read protection
- 0 = Disables EPROM read protection



(A) Counter Preset by Writing "1"

(B) Underflow: MR5 ◗ 1; No Forced Reset

(C) Counter Reset by Writing "0"

(D) Underflow MR5 ◗ 1 Forced Reset

Figure 12. Auxiliary Counter Operation

PROGRAMMING CONTROL REGISTER (PCR) $012

The PCR is an 8-bit register which provides the necessary control bits to program the EPROM. The bootstrap program manipulates the PCR when programming so the user need not be concerned with PCR in most applications.

7	6	5	4	3	2	1	0
1	1	0	1	1	$\overline{VPON}$	$\overline{PGE}$	$\overline{PLE}$
RESET:							
U	U	U	U	U	1	1	1

$\overline{PLE}$ — Programming Latch Enable
Controls address and data being latched into the EPROM. Set during reset, but may be cleared anytime.
1 = Read EPROM
0 = Latch address and data on EPROM

$\overline{PGE}$ — Program Enable
Enables programming of EPROM. Must be set when changing the address and data. Set during reset.
1 = Inhibit EPROM programming
0 = Enable EPROM programming (if $\overline{PLE}$ is low)

$\overline{VPON}$ — V_{PP} On
A read-only bit that indicates high voltage at the V_{PP} pin. When set to "one", disconnects $\overline{PGE}$ and $\overline{PLE}$ from the chip.
1 = No high voltage of V_{PP} pin
0 = High voltage on V_{PP} pin

NOTE

$\overline{VPON}$ being "zero" does not indicate that the V_{PP} level is correct for programming. It is used as a safety interlock for the user in the normal operating mode.

$\overline{VPON}$	$\overline{PGE}$	$\overline{PLE}$	Programming Conditions
0	0	0	Programming enabled (program EPROM byte)
1	0	0	$\overline{PGE}$ and $\overline{PLE}$ bits disabled
0	1	1	Programming disabled (latch address and data in EPROM)
1	1	0	$\overline{PGE}$ and $\overline{PLE}$ disabled
0	0	0	Invalid state
1	0	1	Invalid state
0	1	1	Voltage applied to $\overline{RESET}$/V_{PP} pin
1	1	1	$\overline{PGE}$ and $\overline{PLE}$ disabled (operating mode)

PROGRAMMING

The MCU bootstrap program can be used to program the EPROM. The vectors at address $FF6 and $FF7 are used to start the program. This vector is fetched when V_{IHTP} is applied to the PRESCALER/PC0 pin and the $\overline{RESET}$ pin is allowed to rise above V_{IRES+}. The level on the PRESCALER/PC1 pin, when the $\overline{RESET}$/V_{PP} pin rises above V_{IRES+}, determines which programming mode is selected. A high level on PRESCALER/PC1 selects the auto-programming operation.

A M2532/2732 UV EPROM must first be programmed with the same information that is to be transferred to the MCU EPROM. Unprogrammed EPROM address locations should contain $00 to speed up the programming operation. Figure 13 is a schematic diagram for a board and circuitry that can be used to program the MCU EPROM.

Perform the following steps to program the MCU EPROM:

1. Insert the programmed EPROM and erased MCU EPROM into U2 and U3.
2. Programming operation starts when S1 is placed to the ON position.
   a) DS1 and DS2 illuminate.
   b) MCU control is transferred to the bootstrap ROM, and the programming routine executed by the bootstrap loader program.
   c) DS3 blinks during programming. When programming is complete, DS3 remains illuminated.
   d) After two seconds DS4 will illuminate indicating the MCU has been programmed and verified.
3. Remove power by placing S1 to the OFF position and remove programmed MCU.

NOTE

No programming can be done once the MOR and mirror MOR security bit has been programmed to logic one. The only way to proceed from the secure mode to the non-secure mode is by erasing the MCU. The MCU must be reset following programming of the SEC bits to enable the security feature.

EMULATION

The MCU is designed to emulate the functions of either the MC6805S2 or MC6805S3. However, due to pin assignments, processing, and mask options, the MCU has some differences. The differences are listed as follows:

1. Port A output on the MC6805S2/S3 is a mask option. The CMOS pullup option on port A is not implemented on MC68705S3. If this option is required, pullup resistors must be installed.
2. The RC clock on the MC6805S2/S3 is a mask option. To enable the MC68705S3 RC clock, MOR bit 7 must be programmed to a logical one.
3. The LVI on the MC6805S2/S3 is a mask option. To enable the LVI on MC68705S3, MOR bit 4 must be programmed to a logical one.
4. The MC68705S3 $\overline{RESET}$/V_{PP} and V_{STBY}/AN4/$\overline{INT2}$/PD6 electrical characteristics are different for the MC6805S2/S3.
5. Pin 4 (AN4 and V_{STBY}) on MC6805S2/S3 is a mask option. On the MC68705S3, pin 4 is enabled for V_{STBY}/AN4/$\overline{INT2}$/PD6.
6. On MC6805S2/S3 pin 4, standby RAM contents will be lost if the voltage drops below 3.0 V. Standby RAM on the MC68705S3 will not be lost unless voltage drops below 4.0 V.
7. Above certain voltages (3.7 V typical), pin 4 will exhibit lower input impedance than the MC6805S2/S3. This may cause A/D conversion inaccuracies if the

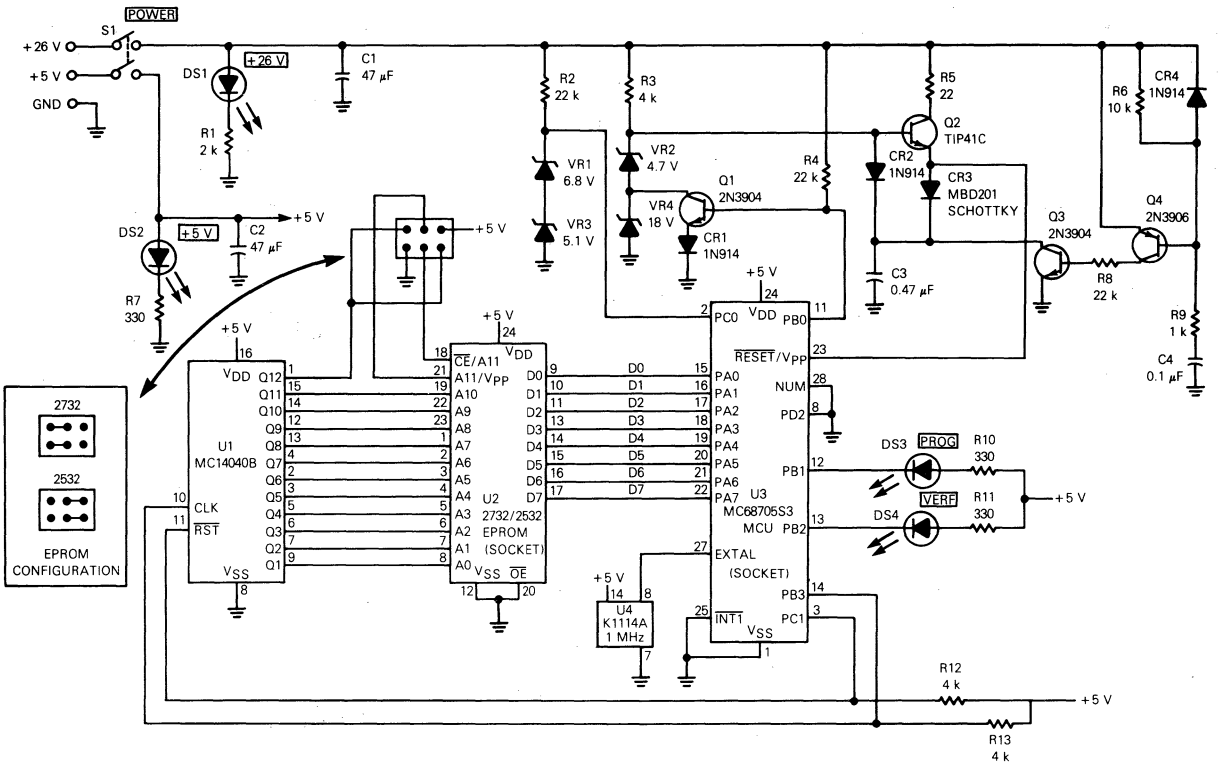


Figure 13. Programming Connections Schematics Diagram

pin is used as fifth A/D input channel. Pin 4 is always a high impedance input on the MC6805S2/S3.

8. Reset and Vpp functions share a common pin (23) on the MC68705S3. Therefore, electrical characteristics on this pin may vary from the same pin on MC6805S2/S3. The input impedance on the MC68705S3 pin is approximately equivalent to the 1.0 ohm pulldown resistor; whereas, on the MC6805S2/S3, this pin is a high impedance (220K ohms) input. Therefore, the MC68705S3 requires a pullup resistor on the $\overline{\text{RESET}}$ pin to recover from a reset condition.

SERIAL PERIPHERAL INTERFACE

The serial peripheral interface (SPI) has arbitration on the data and clock lines. The SPI communicates with the MCU via data and control registers. The SPI data and clock inputs are always taken from their respective I/O ports, regardless of the status of the data direction registers relative to that port. The SPI can operate in modes from auto clocked (NRZ), half duplex, and full duplex with from a one to a four wire combination. Refer to Figure 14 for the SPI block diagram.

SPI CONTROL AND STATUS REGISTER

This 8-bit register contains the status and control bits relative to SPI operations. The SPI control register operation is shown in Figure 15. The SPI control and status register bits can be set or cleared under program control.

7	6	5	4	3	2	1	0
SPICR7	SPICR6	SPICR5	SPICR4	SPICR3	SPICR2	SPICR1	SPICR0
RESET:							
0	1	0	0	0	0	0	0

SPICR7 — SPI Interrupt Request Bit
Set on eighth data input strobe. MCU services this interrupt if I bit is clear in CCR.
1 = Interrupt request (if SPICR6 not masked)
0 = No interrupt pending

SPICR6 — SPI Interrupt Request Mask Bit
1 = Disables interrupt request from SPICR7
0 = Enables interrupt request from SPICR7

SPICR5 — SPI Clock Sense Bit/Bus-Busy Flag
Dual-function bit controlled by the status of SPICR4.
1 = Start SPI operation when SPICR4 = 1. Input data latched on positive edge and output data changed on negative edge of SPI clock when SPICR4 = 0.
0 = Stop SPI operation when SPICR4 = 1. Input data latched on negative edge and output data changed on positive edge of SPI clock when SPICR4 = 0.

SPICR4 — SPI Operation Enable Bit
This bit determines the functions of SPICR5 and SPICR2.
1 = Enables SPI data register shifting, data and clock arbitration logic, and slave select input logic
0 = Disables SPI data register shifting, data and clock arbitration logic, and slave select input logic

SPICR3 — SPI Data Output Select Bit
1 = Output of the SPI data register is loaded to port B3 data register at the appropriate SPI clock edge selected by SPICR5, during the active transaction mode
0 = Output of the SPI data register is loaded to port B2 data register at the appropriate SPI clock edge selected by SPICR5, during the active transaction mode

SPICR2 — Port B1 Toggle Enable/Start Bit
Dual-function bit controlled by the status of SPICR4.
1 = Start bit is set by negative transition of the data input of the SPI data shift register while the clock is at the idle level when SPICR4 = 1. Start bit set under program control to enable port B1 data register toggle facility when SPICR4 = 0.
0 = Stop SPI operation when SPICR4 = 1. Cleared under program control when SPICR4 = 0.

SPICR1 — Mode Fault Flag
1 = (a) Mode flag is set when SPI data output arbitration occurs on the SPI data output port (PB3 or PB2) selected by SPICR3. The MCU loses data mastership, and the SPI data output port DDR is cleared.
(b) Mode flag is set if a low level is detected on slave input PB0. Then, the MCU loses clock mastership switching to the clock slave mode, and port B1 DDR is cleared.
(c) Mode flag is set during the idle mode when a negative clock edge is detected on the SPI clock input, and the port B1 data register is cleared.
0 = Cleared under program control

SPICR0 — SPI Input Data Select Bit
1 = SPI data from port B3 is latched into the SPI data register
0 = SPI data from port B2 is routed to the input of the SPI data register

SPI DATA REGISTER

This register can be written to any time and can also be read, regardless of serial operations, without disturbing the data. A one bit shift to the left occurs each time there is a data input strobe while the LSB is loaded with data from port B2 or B3. The MSB is loaded every time there is data output strobe. Data input and output strobes are generated internally only during the active transaction time.

SPI DIVIDE-BY-EIGHT COUNTER

The counter is cleared during SPI deselect or idle modes. A count occurs at every data input strobe during the active transaction mode. At overflow, SPICR7 is set which puts the SPI in idle mode and blocks all data input and output strobes. The counter is cleared when PB0 is high if the SPI is in the slave mode or when a "start" condition is detected.

SPI OPERATION

The SPI can operate in a variety of modes. Software assisted protocols may be defined to upgrade the hardware versatility and/or system performance of the MCU. Some features common to all operating modes are summarized in Table 1 and in the following paragraphs.

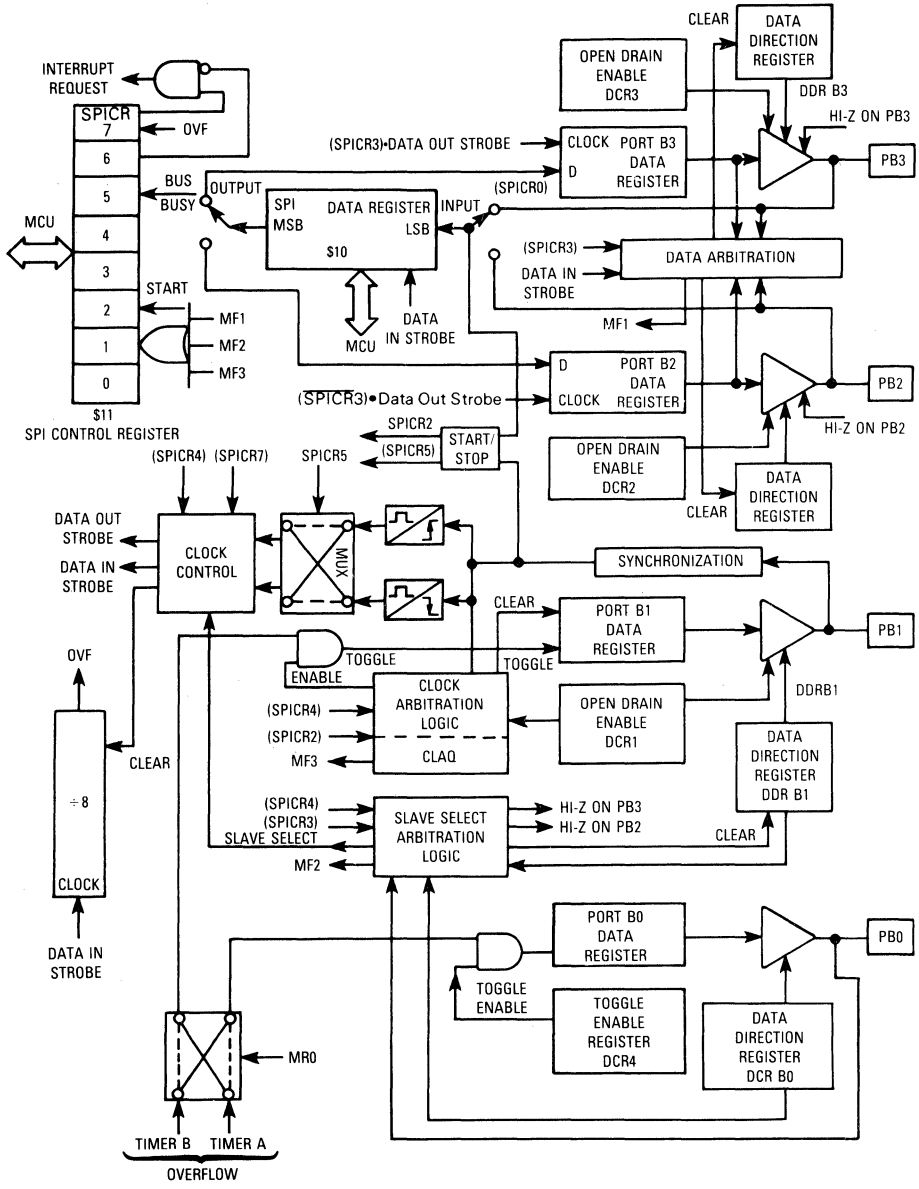


Figure 14. Serial Peripheral Interface Block Diagram

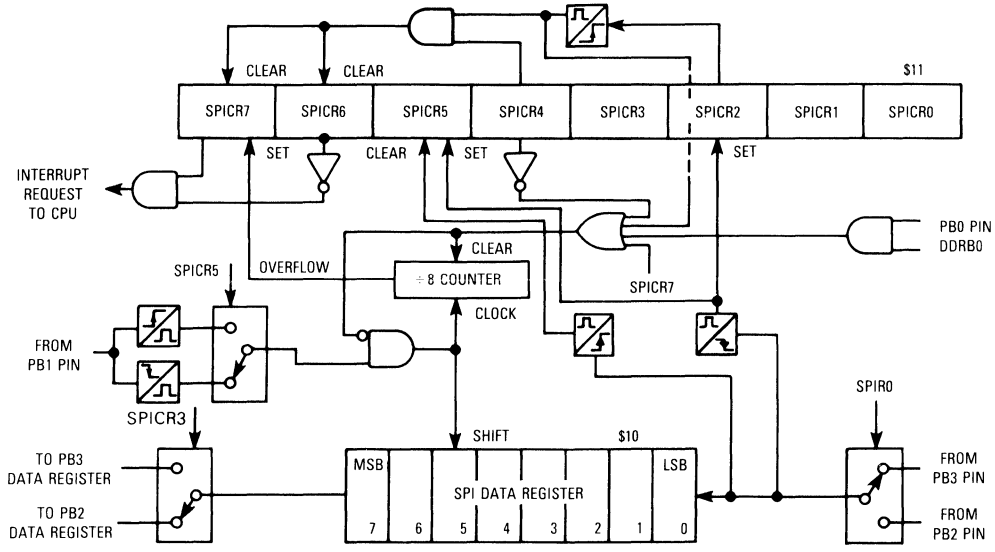


Figure 15. SPI Control Register Operation

1) SPI data input and output may be individually routed to or from PB2 or PB3 (Table 2). These four routings provide half and full duplex operations, as well as allowing bidirectional information to flow in daisy-chained systems.
2) When data input and output is done on PB2, PB3 is available for any other use and vice versa.
3) Data input is always relative to the port pin logic level regardless of the data direction register status on that pin.
4) In full duplex operation, 16 bits of information may be transferred with eight clock pulses between at least two devices with transmit capability. Both PB2 and PB3 are used for data transfer. The same shift register is used for data in and data out. The byte transmitted replaces the byte received. SPICR7 is used to signify that the I/O operation is complete.
5) SPI clock is always provided on port B1. In the clock slave mode, port B1 DDR is in the input mode (cleared). In the clock master mode, port B1 DDR is set; therefore, the MCU imposes the clock level on PB1 until there is clock arbitration on the clock line or until the MCU loses clock mastership when PB0 goes low.
6) No fixed baud rate generation exists. The clock frequency is dependent on the prescaleer clock source option, prescaler divide ratio, and timer divide ratio as well as the port C status in case of external clocking for the timer. Toggling of the port B1 data register is automatically allowed during the active transmission mode.
7) All devices connected to the SPI must have their output and input data strobe on the same clock edge for correct transfer of data.
8) During the active transmission mode, the first clock edge must be the output data strobe. When this occurs, the MSBs of the data registers of all transmitters are copied onto the data output pins, and the MCU copies the MSB of its SPI onto the port B2 or B3 data register.
9) Port B data direction registers and port B data control registers are accessible during SPI operation. During active transaction mode, the PB1 data register, PB2 data register (if SPICR3 = 0), and PB3 (if SPICR3 = 1) are not write accessible under program control.
10) Port B lines not used for SPI can be used for other digital functions.

SELECT INPUT OPERATION

An external device supplies slave select information via port B0. If slave select is not used, set port B0 to output mode to inhibit slave select function.

The following paragraphs describe clock master and clock slave operating modes of the SPI.

Master Mode Slave Select Actions

The MCU monitors slave select input in master mode to assure that it stays false. If slave select goes true, the MCU exits master mode and becomes a slave. This implies that a write collision has occurred which means two

Table 1. Summary of SPI Operations

DEFINITIONS
- Transmitter — Data Master: DDRB2 or 3 = 1
- Receiver — Data Slave: DDRB2 or 3 = 0
- Clock Master: DDRB1 = 1
- Clock Slave: DDRB1 = 0
- Transaction Mode: SPICR4 = 1
  1) Active: $\overline{SPICR7}\bullet(\overline{DDRB0}\bullet\overline{PB0} + DDRB0)$ if DDRB1 = 0 (clock slave mode) or
     $\overline{SPICR7}\bullet(\overline{DDRB0}\bullet\overline{PB0} + DDRB0)$ if DDRB1 = 1 (clock master mode)
     Clock Pulses allowed, data shifted
  2) Idle: $SPICR7 + \overline{DDRB0}\bullet PB0$ if DDRB1 = 0 (clock slave mode)
     Clock pulses blocked, data output line in high-impedance state
- Deselect Mode: SPICR4 = 0 – No SPI Operations

SLAVE SELECT INPUT
- Slave Select Input: SPISS – PB0
- If DDRB0 = 0 then so SPISS action on MCU
  1) Master Mode: SPISS = 1 DDRB1 = 1
     SPISS 1 – 0: Switch to Slave Mode (DDRB1 1 – 0)
     Set SPICR1 (Mode Fault Flag)
  2) Slave Mode: SPISS = 0 DDRB1 = 0
     External clock is allowed to shift data in/out. If SPISS is pulled high, the external clock input pulses are inhibited; no data shift; divide-by-eight counter cleared; SPID (PB2 or PB3) switched to high-impedance state.
- Used as Chip-Select Input

DATA ARBITRATION

Data master loses data mastership when data collision occurs during internal data strobe time.

If SPID output port (PB2 or PB3) = 1 while actual pin level is pulled low externally — conflict detected at internal data strobe time.

Then SPICR1 (mode fault flag) is set; SPID output port DDR (B2 or B3) 1 ➧ 0 (high-impedance state).

CLOCK ARBITRATION

MCU has clock mastership (DDRB1 = 1)
1) Via SPISS line (DDRB0 = 0). If SPISS is pulled low, then clock mastership lost; DDRB1 1 ➧ 0 (high-impedance state); SPICR1 is set (mode fault flag).
2) Via clock line SPICL (DDRB1 = 1 and DCRB5 = 0)
   Condition: SPICL must have open-drain output (DCRB5 = 0)

   If clock line is held low externally then clock mastership is not lost; minimum t_{CLH} and t_{CLK} times are guaranteed.

   If SPICL goes low during idle mode then SPICR1 = 1 and clock line is switched low to inhibit the system clock.

MODE FAULT FLAGE OPERATION (SPICR1)

Flag set when any of the following conditions occur:
- Data arbitration occurs on SPID output.
- Clock arbitration with SPISS during master to slave switching.
- Clock arbitration via clock line if SPICL 1 ➧ 0 during idle.

START, STOP, AND CLOCK IDLE CONDITIONS

Clock Idle: The clock level just prior to the transition that causes data on the serial output data line to be changed is defined as the SPI clock idle state.
SPICR5 = 0: SPICL Idle = Low State
SPICR5 = 1: SPICL Idle = High State

These definitions are necessary for determining start and stop conditions.

NOTE

Clock idle state can only be defined if SPICR4 = 0 (Deselect Mode)

Start Condition: Any negative transition of the data input line (PB2 or PB3) during an SPICL idle state.
Stop Condition: Any positive transition of the data input line during an SPICL idle state.

Table 2. Port B Status During SPI Operation

Port Name	Use	Input	Output	Comments
PB0	SPISS	Yes	No	Used as slave select input
PB0	Data	No	Yes	Used as "busy" signal or any digital output
PB1	SPICL	Yes	No	Clock slave
PB1	SPICL	No	Yes	Clock master
PB2	SPID	Yes	No	SPI data input SPICR0 = 0
PB2	SPID	No	Yes	SPI data output SPICR3 = 0
PB2	Data	Yes	Yes	Any digital signal SPICR3 = 1
PB3	SPID	Yes	No	SPI data input SPICR0 = 1
PB3	SPID	No	Yes	SPI data output SPICR3 = 1
PB3	Data	Yes	Yes	Any digital signal SPICR3 = 0

devices attempted to become masters. Write collisions normally result from a software error, and the default master must clean up the system. The mode fault flag is set to signal that clock mastership is lost. Slave select actions can take place during either active or idle transaction modes.

Slave Select Input Actions During Slave Mode

The current clock master generates slave select to enable one of several slaves to accept or return data. The $\overline{SS}$ signal must go low before serial clock pulses occur and must remain low until after the eighth serial clock cycle. Individual lines or a daisy chain can be used for multiple slaves. When $\overline{SS}$ is high, the following occur:

- Serial data output is forced to a high-impedance state without affecting the DDR status.
- Serial clock input pulses are inhibited from generating internal data output and input strobe pulses.
- The eight-bit counter is cleared.

SPI OPERATING MODES

Six methods of operating the SPI are discussed in the following paragraphs.

One-Wire Autoclocked Mode

Various SPI devices can be connected on a single wire, with data transmission using an implicit clock, and each device being its own clock master.

Two-Wire Half-Duplex Mode

In this mode, separate data and clock lines connect the elements in the system. Data and clock mastership should be monitored via protocol included in the data patterns. A transmitter can send all zeros to take all other transmitters off the bus.

Three-Wire Half-Duplex Mode with Slave Select Input

This mode is the same as the half-duplex mode except that the slave select input allows using the MCU as a peripheral in a system where clock mastership is passed through the slave select line. Typically, the slave select lines can be wired together. The current master sets its slave select line in the output mode prior to a serial transmission and pulls it low to indicate that the system is busy. This allows the clock master to retain mastership until the end of transmission. Software protocol can be arranged so that slaves do not request mastership until their slave select lines go high. At the end of a transmission, the current master pulls $\overline{SPISS}$ high and puts the $\overline{SPISS}$ port (PB0) in the input mode. A slave requesting clock mastership pulls the $\overline{SPISS}$ line low, removing the current master from the line. Time multiplexed protocols may be required to avoid simultaneous mastership requests.

Three-Wire Full-Duplex Mode

This mode allows the MCU to operate simultaneously as transmitter and receiver. Bus or daisy-chain networks are feasible. Protocols in the data stream are required to change:

- Clock masters
- The number of transmitters in the system
- The direction of data flow in daisy-chained systems with collision

It is possible for the MCU to shift out one byte of data while receiving another, as illustrated in Figure 16. This eliminates the need for XMIT EMPTY or REC FULL status bits.

Three-Wire Full-Duplex Mode with Clock Arbitration

This mode is a mix of the three-wire full-duplex mode and two-wire half-duplex mode with clock arbitration, where the SPI clock line operates as a wire-or. Simultaneous masters are allowed, and clock arbitration is via the clock line.

Four-Wire Full-Duplex Mode with Slave-Select Input

This mode is similar to the three-wire full-duplex mode in network construction and to the three-wire half-duplex mode with slave-select input in clock arbitration and slave selection. Refer to Figure 17.

ANALOG-TO-DIGITAL CONVERTER

The chip resident 8-bit analog-to-digital (A/D) converter uses a successive approximation technique as show in

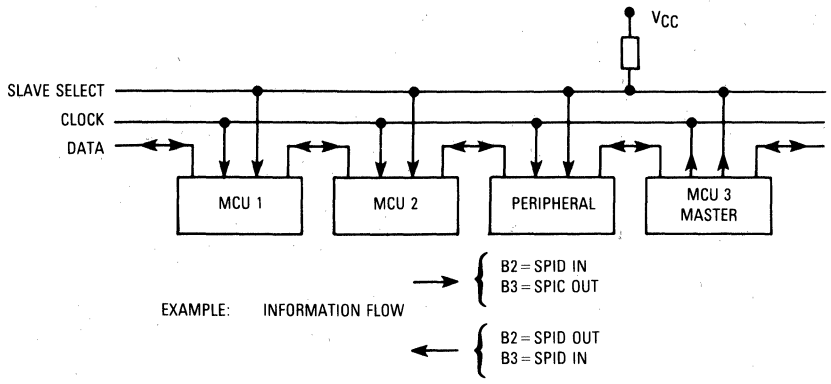


Figure 16. Daisy Chain/Cascade Organization

3

Figure 18. Four external analog inputs can be connected to the A/D through a multiplexer via port D. Four internal analog channels ($V_{RH} - V_{RL}$, $V_{RH} - V_{RL}/2$, $V_{RH} - V_{RL}/4$, and V_{RL}) may be selected for calibration. The accuracy of these internal channels may not meet the accuracy specifications of the external channels.

A fifth external analog input (AN4) is available via the mask option. When selected, it replaces the V_{RH} internal channel. Due to signal routing, the accuracy of this fifth channel may be slightly less than AN0-AN3.

Multiplexer selection is controlled by the A/D control register (ACR) bits 0, 1, and 2. Refer to Table 3 for multiplexer selection. The ACR is shown in Figure 18. The converter uses 30 machine cycles to complete a conversion of a sampled analog input. When the conversion is complete, the digital value is placed in the A/D result register (ARR); the conversion flag is set; selected input is sampled again; and a new conversion begins. When ACR7 is cleared, the conversion in progress is aborted and the selected input, which is held internally, is sampled for five machine cycles.

The converter uses V_{RH} and V_{RL} as reference voltages. An input voltage equal to or greater than V_{RH} converts to $FF. An input voltage equal to or less than V_{RL}, but greater than V_{SS}, converts to $00. Maximum and minimum ratings must not be exceeded. Each analog input source should use V_{RH} as the supply voltage and be referenced to V_{RL} for the ratiometric conversion. To maintain full accuracy of the A/D, three requirements should be followed: (1) V_{RH} should be equal to or less

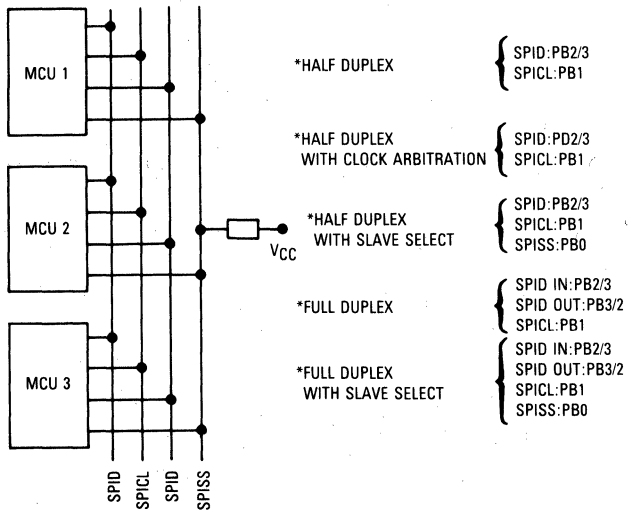


Figure 17. SPI Operation Bus Organization

Table 3. A/D Input MUX Selection

A/D Control Register			Input Selected	A/D Output (Hex)		
ACR2	ACR1	ACR0		Min	Typ	Max
0	0	0	AN0			
0	0	1	AN1			
0	1	0	AN2			
0	1	1	AN3			
1	0	0	V_{RH}**	FE**	FF**	FF**
1	0	1	V_{RL}*	00	00	01
1	1	0	$V_{RH}/4$*	3F	40	41
1	1	1	$V_{RH}/2$*	7F	80	81

*Internal (calibration) levels
**AN4 may replace the V_{RH} calibration channel if selected via mask option.

than V_{CC}, (2) V_{RL} should be equal to or greater than V_{SS} but less than maximum specifications, and (3) $V_{RH} - V_{RL}$ should be equal to or greater than 4 volts.

The A/D has a built-in 1/2 LSB offset intended to reduce the magnitude of the quantizing error to ± 1/2 LSB, rather than +0, −1 LSB with no offset. This implies that, ignoring errors, the transition point from $00 to $01 occurs at 1/2 LSB above V_{RL}. Similarly, the transition from $FE to $FF occurs 1-1/2 LSB below V_{RH}, ideally.

INSTRUCTION SET

The MCU has a set of 59 basic instructions which can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following instruction list.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the first 256 bytes of the memory space where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear and bit test, and

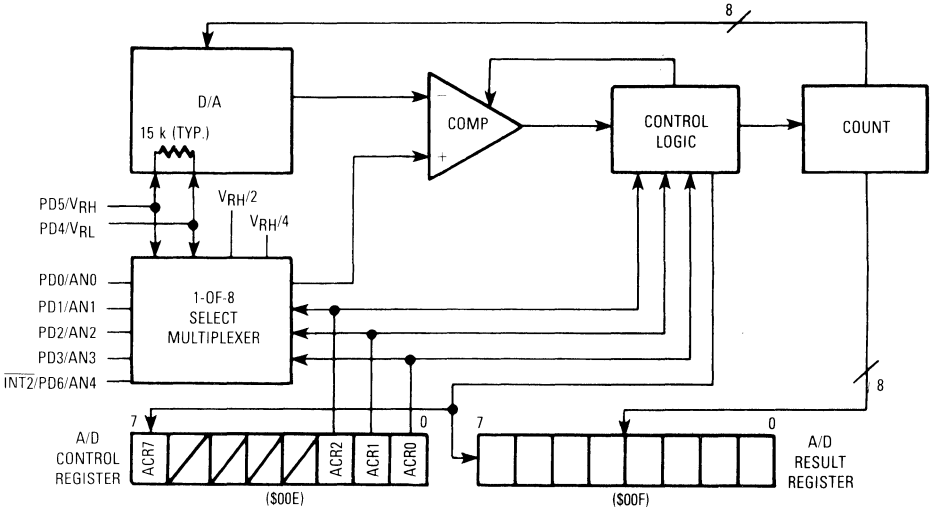


Figure 18. A/D Block Diagram

branch functions are all implemented with a single instruction. For test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch if Bit n is Set	BRSET n (n=0 . . . 7)
Branch if Bit n is Clear	BRCLR n (n=0 . . . 7)
Set Bit n	BSET n (n=0 . . . 7)
Clear Bit n	BCLR n (n=0 . . . 7)

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No-Operation	NOP

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (2's Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch IFF Higher	BHI
Branch IFF Lower or Same	BLS
Branch IFF Carry Clear	BCC
(Branch IFF Higher or Same)	(BHS)
Branch IFF Carry Set	BCS
(Branch IFF Lower)	(BLO)
Branch IFF Not Equal	BNE
Branch IFF Equal	BEQ
Branch IFF Half Carry Clear	BHCC
Branch IFF Half Carry Set	BHCS
Branch IFF Plus	BPL
Branch IFF Minus	BMI
Branch IFF Interrupt Mask Bit is Clear	BMC
Branch IFF Interrupt Mask Bit is Set	BMS
Branch IFF Interrupt Line is Low	BIL
Branch IFF Interrupt Line is High	BIH
Branch to Subroutine	BSR

OPCODE MAP SUMMARY

Table 4 is an opcode map for the instructions used on the MCU.

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single-byte instructions, while the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. Two-byte direct addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The

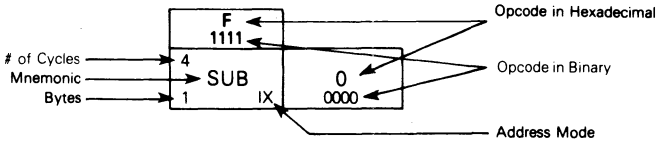
Table 4. Opcode Map

	Bit Manipulation		Branch	Read-Modify-Write					Control		Register/Memory						
	BTB	BSC	REL	DIR	INH	INH	IX1	IX	INH	INH	IMM	DIR	EXT	IX2	IX1	IX	
Hi / Low	0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111	8 1000	9 1001	A 1010	B 1011	C 1100	D 1101	E 1110	F 1111	Hi / Low
0 0000	10 3 BRSET0 BTB	7 2 BSET0 BSC	4 2 BRA REL	6 2 NEG DIR	4 1 NEG INH	4 1 NEG INH	7 2 NEG IX1	6 1 NEG IX	9 1 RTI INH		2 2 SUB IMM	4 2 SUB DIR	5 3 SUB EXT	6 3 SUB IX2	5 2 SUB IX1	4 1 SUB IX	0 0000
1 0001	10 3 BRCLR0 BTB	7 2 BCLR0 BSC	4 2 BRN REL						6 1 RTS INH		2 2 CMP IMM	4 2 CMP DIR	5 3 CMP EXT	6 3 CMP IX2	5 2 CMP IX1	4 1 CMP IX	1 0001
2 0010	10 3 BRSET1 BTB	7 2 BSET1 BSC	4 2 BHI REL								2 2 SBC IMM	4 2 SBC DIR	5 3 SBC EXT	6 3 SBC IX2	5 2 SBC IX1	4 1 SBC IX	2 0010
3 0011	10 3 BRCLR1 BTB	7 2 BCLR1 BSC	4 2 BLS REL	6 2 COM DIR	4 1 COMA INH	4 1 COMX INH	7 2 COM IX1	6 1 COM IX	11 1 SWI INH		2 2 CPX IMM	4 2 CPX DIR	5 3 CPX EXT	6 3 CPX IX2	5 2 CPX IX1	4 1 CPX IX	3 0011
4 0100	10 3 BRSET2 BTB	7 2 BSET2 BSC	4 2 BCC REL	6 2 LSR DTR	4 1 LSRA INH	4 1 LSRX INH	7 2 LSR IX1	6 1 LSR IX			2 2 AND IMM	4 2 AND DIR	5 3 AND EXT	6 3 AND IX2	5 2 AND IX1	4 1 AND IX	4 0100
5 0101	10 3 BRCLR2 BTB	7 2 BCLR2 BSC	4 2 BCS REL								2 2 BIT IMM	4 2 BIT DIR	5 3 BIT EXT	6 3 BIT IX2	5 2 BIT IX1	4 1 BIT IX	5 0101
6 0110	10 3 BRSET3 BTB	7 2 BSET3 BSC	4 2 BNE REL	6 2 ROR DIR	4 1 RORA INH	4 1 RORX INH	7 2 ROR IX1	6 1 ROR IX			2 2 LDA IMM	4 2 LDA DIR	5 3 LDA EXT	6 3 LDA IX2	5 2 LDA IX1	4 1 LDA IX	6 0110
7 0111	10 3 BRCLR3 BTB	7 2 BCLR3 BSC	4 2 BEQ REL	6 2 ASR DIR	4 1 ASRA INH	4 1 ASRX INH	7 2 ASR IX1	6 1 ASR IX		2 1 TAX INH		5 2 STA DIR	6 3 STA EXT	7 3 STA IX2	6 2 STA IX1	5 1 STA IX	7 0111
8 1000	10 3 BRSET4 BTB	7 2 BSET4 BSC	4 2 BHCC REL	6 2 LSL DIR	4 1 LSLA INH	4 1 LSLX INH	7 2 LSL IX1	6 1 LSL IX		2 1 CLC INH	2 2 EOR IMM	4 2 EOR DIR	5 3 EOR EXT	6 3 EOR IX2	5 2 EOR IX1	4 1 EOR IX	8 1000
9 1001	10 3 BRCLR4 BTB	7 2 BCLR4 BSC	4 2 BHCS REL	6 2 ROL DIR	4 1 ROLA INH	4 1 ROLX INH	7 2 ROL IX1	6 1 ROL IX		2 1 SEC INH	2 2 ADC IMM	4 2 ADC DIR	5 3 ADC EXT	6 3 ADC IX2	5 2 ADC IX1	4 1 ADC IX	9 1001
A 1010	10 3 BRSET5 BTB	7 2 BSET5 BSC	4 2 BPL REL	6 2 DEC DIR	4 1 DECA INH	4 1 DECX INH	7 2 DEC IX1	6 1 DEC IX		2 1 CLI INH	2 2 ORA IMM	4 2 ORA DIR	5 3 ORA EXT	6 3 ORA IX2	5 2 ORA IX1	4 1 ORA IX	A 1010
B 1011	10 3 BRCLR5 BTB	7 2 BCLR5 BSC	4 2 BMI REL							2 1 SEI INH	2 2 ADD IMM	4 2 ADD DIR	5 3 ADD EXT	6 3 ADD IX2	5 2 ADD IX1	4 1 ADD IX	B 1011
C 1100	10 3 BRSET6 BTB	7 2 BSET6 BSC	4 2 BMC REL	6 2 INC DIR	4 1 INCA INH	4 1 INCX INH	7 2 INC IX1	6 1 INC IX		2 1 RSP INH		3 2 JMP DIR	4 3 JMP EXT	5 3 JMP IX2	4 2 JMP IX1	3 1 JMP IX	C 1100
D 1101	10 3 BRCLR6 BTB	7 2 BCLR6 BSC	4 2 BMS REL	6 2 TST DIR	4 1 TSTA INH	4 1 TSTX INH	7 2 TST IX1	6 1 TST IX		2 1 NOP INH	8 2 BSR REL	7 2 JSR DIR	8 3 JSR EXT	9 3 JSR IX2	8 2 JSR IX1	7 1 JSR IX	D 1101
E 1110	10 3 BRSET7 BTB	7 2 BSET7 BSC	4 2 BIL REL								2 2 LDX IMM	4 2 LDX DIR	5 3 LDX EXT	6 3 LDX IX2	5 2 LDX IX1	4 1 LDX IX	E 1110
F 1111	10 3 BRCLR7 BTB	7 2 BCLR7 BSC	4 2 BIH REL	6 2 CLR DIR	4 1 CLRA INH	4 1 CLRX INH	7 2 CLR IX1	6 1 CLR IX		2 1 TXA INH		5 2 STX DIR	6 3 STX EXT	7 3 STX IX2	6 2 STX IX1	5 1 STX IX	F 1111

Abbreviations for Address Modes

INH	Inherent
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction.

RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −126 to +129 from the opcode address.

INDEXED, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. Thus, this addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this two-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This addressing mode can be used in a manner similar to indexed 8-bit offset except that this three-byte instruction allows tables to be anywhere in memory.

BIT SET/CLEAR

In the bit set clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte to which the specified bit is to be set or cleared. Thus, any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single two-byte instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit to be tested, and its condition (set or clear), is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage PC0 in Self-Check Mode All Other	V_{in}	0.3 to +15.0 −0.3 to +7.0	V
Port A and C Source Current per Pin (One at a Time)	I_{out}	10	mA
Operating Temperature Range MC68705S3S MC68705S3CS	T_A	 0 to 70 −40 to +85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature Cerdip	T_J	 175	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation, it is recommended the V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{CC}$. Reliability of operation is enhanced if unused inputs except EXTAL are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Cerdip	θ_{JA}	 60	°C/W

3

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications, $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

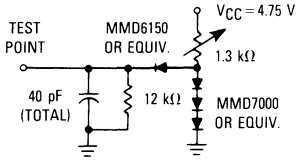


Figure 19. TTL Equivalent Test Load (Port B)

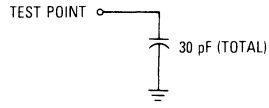


Figure 20. CMOS Equivalent Test Low (Port A)

ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
RESET Hysteresis Voltages "Out of Reset" "Into Reset"	 V_{IRES+} V_{IRES-}	1.3 1.5 0.8	— — —	2.0 2.5 1.8	V
Standby Supply Voltage @ V_{CC} = 0 V	V_{STBY}	4.0	—	V_{CC} + 0.7	V
Standby Current (V_{STBY} = 4.0 V)	I_{STBY}	—	1.0	5.0	mA
Power Dissipation — No Port Loading (V_{CC} = 5.75 V, T_A = 0°C) (V_{CC} = 5.75 V, T_A = −40°C)	P_D	 — —	 800 925	 1006 1092	mW
Low Voltage Recover	V_{LVR}	—	—	4.75	V
Low Voltage Inhibit	V_{LVI}	—	3.75	—	V
Input Current $\overline{INT}$ (V_{in} = 2.4 V to V_{CC}) EXTAL (V_{in} = 2.4 V to V_{CC} Crystal Option) (V_{in} = 0.4 V Crystal Option) $\overline{RESET}$ (V_{in} = 5.75 V)	I_{in}	 — — — —	 20 — — 2500	 50 10 −1600 3800	µA

PROGRAMMING OPERATION ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 20° to 30°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Programming Voltage	V_{PP}	20.0	21.0	22.0	V
V_{PP} Supply Current V_{PP} = 21.0 V	I_{PP}	 —	 —	 30	mA
Programming Oscillator Frequency	f_{oscp}	0.9	1.0	1.1	MHz
Bootstrap Programming Mode Voltage (PC0 Pin) (@ I_{IHTP} = 100 µA Max)	V_{IHTP}	9.0	12.0	15.0	V

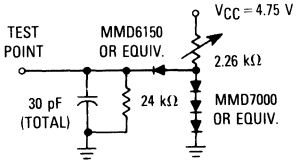


Figure 21. TTL Equivalent Test Load (Ports A and C)

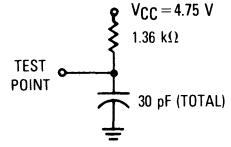


Figure 22. Open-Drain Equivalent Test Load (PB1, PB2, and PB3)

SWITCHING CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H), unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Oscillator Frequency	f_{osc}	0.4	—	4.2	MHz
Cycle time ($4/f_{osc}$)	t_{cyc}	0.95	—	10	µs
$\overline{INT}$, $\overline{INT2}$, and TIMER Pulse Width $\overline{RESET}$ Pulse Width	t_{WL}, t_{WH} t_{RWL}	t_{cyc} + 250	—	—	ns
$\overline{RESET}$ Delay Time (External Capacitance = 1 µF)	t_{RHL}	—	100	—	ns
$\overline{INT}$ Zero-Crossing Detection Input Frequency (for ±5% Accuracy)	f_{INT}	0.03	—	1	kHz
External Clock Input Duty Cycle (EXTAL)	—	40	50	60	%
Oscillator Startup Time Crystal	t_{su}	—	—	100	ms
SPICL High Time	t_{SPICLH}	4	—	—	t_{cyc}
SPICL Low Time	t_{SPICHL}	4	—	—	t_{cyc}
SPICL Rise and Fall Time	t_{Sr}, t_{Sf}	—	—	1	µs
SPID Input Data Setup Time	t_{SDS}	2	—	—	tcyc
SPID Input Data Hold Time	t_{SDH}	2	—	—	t_{cyc}
SPICL to SPISS Lag Time	t_{SStG}	4	—	—	t_{cyc}
SPISS to SPICL Lead Time	t_{SSLD}	4	—	—	t_{cyc}
Start Bit to First Clock Lead Time	t_{STL}	1	—	—	t_{cyc}
External Timer Input to Timer Change Time	t_{PCT}	3	—	—	t_{cyc}
Timer Change to Port B Toggle Time	t_{TPB}	2	—	—	t_{cyc}
$\overline{INT2}$ to Timer A Load Time	t_{INTL}	3	—	—	t_{cyc}

A/D CONVERTER CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Min	Typ	Max	Unit	Comments
Resolution	8	8	8	Bits	
Non-Linearity*	—	—	±1/2	LSB	After removing zero-offset and full-scale errors
Quantizing Error	—	—	±1/2	LSB	
Conversion Range V_{RH} V_{RL}	 — V_{SS}	 — —	V_{CC} 0.2	V	A/D accuracy may decrease proportionately as V_{RH}–V_{RL} is reduced below 4.0 V. The sum of V_{RH} and V_{RL} must not exceed V_{CC}
Conversion Time	30	30	30	t_{cyc}	Includes sampling time
Monotonicity	(Inherent within total error)				
Sample Time	5	5	5	t_{cyc}	
Sample/Hold Capacitance, Input	—	—	25	pF	
Analog Input Voltage	V_{RL}	—	V_{RH}	V	Transients on any analog lines are not allowed at any time during sampling or accuracy may be degraded

*For V_{RH} = 4.0 V to 5.0 V and V_{RL} = 0 V.

PORT ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Port B					
Output Low Voltage, I_{Load} = 3.2 mA	V_{OL}	—	—	0.4	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.0	V
Output High Voltage, I_{Load} = −200 μA	V_{OH}	2.4	8	—	V
Darlington Current Drive (Source)*, V_O = 1.5 V	I_{OH}	−1.0	—	−10	mA
Input High Voltage	V_{IH}	2.0	—	V_{CC} + 0.7	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port C and Port A					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC} + 0.7	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port D (Digital Inputs Only)					
Input High Voltage	V_{IH}	2.0	—	V_{CC} + 0.7	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Input Current**	I_{in}	—	<1	10	μA

*Not applicable if programmed to open-drain state.

**PD4/V_{RL} — PD5/V_{RH}.
The A/D conversion resistor (15 kΩ typical) is connected internally between these two lines, impacting their use as digital inputs in some applications.

ORDERING INFORMATION

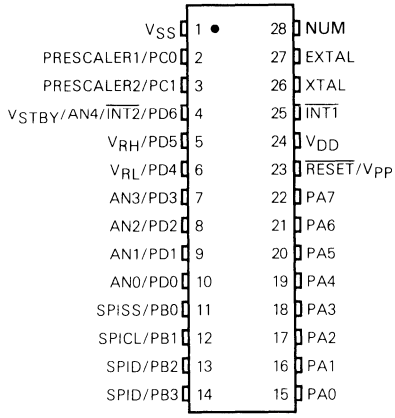
The following table provides generic information pertaining to the package type, temperature, and MC order numbers for the MC68705S3.

Table 5. Generic Information

Package Type	Temperature	Order Number
Cerdip (S Suffix)	0°C to 70°C −40°C to +85°C	MC68705S3S MC68705S3CS

MECHANICAL DATA

PIN ASSIGNMENTS



3

MC68705U3

Technical Summary

8-Bit EPROM Microcontroller Unit

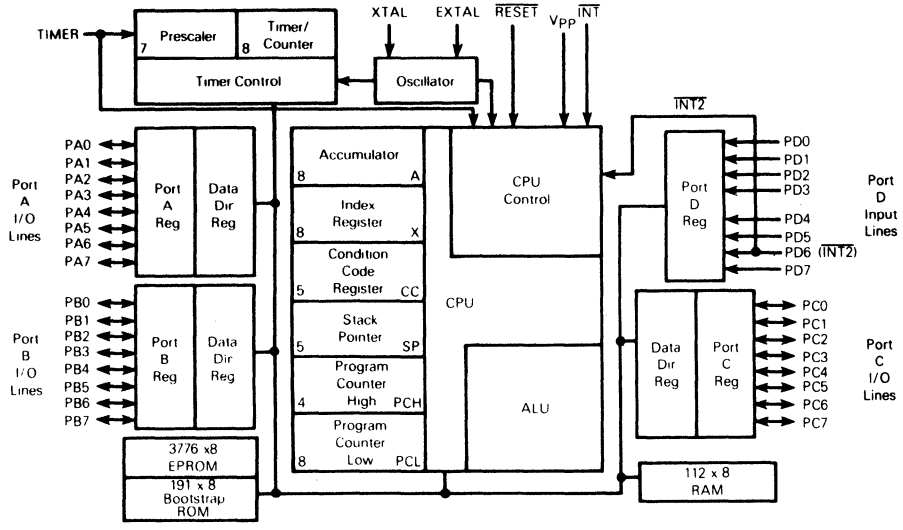
The MC68705U3 (HMOS) Microcontroller Unit (MCU) is an EPROM member of the MC6805 Family of microcontrollers. The user programmable EPROM allows program changes and lower volume applications. This low cost MCU has parallel I/O capability with pins programmable as input or output. This publication contains condensed information on the MCU; for detailed information, refer to *M6805 HMOS, M146805 CMOS Family User's Manual* (M6805UM(AD2)) or contact your local Motorola sales office.

Refer to the block diagram for the hardware features and to the list below for additional features available on the MCU.

- Internal 8-Bit Timer with 7-Bit Programmable Prescaler
- On-chip Oscillator
- Memory Mapped I/O
- Versatile Interrupt Handling
- Bit Manipulation
- Bit Test and Branch Instruction
- Vectored Interrupts
- Bootstrap Program in ROM
- 112 Bytes of RAM
- 3776 Bytes of EPROM
- 24 I/O Pins

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

V_{CC} AND V_{SS}

Power is supplied to the microcontroller using these two pins. V_{CC} is +5.25 volts (±0.5Δ) power, and V_{SS} is ground.

V_{PP}

This pin is used when programming the EPROM. In normal operation, this pin is connected to V_{CC}.

$\overline{INT}$

This pin provides the capability for asynchronously applying on external interrupt to the MCU. Refer to **INTERRUPTS** for more detailed information

EXTAL, XTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a resistor/capacitor combination, or an external signal (depending on mask option register setting) is connected to these pins to provide a system clock.

RC Oscillator

With this option, a resistor is connected to the oscillator pins as shown in Figure 1. The relationship between R and f_{osc} is shown in Figure 2.

Crystal

The circuit shown in Figure 1 is recommended when using a crystal. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time. Refer to **ELECTRICAL SPECIFICATIONS** for V_{CC} specifications.

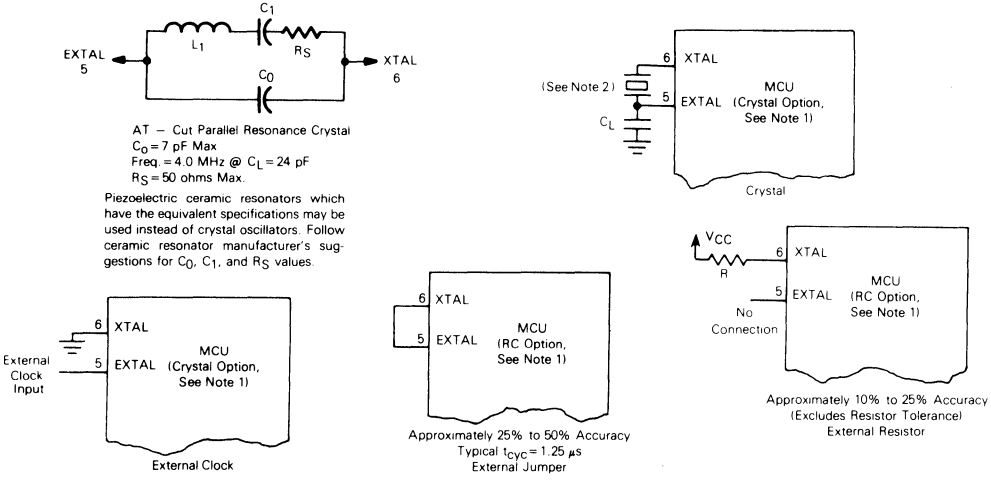
External Clock

An external clock should be applied to the EXTAL input with the XTAL input connected to V_{SS}, as shown in Figure 1. This option may only be used with the crystal oscillator option selected in the mask option register.

TIMER

This pin is used as an external input to control the internal timer/counter circuitry. This pin also detects a

3



NOTES:

1. For the MC68705U3 MOR b7 = 0 for the crystal option and MOR b7 = 1 for the RC option. When the TIMER input pin is in the V_{IHTP} range (in the bootstrap EPROM programming mode), the crystal option is forced. When the TIMER input is at or below V_{CC}, the clock generator option is determined by bit 7 of the mask option register (CLK).
2. The recommended C_L value with a 4.0 MHz crystal is 27 pF maximum, including system distributed capacitance. There is an internal capacitance of approximately 25 pF on the XTAL pin. For crystal frequencies other than 4 MHz, the total capacitance on each pin should be scaled as the inverse of the frequency ratio. For example, with a 2 MHz crystal, use approximately 50 pF on EXTAL and approximately 25 pF on XTAL. The exact value depends on the motional-arm parameters of the crystal used.

Figure 1. Oscillator Connections

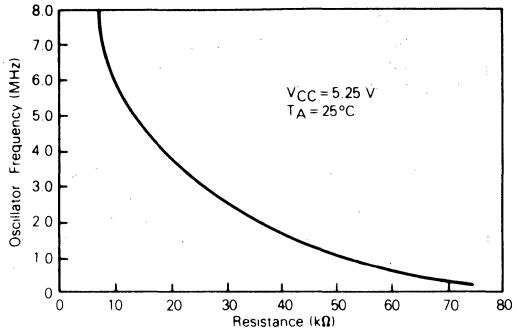


Figure 2. Typical Frequency vs Resistance for RC Oscillator Option Only

higher voltage level used to initiate the bootstrap program.

3

RESET

This pin has a Schmitt trigger input and an on-chip pullup. The MCU can be reset by pulling RESET low.

INPUT/OUTPUT LINES (PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD7)

These 32 lines are arranged into four 8-bit ports (A, B, C, and D). Ports A, B, and C are programmable as either inputs or outputs under software control of the data direction registers. Port D is a fixed input port. Port D bit 6 may be used for a second interrupt (INT2). Refer to **PROGRAMMING** for additional information.

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

Port A, B, and C pins are programmable as either input or output under software control of the corresponding data direction register (DDR). Port D is input only. The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic one for output and a logic zero for input. On reset, all the DDRs are initialized to a logic zero state to put the ports in the input mode. The port output registers are not initialized on reset and should be written to before setting the DDR bits.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This port write may be used to initialize the data registers and avoid undefined outputs. Care must be exercised when using read-modify-write instructions since the data read corresponds to the pin level if the DDR is an input (zero) and, also, corresponds to the latched output when the DDR is an output (one). Refer to Table 1 for I/O functions and to Figure 3 for typical port circuitry.

NOTE

Read-modify-write instructions should not be used when writing to the DDRs, because DDRs always read as 'one'.

Table 1. I/O Pin Functions

Data Direction Register Bit	Latched Output Data Bit	Output State	Input To MCU
1	0	0	0
1	1	1	1
0	X	Hi-Z**	Pin

**Ports B and C are three-state ports. Port A has an internal pullup devices to provide CMOS data drive capability.

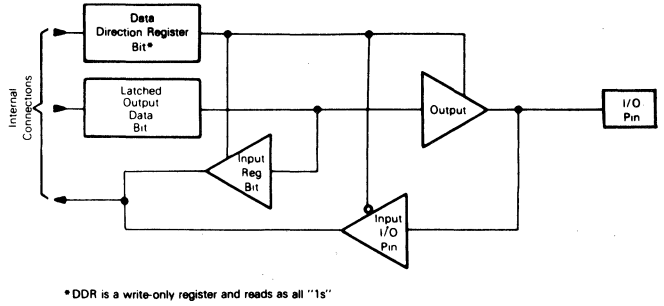


Figure 3. Typical Port I/O Circuitry and Register Configuration

MEMORY

The MCU is capable of addressing 4096 bytes of memory and I/O registers. The memory map is shown in Figure 4. The locations consist of user EPROM, bootstrap ROM, user RAM, a mask option register (MOR), a program control register, and I/O. The interrupt vectors are located from $FF8 to $FFF. The bootstrap is a mask-programmed ROM that allows the MCU to program its own EPROM.

The stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer decrements during pushes and increments during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

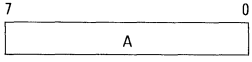
Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

REGISTERS

The MCU contains the registers described in the following paragraphs.

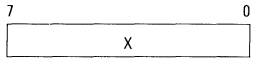
ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



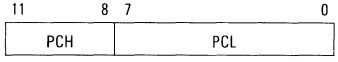
INDEX REGISTER (X)

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16- bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



PROGRAM COUNTER (PC)

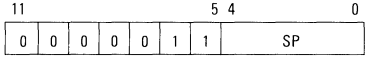
The program counter is a 12-bit register that contains the address of the next byte to be fetched.

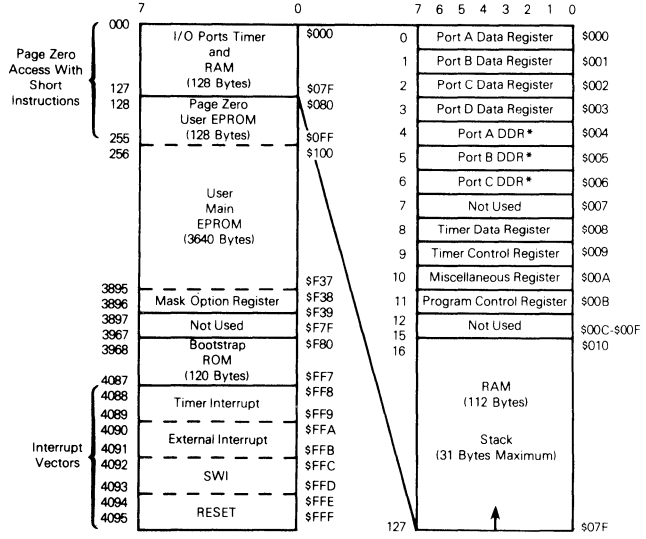


STACK POINTER (SP)

The stack pointer is a 12-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set at location $07F. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

The seven most-significant bits of the stack pointer are permanently set at 0000011. Subroutines and interrupts may be nested down to location $061 (31 bytes maximum), which allows the programmer to use up to 15 levels of subroutine calls (less if interrupts are allowed).



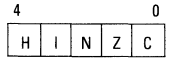


*Caution: Data direction registers (DDRs) are write-only; they read as $FF.

Figure 4. Memory Map

CONDITION CODE REGISTER (CC)

The condition code register is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program, and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.



Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, the timer and external interrupt is masked (disabled). If an external interrupt occurs while this bit is set, the interrupt is latched and is processed as soon as the interrupt bit is cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic 1).

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/Borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions, and during shifts and rotates.

RESETS

The MCU can be reset two ways: by initial power-up and by the external reset input ($\overline{\text{RESET}}$). The $\overline{\text{RESET}}$ input consists mainly of a Schmitt trigger that senses the line logic level.

POWER-ON-RESET (POR)

An internal reset is generated on power-up that allows the internal clock generator to stabilize. The power-on reset is used strictly for power turn-on voltatge. A delay of t_{RHL} milliseconds is required before allowing $\overline{\text{RESET}}$ input to go high. Connecting a capacitor to the $\overline{\text{RESET}}$ input (Figure 5) typically provides sufficient delay.

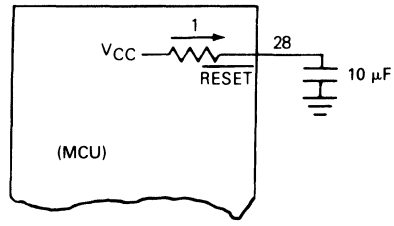


Figure 5. Power-Up $\overline{\text{RESET}}$ Delay Circuit

EXTERNAL $\overline{\text{RESET}}$ INPUT

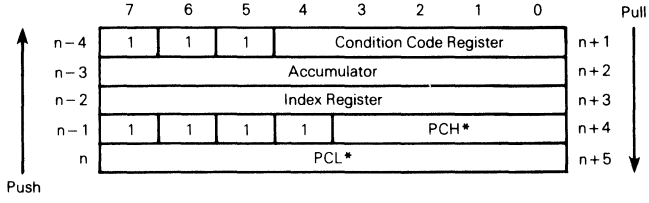
The MCU is reset when a logic zero is applied to the $\overline{\text{RESET}}$ input for a period longer than one machine cycle (t_{cyc}). Under this type of reset, the Schmitt trigger switches off at V_{IRES-} to provide an internal reset voltage.

INTERRUPTS

The MCU can be interrupted four different ways: (1) through the external interrupt $\overline{\text{INT}}$ input pin, (2) with the internal timer interrupt request, (3) using the software interrupt instruction (SWI), or (4) the external Port D (INT2) input pin.

Interrupts cause the processor registers to be saved on the stack and the interrupt mask (I bit) set to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack after which normal processing resumes. The stacking order is shown in Figure 6.

Unlike $\overline{\text{RESET}}$, hardware interrupts do not cause the current instruction execution to be halted but are considered pending until the current instruction is complete.



*For subroutine calls, only PCH and PCL are stacked.

Figure 6. Interrupt Stacking Order

NOTE

The current instruction is considered to be the one already fetched and being operated on.

When the current instruction is complete, the processor checks all pending hardward interrupts and, if unmasked (I bit clear), proceeds with interrupt processing; otherwise, the next instruction if fetched and executed. Masked interrupts are latched for later interrupt service. If the timer interrupt status bit is cleared before unmasking the interrupt, then the interrupt is not latched.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction regardless of the setting of the I bit. Refer to Figure 7 for the reset and interrupt instruction processing sequence.

TIMER INTERRUPT

If the time mask bit (TCR6) is cleared, then, each time the timer decrements to zero (transitions from $01 to $00), an interrupt request is generated. The actual processor interrupt is generated only if the interrupt mask bit of the condition code register (CCR) is also cleared. When the interrupt is recognized, the current state of the machine is pushed onto the stack and the I bit in the CCR is set, masking further interrupts until the present one is serviced. The contents of the timer interrupt vector, containing the location of the timer interrupt service routine, is then loaded into the program counter. At the end of the timer interrupt service routine, the software normally executes an RTI instruction which restores the machine state and starts executing the interrupted program. The timer interrupt status bit can only be cleared by software.

EXTERNAL INTERRUPT

The external interrupt is internally synchronized and then latched on the falling edge of INT and INT2. Clearing the I bit enables the external interrupt. The INT2 interrupt has an interrupt request bit (bit 7) and a mask bit (bit 6) in the miscellaneous register (MR). The INT2 interrupt is inhibited when the mask bit is set. The INT2 is always read as a digital input on port D. The INT2 and timer interrupt request bits, if set, cause the MCU to process an interrupt when the condition code I bit is clear. The

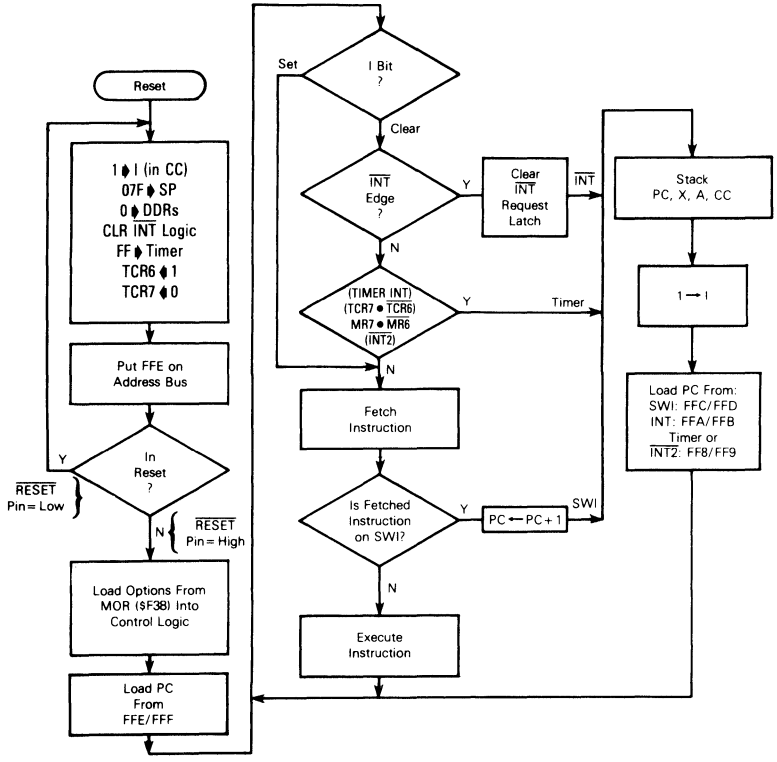


Figure 7. Reset and Interrupt Processing Flowchart

following paragraphs describe two typical external interrupt circuits.

Zero-Crossing Interrupt

A sinusoidal input signal (f_{INT} maximum) can be used to generate an external interrupt (see Figure 8a) for use as a zero-crossing detector (for negative transitions of the ac sinusoid). This type of circuit allows applications such as servicing time-of-day routines and engaging/disengaging ac power control devices. Off-chip, full-wave rectification provides an interrupt at every zero crossing of the ac signal and, thereby, provides a 2f clock.

Digital-Signal Interrupt

With this type of circuit (Figure 8b), the $\overline{INT}$ pin can be driven by a digital signal. The maximum frequency of a signal that can be recognized by the TIMER or $\overline{INT}$ pin logic is dependent on the parameter labeled t_{WL}, t_{WH}. Refer to **TIMER** for additional information.

3

SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. The SWI execution is similar to the hardware interrupts.

MODES OF OPERATION

The MCU has two modes of operations: normal and bootstrap. The following paragraphs describe these modes.

NORMAL MODE

This mode is a single-chip mode and is entered if the following conditions are met: (1) the $\overline{RESET}$ line is low, (2) the PC0 pin is within its normal operational range, and (3) the V_{PP} pin is connected to V_{SS}. The next rising edge of the $\overline{RESET}$ pin then causes the part to enter the normal mode.

BOOTSTRAP

The bootstrap mode is entered if the TIMER pin is equal to +12 V. For more information refer to application note, *MC68705P3/R3/U3 8-Bit EPROM Microcomputer Programming Module* (AN-857/D Rev. 2).

TIMER

The MCU consists of an 8-bit software programmable counter driven by a 7-bit software programmable prescaler. The various timer sources are made via the timer control register (TCR) and/or the mask option register (MOR). The 8-bit counter may be loaded under program control and is decremented toward zero. When the timer reaches zero, the timer interrupt request bit (bit 7) in the timer control register (TCR) is set. Refer to Figure 9 for timer block diagram.

The timer interrupt can be masked (disabled) by setting the timer interrupt mask bit (bit 6) in the TCR. When the I bit in the condition code register is cleared and the TCR bit 6 is cleared, the processor receives the interrupt. The MCU responds to this interrupt by (1) saving the present CPU state on the stack, (2) fetching the timer interrupt vector, and (3) executing the interrupt routine. The timer interrupt request bit must be cleared by software. Refer to **RESETS** and **INTERRUPTS** for additional information.

The prescaler is a 7-bit divider which is used to extend the maximum length of the timer. To avoid truncation errors, the prescaler is cleared when TCR bit 3 is set to a logic one; however, the TCR bit 3 always reads as a logic zero to ensure proper operation with read-modify-write instructions.

The timer continues to count past zero, falling from $00 through $FF, and continues the countdown. The counter can be read at any time by reading the timer data register (TDR). This allows a program to determine the length of time since a timer interrupt has occurred without disturbing the counting process. The TDR is unaffected by reset.

SOFTWARE CONTROLLED MODE

This mode is selected when TOPT (bit 6) in the MOR is programmed to zero. The timer prescaler input can be configured for three different operating modes plus a disable mode, depending on the value written to TCR control bits 4 and 5 (TIE and TIN). The following paragraphs describe the different modes.

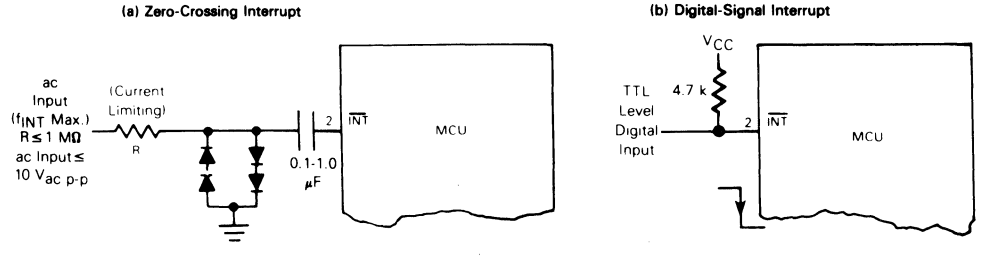


Figure 8. Typical Interrupt Circuits

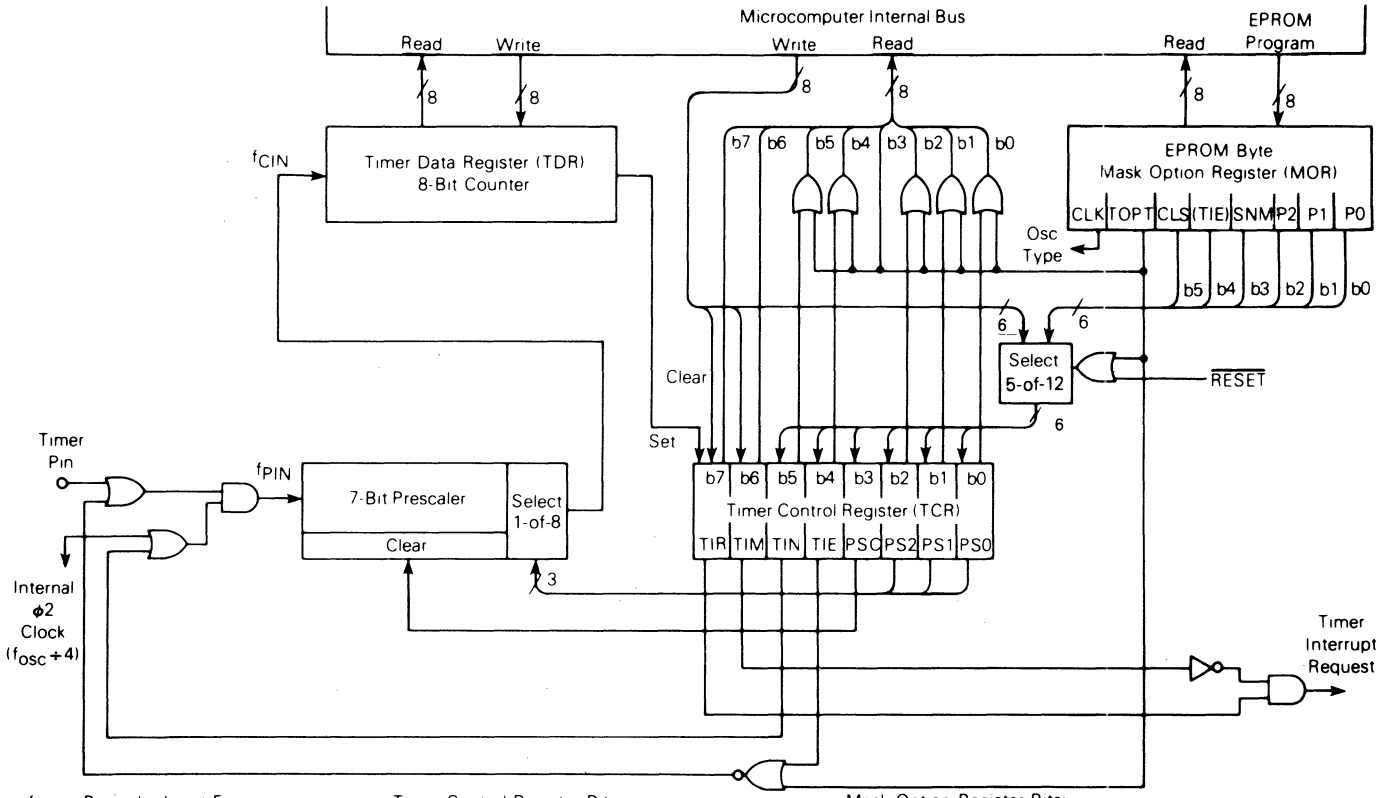


Figure 9. Timer Block Diagram

Timer Input Mode 1

When TIE and TIN are both programmed to zero, the timer input is from the internal clock (phase two) and the timer input pin is disabled. The internal clock mode can be used for periodic interrupt generation as well as a reference for frequency and event measurement.

Timer Input Mode 2

When TIE = 1 and TIN = 0, the internal clock and the timer input signals are ANDed to form the timer input. This mode can be used to measure external pulse widths. The active high, external pulse gates in the internal clock for the duration of the external pulse. The accuracy of the count is ±1.

Timer Input Mode 3

When TIE = 0 and TIN = 1, no prescaler input frequency is applied to the prescaler and the timer is disabled.

Timer Input Mode 4

When TIE and TIN are both one, the timer input is from the external clock. The external clock can be used to count external events as well as to provide an external frequency for generating periodic interrupts. Frequency of external input must be $\leq f_{OSC}/8$.

MOR CONTROLLED MODE

This mode is selected when TOPT (bit 6) in the MOR is programmed to logic one. The timer circuits are the same as described in **SOFTWARE CONTROLLED MODE**. The logic levels of TCR bits 0, 1, 2, and 5 are determined during EPROM programming by the same bits in the MOR. Therefore, bits 0, 1, 2, and 5 in the MOR control the prescaler division and the timer clock selection. TIE (bit 4) and PSC (bit 3) in the TCR are set to a logic one when in the MOR controlled mode. TIM (bit 6) and TIR (bit 7) are controlled by the counter and software.

TIMER CONTROL REGISTER (TCR) $009

This is an 8-bit register that controls various functions such as configuring operation mode, setting ratio of the prescaler, and generating timer interrupt request signal. All bits are read/write except bit 3. The configuration of the TCR is determined by the TOPT (bit 6) in the MOR. When TOPT = 1, the TCR emulates the MC6805U2; when TOPT = 0, the TCR is controlled by software.

TCR with MOR TOPT = 1

7	6	5	4	3	2	1	0
TIR	TIM	*	*	PSC	*	*	*

TCR with MOR TOPT = 0

7	6	5	4	3	2	1	0
TIR	TIM	TIN	TIE	PSC	PS2	PS1	PS0

RESET:

0	1	U	U	U	U	U	U

*The value of corresponding bits in MOR is written during $\overline{\text{RESET}}$ rising edge. These bits always read 'one'.

TIR — Timer Interrupt Request
Used to indicate the timer interrupt when it is logic one
1 = Set when the timer data register changes to all zeros
0 = Cleared by external reset, power-on reset, or under program control

TIM — Timer Interrupt Mask
Used to inhibit the timer interrupt
1 = Interrupt inhibited
0 = Interrupt enabled

TIN — External or Internal
Selects input clock source
1 = External clock selected
0 = Internal clock selected ($f_{OSC}/4$)

TIE — TIMER External Enable
Used to enable external TIMER pin. When TOPT = 1, TIE is always a logical "one".
1 = Enables external timer pin
0 = Disables external timer pin

PSC — Prescaler Clear
Write only bit. Writing a one to this bit resets the prescaler to zero. A read of this location always indicates a zero when TOPT = 0. When TOPT = 1, this bit will read a logical "one" and has no effect on the prescaler.

PS2, PS1, PS0 — Prescaler Clear
Decoded to select one of eight outputs of the prescaler

Prescaler

PS2	PS1	PS0	Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

NOTES

When changing the PS bits in software, the PSC bit should be written to a "one" in the same write cycle to clear the prescaler. Changing the PS bits without clearing the prescaler may cause prescaler truncation.

MASK OPTION REGISTER (MOR) $F38

The MOR is implemented in EPROM. This register contains all zeros prior to programming and is not affected by reset. The MOR bits are described in the following paragraphs.

7	6	5	4	3	2	1	0
CLK	TOPT	CLS			P2	P1	P0

CLK — Clock (oscillator type)
1 = Resistor Capacitor (RC)
0 = Crystal

TOPT — Timer Option
1 = MC6805U2 type timer/prescaler. All bits except 6 and 7, of the TCR are invisible to the user. Bits 5, 2, 1, and 0 of the MOR determine the equivalent MC6805U2 mask options.
0 = All TCR bits are implemented as a software programmable timer. The state of MOR bits 5, 4, 2, 1, and 0 sets the initial values of their respective TCR bits.

CLS — Timer/Prescaler Clock Source
1 = External TIMER pin
0 = Internal clock

Bit 4
Not used if TOPT = 1. Sets the initial value of TIE in the TCR if TOPT = 0.
1 = Not used
0 = Sets initial value of TIE in the TCR

Bit 3
Not used

P2, P1, P0
The logical levels of these bits, when decoded, select one of eight outputs on the timer prescaler.

Prescaler

P2	P1	P0	Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

PROGRAMMING CONTROL REGISTER (PCR) $00B

The PCR is an 8-bit register which provides the necessary control bits to program the EPROM. The bootstrap program manipulates the PCR when programming so the user need not be concerned with PCR in most applications.

7	6	5	4	3	2	1	0
1	1	1	1	1	$\overline{VPON}$	$\overline{PGE}$	$\overline{PLE}$
RESET:							
U	U	U	U	U	U	1	1

$\overline{PLE}$ — Programming Latch Enable
Controls address and data being latched into the EPROM. Set during reset, but may be cleared anytime.
1 = Read EPROM
0 = Latch address and data on EPROM

$\overline{PGE}$ — Program Enable
Enables programming of EPROM. Must be set when changing the address and data. Set during reset.
1 = Inhibit EPROM programming
0 = Enable EPROM programming (if $\overline{PLE}$ is low)

$\overline{VPON}$ — V_{PP} On
A read-only bit that indicates high voltage at the V_{PP} pin. When set to "one", disconnects $\overline{PGE}$ and $\overline{PLE}$ from the chip.
1 = No high voltage on V_{PP} pin
0 = High voltage on V_{PP} pin

NOTE

$\overline{VPON}$ being "zero" does not indicate that the V_{PP} level is correct for programming. It is used as a safety interlock for the user in the normal operating mode.

$\overline{VPON}$	$\overline{PGE}$	$\overline{PLE}$	Programming Conditions
0	0	0	Programming mode (program EPROM byte)
1	0	0	$\overline{PGE}$ and $\overline{PLE}$ disabled from system
0	1	0	Programming disabled (latch address and data in EPROM)
1	1	0	$\overline{PGE}$ and $\overline{PLE}$ disabled from system
0	0	1	Invalid state; $\overline{PGE} = 0$ if $\overline{PLE} = 0$
1	0	1	Invalid state; $\overline{PGE} = 0$ if $\overline{PLE} = 0$
0	1	1	"High voltage" on V_{PP}
1	1	1	$\overline{PGE}$ and $\overline{PLE}$ disabled from system (operating mode)

3

EPROM PROGRAMMING

ERASING THE EPROM

The EPROM can be erased by exposure to high-intensity ultraviolet (UV) light with a wavelength of 2537 angstroms. The recommended integrated dose (UV intensity × exposure time) is 25Ws/cm^2. The lamps should be used without software filters, and the MCU should be positioned about one inch from the UV tubes. Ultraviolet erasure clears all bits of the MCU EPROM to the "zero" state. Data then can be entered by programming "ones" into the desired bit locations.

PROGRAMMING

The MCU bootstrap program can be used to program the MCU EPROM. The alternate vectoring used to implement the self check is used to start execution of the bootstrap program.

A MCM2532 UV EPROM (other industry standard EPROMs may be used) must first be programmed with the same information that is to be transferred to the MCU EPROM. Refer to application note, *MC68705P3/R3/U3 8-bit EPROM Microcomputer Programming Module* (AN-857/D Rev.2) for schematic diagrams and instructions on programming the MCU EPROM.

EMULATION

The MC68705U3 emulates the MC6805U2 and MC6805U3 "exactly". The MC6805U2 and MC6805U3 mask features are implemented in the mask option register EPROM byte. The following identify the few minor exceptions to the exactness of the emulation.

1. The MC6805U2 "future ROM" areas are implemented in the MC68705U3 and these 1728 bytes

must be left unprogrammed to accurately simulate the MC6805U2.
2. The reserved ROM areas have different data stored in them. In the MC6805U2 this area is used for self check, and in the MC68705U3 this area is used for the bootstrap program.
3. The MC6805U2 reads all ones in the 48 byte "future RAM" area. This area is not implemented on the MC6805U2/U3 mask ROM version but is implemented on the MC68705U3.
4. The MC68705U3 V_{PP} (pin 7) line is tied to V_{CC} during normal operations. On MC6805U2, this pin is grounded during normal operations, and on the MC6805U3, this pin is not connected.

INSTRUCTION SET

The MCU has a set of 59 basic instructions which can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

3

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following instruction listing.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (2's Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two-byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Higher	BHI
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
(Branch if Higher or Same)	(BHS)
Branch if Carry Set	BCS
(Branch if Lower)	(BLO)
Branch if Not Equal	BNE
Branch if Equal	BEQ
Branch if Half Carry Clear	BHCC
Branch if Half Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Bit is Clear	BMC
Branch if Interrupt Mask Bit is Set	BMS
Branch if Interrupt Line is Low	BIL
Branch if Interrupt Line is High	BIH
Branch to Subroutine	BSR

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the first 256 bytes of the memory space where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within

these 256 locations. The bit set, bit clear and bit test, and branch functions are all implemented with a single instruction. For test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch if Bit n is Set	BRSET n (n=0 . . . 7)
Branch if Bit n is Clear	BRCLR n (n=0 . . . 7)
Set Bit n	BSET n (n=0 . . . 7)
Clear Bit n	BCLR n (n=0 . . . 7)

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No-Operation	NOP

OPCODE MAP SUMMARY

Table 3 is an opcode map for the instructions used on the MCU.

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single-byte instructions, while the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. Two-byte direct-addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction.

RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −126 to +129 from the opcode address.

INDEXED, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. Thus, this addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this two-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This addressing mode can be used in a manner similar to indexed, 8-bit offset except that this three-byte instruction allows tables to be anywhere in memory.

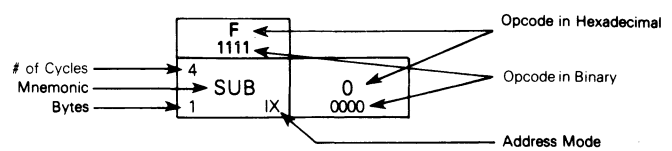
Table 3. Opcode Map

	Bit Manipulation		Branch	Read-Modify-Write					Control		Register/Memory						
	BTB	BSC	REL	DIR	INH	INH	IX1	IX	INH	INH	IMM	DIR	EXT	IX2	IX1	IX	
Hi / Low	0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111	8 1000	9 1001	A 1010	B 1011	C 1100	D 1101	E 1110	F 1111	Hi / Low
0 0000	10 BRSET0 3 BTB	7 BSET0 2 BSC	4 BRA 2 REL	6 NEG 2 DIR	4 NEG 1 INH	4 NEG 1 INH	7 NEG 2 IX1	6 NEG 1 IX	9 RTI 1 INH		2 SUB 2 IMM	4 SUB 2 DIR	5 SUB 3 EXT	6 SUB 3 IX2	5 SUB 2 IX1	4 SUB 1 IX	0 0000
1 0001	10 BRCLR0 3 BTB	7 BCLR0 2 BSC	4 BRN 2 REL						6 RTS 1 INH		2 CMP 2 IMM	4 CMP 2 DIR	5 CMP 3 EXT	6 CMP 3 IX2	5 CMP 2 IX1	4 CMP 1 IX	1 0001
2 0010	10 BRSET1 3 BTB	7 BSET1 2 BSC	4 BHI 2 REL								2 SBC 2 IMM	4 SBC 2 DIR	5 SBC 3 EXT	6 SBC 3 IX2	5 SBC 2 IX1	4 SBC 1 IX	2 0010
3 0011	10 BRCLR1 3 BTB	7 BCLR1 2 BSC	4 BLS 2 REL	6 COM 2 DIR	4 COMA 1 INH	4 COMX 1 INH	7 COM 2 IX1	6 COM 1 IX	11 SWI 1 INH		2 CPX 2 IMM	4 CPX 2 DIR	5 CPX 3 EXT	6 CPX 3 IX2	5 CPX 2 IX1	4 CPX 1 IX	3 0011
4 0100	10 BRSET2 3 BTB	7 BSET2 2 BSC	4 BCC 2 REL	6 LSR 2 DTR	4 LSRA 1 INH	4 LSRX 1 INH	7 LSR 2 IX1	6 LSR 1 IX			2 AND 2 IMM	4 AND 2 DIR	5 AND 3 EXT	6 AND 3 IX2	5 AND 2 IX1	4 AND 1 IX	4 0100
5 0101	10 BRCLR2 3 BTB	7 BCLR2 2 BSC	4 BCS 2 REL								2 BIT 2 IMM	4 BIT 2 DIR	5 BIT 3 EXT	6 BIT 3 IX2	5 BIT 2 IX1	4 BIT 1 IX	5 0101
6 0110	10 BRSET3 3 BTB	7 BSET3 2 BSC	4 BNE 2 REL	6 ROR 2 DIR	4 RORA 1 INH	4 RORX 1 INH	7 ROR 2 IX1	6 ROR 1 IX			2 LDA 2 IMM	4 LDA 2 DIR	5 LDA 3 EXT	6 LDA 3 IX2	5 LDA 2 IX1	4 LDA 1 IX	6 0110
7 0111	10 BRCLR3 3 BTB	7 BCLR3 2 BSC	4 BEQ 2 REL	6 ASR 2 DIR	4 ASRA 1 INH	4 ASRX 1 INH	7 ASR 2 IX1	6 ASR 1 IX		2 TAX 1 INH		5 STA 2 DIR	6 STA 3 EXT	7 STA 3 IX2	6 STA 2 IX1	5 STA 1 IX	7 0111
8 1000	10 BRSET4 3 BTB	7 BSET4 2 BSC	4 BHCC 2 REL	6 LSL 2 DIR	4 LSLA 1 INH	4 LSLX 1 INH	7 LSL 2 IX1	6 LSL 1 IX		2 CLC 1 INH	2 EOR 2 IMM	4 EOR 2 DIR	5 EOR 3 EXT	6 EOR 3 IX2	5 EOR 2 IX1	4 EOR 1 IX	8 1000
9 1001	10 BRCLR4 3 BTB	7 BCLR4 2 BSC	4 BHCS 2 REL	6 ROL 2 DIR	4 ROLA 1 INH	4 ROLX 1 INH	7 ROL 2 IX1	6 ROL 1 IX		2 SEC 1 INH	2 ADC 2 IMM	4 ADC 2 DIR	5 ADC 3 EXT	6 ADC 3 IX2	5 ADC 2 IX1	4 ADC 1 IX	9 1001
A 1010	10 BRSET5 3 BTB	7 BSET5 2 BSC	4 BPL 2 REL	6 DEC 2 DIR	4 DECA 1 INH	4 DECX 1 INH	7 DEC 2 IX1	6 DEC 1 IX		2 CLI 1 INH	2 ORA 2 IMM	4 ORA 2 DIR	5 ORA 3 EXT	6 ORA 3 IX2	5 ORA 2 IX1	4 ORA 1 IX	A 1010
B 1011	10 BRCLR5 3 BTB	7 BCLR5 2 BSC	4 BMI 2 REL							2 SEI 1 INH	2 ADD 2 IMM	4 ADD 2 DIR	5 ADD 3 EXT	6 ADD 3 IX2	5 ADD 2 IX1	4 ADD 1 IX	B 1011
C 1100	10 BRSET6 3 BTB	7 BSET6 2 BSC	4 BMC 2 REL	6 INC 2 DIR	4 INCA 1 INH	4 INCX 1 INH	7 INC 2 IX1	6 INC 1 IX		2 RSP 1 INH		3 JMP 2 DIR	4 JMP 3 EXT	5 JMP 3 IX2	4 JMP 2 IX1	3 JMP 1 IX	C 1100
D 1101	10 BRCLR6 3 BTB	7 BCLR6 2 BSC	4 BMS 2 REL	6 TST 2 DIR	4 TSTA 1 INH	4 TSTX 1 INH	7 TST 2 IX1	6 TST 1 IX		2 NOP 1 INH	8 BSR 2 REL	7 JSR 2 DIR	8 JSR 3 EXT	9 JSR 3 IX2	8 JSR 2 IX1	7 JSR 1 IX	D 1101
E 1110	10 BRSET7 3 BTB	7 BSET7 2 BSC	4 BIL 2 REL								2 LDX 2 IMM	4 LDX 2 DIR	5 LDX 3 EXT	6 LDX 3 IX2	5 LDX 2 IX1	4 LDX 1 IX	E 1110
F 1111	10 BRCLR7 3 BTB	7 BCLR7 2 BSC	4 BIH 2 REL	6 CLR 2 DIR	4 CLRA 1 INH	4 CLRX 1 INH	7 CLR 2 IX1	6 CLR 1 IX		2 TXA 1 INH		5 STX 2 DIR	6 STX 3 EXT	7 STX 3 IX2	6 STX 2 IX1	5 STX 1 IX	F 1111

Abbreviations for Address Modes

INH	Inherent
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte to which the specified bit is to be set or cleared. Thus, any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single two-byte instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit to be tested, and its condition (set or clear), is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

3

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage EPROM Programming Voltage (Vpp Pin) TIMER Pin — Normal Mode TIMER Pin — Bootstrap Programming Mode All Others	 V_{PP} V_{in} V_{in} V_{in}	 −0.3 to +22.0 −0.3 to +7.0 −0.3 to +15.0 −0.3 to +7.0	V
Operating Temperature Range MC68705U3 MC68705U3C	T_A	T_L to T_H 0 to +70 −40 to +85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature Cerdip	T_J	 175	°C/W

These devices contain circuity to protect the inputs against damage due to high static voltages or electrical fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high-impedance circuit. For proper operation, it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ and $V_{out}) \leq V_{CC}$. Reliability of operation is enhanced if unused inputs except EXTAL are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Cerdip	 θ_{JA}	 60	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A

PROGRAMMING OPERATION ELECTRICAL CHARACTERISTICS

(V_{DD} = 5.25 Vdc ±0.5%, V_{SS} = 0 Vdc, T_A = 20 to 30°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Programming Voltage	V_{PP}	20.0	21.0	22.0	V
V_{PP} Supply Current V_{PP} = 5.25 V V_{PP} = 21.0 V	I_{PP}	 — —	 — —	 8 30	mA
Programming Oscillator Frequency	f_{oscp}	0.9	1.0	1.1	MHz
Bootstrap Programming Mode Voltage (TIMER Pin) @ I_{IHTP} = 100 µA Maximum	V_{IHTP}	9.0	12.0	15.0	V

ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage $\overline{RESET}$ (4.99 ≤ V_{CC} ≤ 5.51) (V_{CC} < 4.75) $\overline{INT}$ 4.99 ≤ V_{CC} ≤ 5.51) (V_{CC} < 4.75) All Other	V_{IH}	 4.0 V_{CC} − 0.5 4.0 V_{CC} − 0.5 2.0	 — — ** ** —	 V_{CC} V_{CC} V_{CC} V_{CC} V_{CC}	V
Input High Voltage (TIMER Pin) Timer Mode Bootstrap Programming Mode	V_{IH}	 2.0 9.0	 — 12.0	 V_{CC} + 1.0 15.0	V
Input Low Voltage $\overline{RESET}$ $\overline{INT}$ All Other	V_{IL}	 V_{SS} V_{SS} V_{SS}	 — ** —	 0.8 1.5 0.8	V
Internal Power Dissipation (No Port Loading, V_{CC} = 5.25 V for Steady-State Operation) T_A = 0°C T_A = −40°C	P_{INT}	 —	520 580	740 800	mW
Input Capacitance EXTAL All Other	C_{in}	 — —	 25 10	 — —	pF
$\overline{INT}$ Zero-Crossing Input Voltage — Through a Capacitor	V_{INT}	2.0	—	4.0	$V_{ac\ p\text{-}p}$
$\overline{RESET}$ Hysteresis Voltage Out of Reset Voltage Into Reset Voltage	 V_{IRES+} V_{IRES-}	 2.1 0.8	 — —	 4.0 2.0	V
Programming Voltage (V_{PP} Pin) Programming EPROM Operating Mode	V_{PP}*	 20.0 4.75	 21.0 V_{CC}	 22.0 5.75	V
Input Current $\overline{TIMER}$ (V_{in} = 0.4 V) $\overline{INT}$ (V_{in} = 0.4 V) EXTAL (V_{in} = 2.4 V to V_{CC} Crystal Option) (V_{in} = 0.4 V Crystal Option) $\overline{RESET}$ (V_{in} = 0.8 V) (External Capacitor Changing Current)	 I_{in} I_{RES}	 — — — — −4.0	 — 20 — — —	 20 50 10 −1600 −40	µA

*V_{PP} (pin 7) is connected to V_{CC} in the normal operating mode.
**Due to internal biasing, this input (when not used) floats to approximately 2.0 V.

SWITCHING CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 V, V_{SS} = 0 Vdc, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Oscillator Frequency Normal	f_{osc}	0.4	—	4.2	MHz
Instruction Cycle Time ($4/f_{osc}$)	t_{cyc}	0.950	—	10	μs
$\overline{INT}$, $\overline{INT2}$, or Timer Pulse Width	t_{WL}, t_{WH}	t_{cyc} + 250	—	—	ns
$\overline{RESET}$ Pulse Width	t_{RWL}	t_{cyc} + 250	—	—	ns
$\overline{RESET}$ Delay Time (External Cap = 1.0 μF)	t_{RHL}	100	—	—	ms
$\overline{INT}$ Zero Crossing Detection Input Frequency	f_{INT}	0.03	—	1.0	kHz
External Clock Duty Cycle (EXTAL)	—	40	50	60	%
Crystal Oscillator Start-Up Time	—	—	—	100	ms

PORT ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0° to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Port A					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Output High Voltage, I_{Load} = −10 μA	V_{OH}	V_{CC} − 1.0	—	—	V
Input High Voltage, I_{Load} = −300 μA (Max)	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage, I_{Load} = −500 μA (Max)	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current (V_{in} = 2.0 V to V_{CC})	I_{IH}	—	—	−300	μA
Hi-Z State Input Current (V_{in} = 0.4 V)	I_{IL}	—	—	−500	μA
Port B					
Output Low Voltage, I_{Load} = 3.2 mA	V_{OL}	—	—	0.4	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.0	V
Output High Voltage, I_{Load} = −200 μA	V_{OH}	2.4	—	—	V
Darlington Current Drive (Source), V_O = 1.5 V	I_{OH}	−1.0	—	−10	mA
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port C					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port D (Input Only)					
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Input Current	I_{in}	—	<1	5	μA

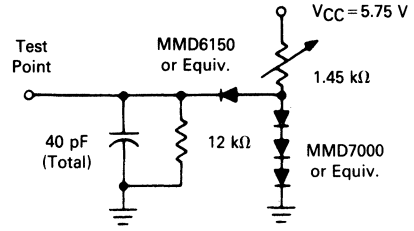


Figure 10. TTL Equivalent Test Load (Port B)

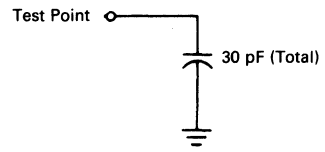


Figure 11. CMOS Equivalent Test Load (Port A)

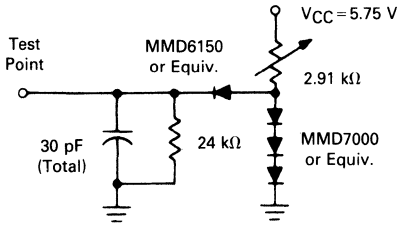


Figure 12. TTL Equivalent Test Load (Ports A and C)

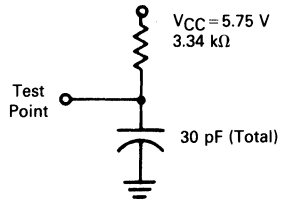


Figure 13. Open-Drain Equivalent Test Load (Port C)

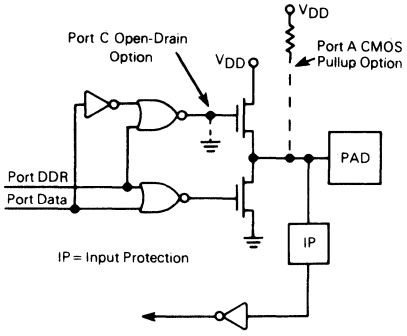


Figure 14. Ports A and C Logic Diagram

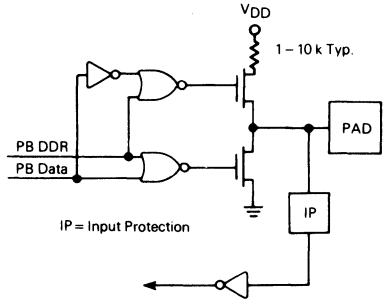


Figure 15. Port B Logic Diagram

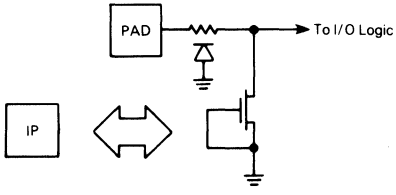


Figure 16. Typical Input Protection

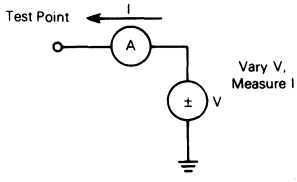


Figure 17. I/O Characteristic Measurement Circuit

ORDERING INFORMATION

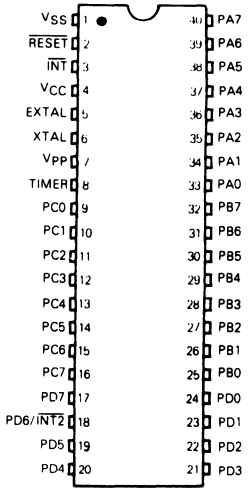
The following table provides generic information pertaining to the package type, temperature, and MC order numbers for the MC68705U3.

Table 3. Generic Information

Package Type	Temperature	Order Number
Cerdip S Suffix	0° to 70°C −40° to +85°C	MC68705U3S MC68705U3CS

MECHANICAL DATA

PIN ASSIGNMENTS



MC68705U5

Technical Summary
8-Bit EPROM Microcontroller Unit

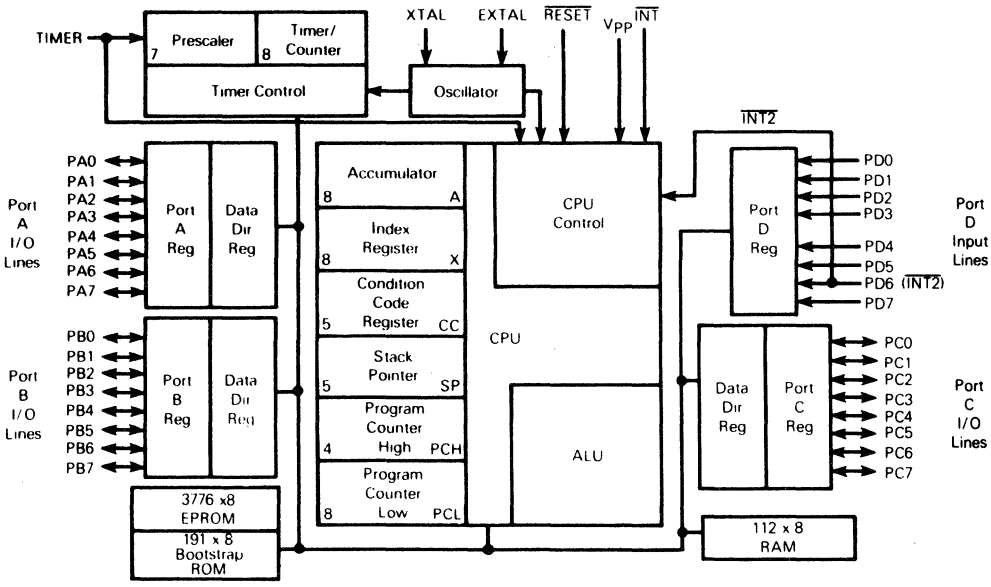
The MC68705U5 (HMOS) Microcontroller Unit (MCU) is an EPROM member of the MC6805 Family of microcontrollers. The user programmable EPROM allows program changes and lower volume applications. This low cost MCU has parallel I/O capability with pins programmable as input or output. This publication contains condensed information on the MCU; for detailed information, refer to *M6805 HMOS, M146805 CMOS Family User's Manual* (M6805UM(AD2)) or contact your local Motorola sales office.

Refer to the block diagram for the hardware features and to the list below for additional features available on the MCU.

- Internal 8-Bit Timer with 7-Bit Programmable Prescaler
- On-chip Oscillator
- Memory Mapped I/O
- Versatile Interrupt Handling
- Bit Manipulation
- Bit Test and Branch Instruction
- Vectored Interrupts
- Bootstrap Program in ROM
- 3776 Bytes of EPROM
- 112 Bytes of RAM
- 24 I/O Pins
- EPROM Security Feature

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

V_{CC} AND V_{SS}

Power is supplied to the microcontroller using these two pins. V_{CC} is +5.25 volts (±0.5Δ) power, and V_{SS} is ground.

V_{PP}

This pin is used when programming the EPROM. In normal operation, this pin is connected to V_{CC}.

$\overline{INT}$

This pin provides the capability for asynchronously applying on external interrupt to the MCU. Refer to **INTERRUPTS** for more detailed information.

EXTAL, XTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a resistor/capacitor combination, or an external signal (depending on mask option register setting) is connected to these pins to provide a system clock.

RC Oscillator

With this option, a resistor is connected to the oscillator pins as shown in Figure 1. The relationship between R and f_{osc} is shown in Figure 2.

Crystal

The circuit shown in Figure 1 is recommended when using a crystal. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time. Refer to **ELECTRICAL SPECIFICATIONS** for V_{CC} specifications.

External Clock

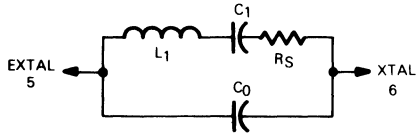
An external clock should be applied to the EXTAL input with the XTAL input connected to V_{SS}, as shown in Figure 1. This option may only be used with the crystal oscillator option selected in the mask option register.

TIMER

This pin is used as an external input to control the internal timer/counter circuitry. This pin also detects a higher voltage level used to initiate the bootstrap program.

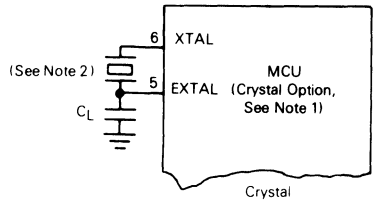
$\overline{RESET}$

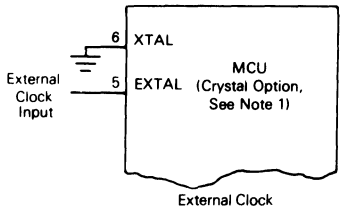
This pin has a Schmitt trigger input and an on-chip pullup. The MCU can be reset by pulling $\overline{RESET}$ low.

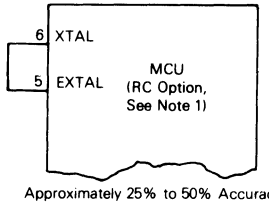


AT – Cut Parallel Resonance Crystal
C_0 = 7 pF Max
Freq. = 4.0 MHz @ C_L = 24 pF
R_S = 50 ohms Max.

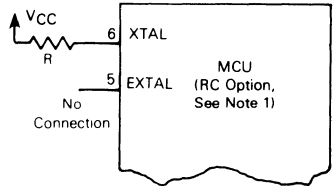
Piezoelectric ceramic resonators which have the equivalent specifications may be used instead of crystal oscillators. Follow ceramic resonator manufacturer's suggestions for C_0, C_1, and R_S values.







Approximately 25% to 50% Accuracy
Typical t_{CYC} = 1.25 μs
External Jumper



Approximately 10% to 25% Accuracy
(Excludes Resistor Tolerance)
External Resistor

NOTES:
1. For the MC68705U5 MOR b7 = 0 for the crystal option and MOR b7 = 1 for the RC option. When the TIMER input pin is in the V_{IHTP} range (in the bootstrap EPROM programming mode), the crystal option is forced. When the TIMER input is at or below V_{CC}, the clock generator option is determined by bit 7 of the Mask Option Register (CLK).
2. The recommended C_L value with a 4.0 MHz crystal is 27 pF maximum, including system distributed capacitance. There is an internal capacitance of approximately 25 pF on the XTAL pin. For crystal frequencies other than 4 MHz, the total capacitance on each pin should be scaled as the inverse of the frequency ratio. For example, with a 2 MHz crystal, use approximately 50 pF on EXTAL and approximately 25 pF on XTAL. The exact value depends on the motional-arm parameters of the crystal used.

Figure 1. Oscillator Connections

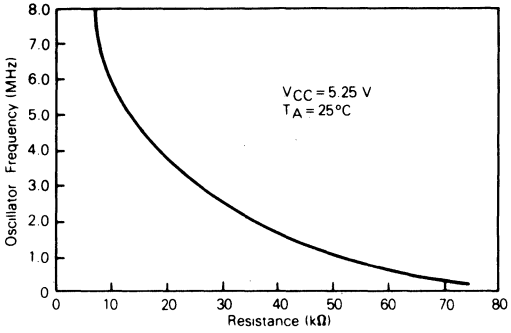


Figure 2. Typical Frequency vs Resistance for RC Oscillator Option Only

3

INPUT/OUTPUT LINES (PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD7)

These 32 lines are arranged into four 8-bit ports (A, B, C, and D). Ports A, B, and C are programmable as either inputs or outputs under software control of the data direction registers. Port D is a fixed input port and is not controlled by any data register. Port D bit 6 may be used for a second interrupt ($\overline{INT2}$). Refer to **PROGRAMMING** for additional information.

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

Ports A, B, and C are programmable as either input or output under software control of the corresponding data direction register (DDR). Port D is input only. The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic one for output and a logic zero for input. On reset, all the DDRs are initialized to a logic zero state to put the ports in the input mode. The port output registers are not initialized on reset and should be written to before setting the DDR bits.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This port write may be used to initialize the data registers and avoid undefined outputs. Care must be exercised when using read-modify-write instructions since the data read corresponds to the pin level if the DDR is an input (zero) and, also, to the latched output when the DDR is an output (1). Refer to Table 1 for I/O functions and to Figure 3 for typical port circuitry.

NOTE

Read-modify-write instructions should not be used when writing to the DDR since DDRs always read as 'one'.

Table 1. I/O Pin Functions

Data Direction Register Bit	Latched Output Data Bit	Output State	Input To MCU
1	0	0	0
1	1	1	1
0	X	Hi-Z**	Pin

**Ports B and C are three-state ports. Port A has an internal pullup devices to provide CMOS data drive capability.

MEMORY

The MCU is capable of addressing 4096 bytes of memory and I/O registers. The memory map is shown in Figure 4. The locations consist of user EPROM, bootstrap ROM, user RAM, a mask option register (MOR), a program control register, and I/O. The interrupt vectors are located from $FF8 to $FFF. The bootstrap is a mask-programmed ROM that allows the MCU to program its own EPROM.

The stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer

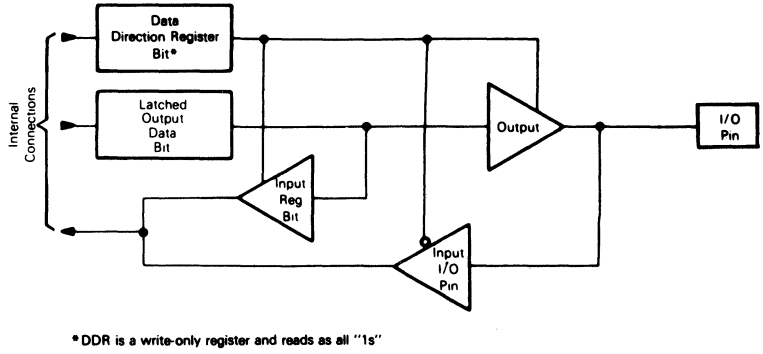


Figure 3. Typical Port I/O Circuitry and Register Configuration

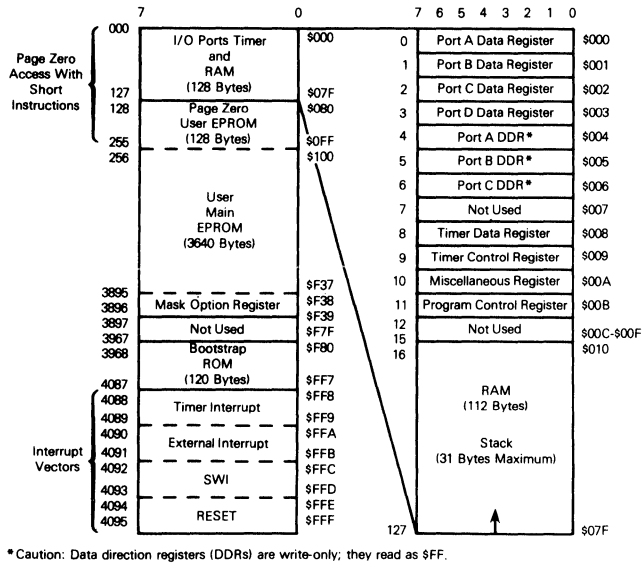


Figure 4. Memory Map

decrements during pushes and increments during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

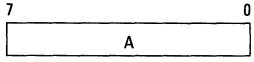
Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

REGISTERS

The MCU contains the registers described in the following paragraphs.

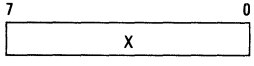
ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



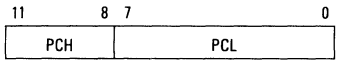
INDEX REGISTER (X)

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16- bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



PROGRAM COUNTER (PC)

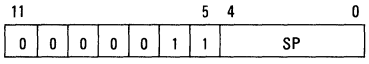
The program counter is a 12-bit register that contains the address of the next byte to be fetched.



STACK POINTER (SP)

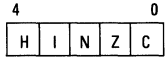
The stack pointer is a 12-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set at location $07F. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

The seven most-significant bits of the stack pointer are permanently set at 0000011. Subroutines and interrupts may be nested down to location $061 (31 bytes maximum), which allows the programmer to use up to 15 levels of subroutine calls (less if interrupts are allowed).



CONDITION CODE REGISTER (CC)

The condition code register is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program, and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.



Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, the timer and external interrupt is masked (disabled). If an external interrupt occurs while this bit is set, the interrupt is latched and is processed as soon as the interrupt bit is cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic one).

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/Borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occured during the last arithmetic operation. This bit is also affected during bit test and branch instructions, and during shifts and rotates.

RESETS

The MCU can be reset two ways: by initial power-up and by the external reset input ($\overline{\text{RESET}}$). The $\overline{\text{RESET}}$ input consists mainly of a Schmitt trigger that senses the line logic level.

POWER-ON-RESET (POR)

An internal reset is generated on power-up that allows the internal clock generator to stabilize. The power-on reset is used strictly for power turn-on voltatge. A delay of t_{RHL} milliseconds is required before allowing $\overline{\text{RESET}}$ input to go high. Connecting a capacitor to the $\overline{\text{RESET}}$ input (Figure 5) typically provides sufficient delay.

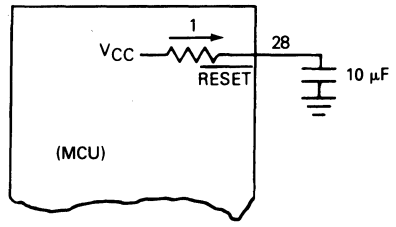


Figure 5. Power-Up $\overline{\text{RESET}}$ Delay Circuit

EXTERNAL $\overline{\text{RESET}}$ INPUT

The MCU is reset when a logic zero is applied to the $\overline{\text{RESET}}$ input for a period longer than one machine cycle (t_{cyc}). Under this type of reset, the Schmitt trigger switches off at V_{IRES-} to provide an internal reset voltage.

INTERRUPTS

The MCU can be interrupted four different ways: (1) through the external interrupt $\overline{\text{INT}}$ input pin, (2) with the internal timer interrupt request, (3) using the software interrupt instruction (SWI), or (4) the external Port D ($\overline{\text{INT2}}$) input pin.

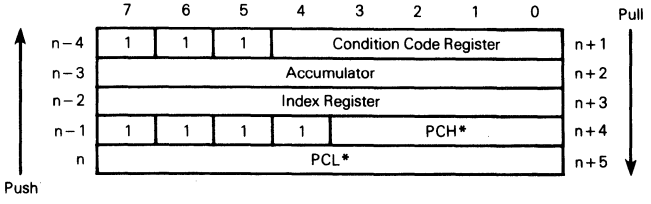
Interrupts cause the processor registers to be saved on the stack and the interrupt mask (I bit) set to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack after which normal processing resumes. The stacking order is shown in Figure 6.

Unlike $\overline{\text{RESET}}$, hardware interrupts do not cause the current instruction execution to be halted but are considered pending until the current instruction is complete.

NOTE

The current instruction is considered to be the one already fetched and being operated on.

When the current instruction is complete, the processor checks all pending hardward interrupts and, if unmasked



*For subroutine calls, only PCH and PCL are stacked.

Figure 6. Interrupt Stacking Order

(I bit clear), proceeds with interrupt processing; otherwise, the next instruction if fetched and executed. Masked interrupts are latched for later interrupt service. If the timer interrupt status bit is cleared before unmasking the interrupt, then the interrupt is not latched.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction regardless of the setting of the I bit. Refer to Figure 7 for the reset and interrupt instruction processing sequence.

TIMER INTERRUPT

If the time mask bit (TCR6) is cleared, then, each time the timer decrements to zero (transitions from $01 to $00), an interrupt request is generated. The actual processor interrupt is generated only if the interrupt mask bit of the condition code register (CCR) is also cleared. When the interrupt is recognized, the current state of the machine is pushed onto the stack and the I bit in the CCR is set, masking further interrupts until the present one is serviced. The contents of the timer interrupt vector, containing the location of the timer interrupt service routine, is then loaded into the program counter. At the end of the timer interrupt service routine, the software normally executes an RTI instruction which restores the machine state and starts executing the interrupted program. The timer interrupt status bit can only be cleared by software.

EXTERNAL INTERRUPT

The external interrupt is internally synchronized and then latched on the falling edge of $\overline{INT}$ and $\overline{INT2}$. Clearing the I bit enables the external interrupt. The $\overline{INT2}$ interrupt has an interrupt request bit (bit 7) and a mask bit (bit 6) in the miscellaneous register (MR). The $\overline{INT2}$ interrupt is inhibited when the mask bit is set. The $\overline{INT2}$ is always read as a digital input on port D. The $\overline{INT2}$ and timer interrupt request bits, if set, cause the MCU to process an interrupt when the condition code I bit is clear. The following paragraphs describe two typical external interrupt circuits.

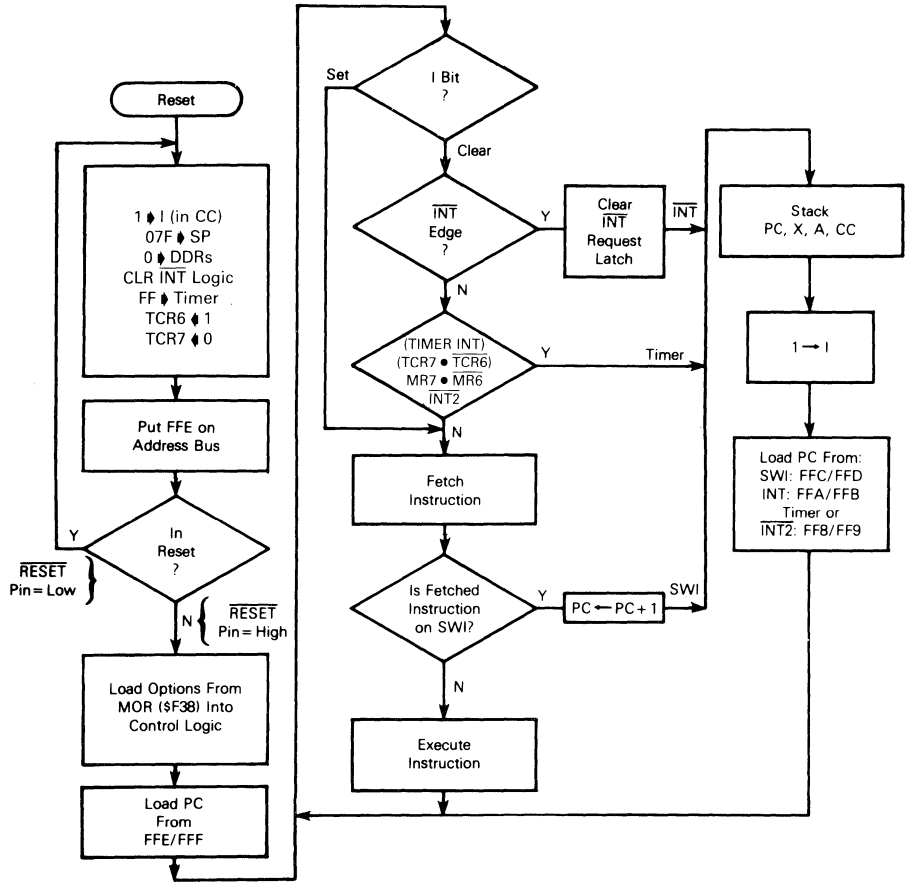


Figure 7. Reset and Interrupt Processing Flowchart

Zero-Crossing Interrupt

A sinusoidal input signal (f_{INT} maximum) can be used to generate an external interrupt (see Figure 8a) for use as a zero-crossing detector (for negative transitions of the ac sinusoid). This type of circuit allows applications such as servicing time-of-day routines and engaging/disengaging ac power control devices. Off-chip, full-wave rectification provides an interrupt at every zero crossing of the ac signal and, thereby, provides a 2f clock.

Digital-Signal Interrupt

With this type of circuit (Figure 8b), the $\overline{INT}$ pin can be driven by a digital signal. The maximum frequency of a signal that can be recognized by the TIMER or $\overline{INT}$ pin logic is dependent on the parameter labeled t_{WL}, t_{WH}. Refer to **TIMER** for additional information.

SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. The SWI execution is similar to the hardware interrupts.

MODES OF OPERATION

The MCU has two modes of operations: normal and bootstrap. The following paragraphs describe these modes.

NORMAL MODE

This mode is a single-chip mode and is entered if the following conditions are met: (1) the $\overline{RESET}$ line is low, (2) the PC0 pin is within its normal operational range, and (3) the V_{PP} pin is connected to V_{SS}. The next rising edge of the $\overline{RESET}$ pin then causes the part to enter the normal mode.

BOOTSTRAP MODE

The bootstrap mode is entered if the TIMER pin = + 12 V. Refer to application note, *MC6805P3/R3/U3 8-Bit EPROM Microcomputer Programming Module* (AN–857 Rev.2).

TIMER

The MCU consists of an 8-bit software programmable counter driven by a 7-bit software programmable prescaler. The various timer sources are made via the timer control register (TCR) and/or the mask option register (MOR). The 8-bit counter may be loaded under program control and is decremented toward zero. When the timer reaches zero, the timer interrupt request bit (bit 7) in the timer control register (TCR) is set. Refer to Figure 9 for timer block diagram.

The timer interrupt can be masked (disabled) by setting the timer interrupt mask bit (bit 6) in the TCR. When the I bit in the condition code register is cleared and TCR bit 6 is cleared, the processor receives the interrupt. The MCU responds to this interrupt by (1) saving the present CPU state on the stack, (2) fetching the timer interrupt vector, and (3) executing the interrupt routine. The timer interrupt request bit must be cleared by software. Refers to **RESETS** and **INTERRUPTS** for additional information.

The prescaler is a 7-bit divider which is used to extend the maximum length of the timer. To avoid truncation errors, the prescaler is cleared when TCR bit 3 is set to a logic one; however, the TCR bit 3 always reads as a logic zero to ensure proper operation with read-modify-write instructions.

The timer continues to count past zero, falling from $00 through $FF, and continues the countdown. The counter can be read at any time by reading the timer data register (TDR). This allows a program to determine the length of time since a timer interrupt has occurred without disturbing the counting process. The TDR is unaffected by reset.

SOFTWARE CONTROLLED MODE

This mode is selected when TOPT (bit 6) in the MOR is programmed to zero. The timer prescaler input can be configured for three different operating modes plus a disable mode, depending on the value written to TCR control bits 4 and 5 (TIE and TIN). The following paragraphs describe the different modes.

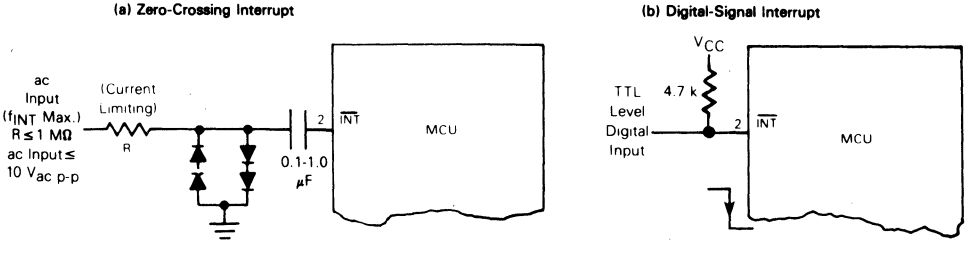


Figure 8. Typical Interrupt Circuits

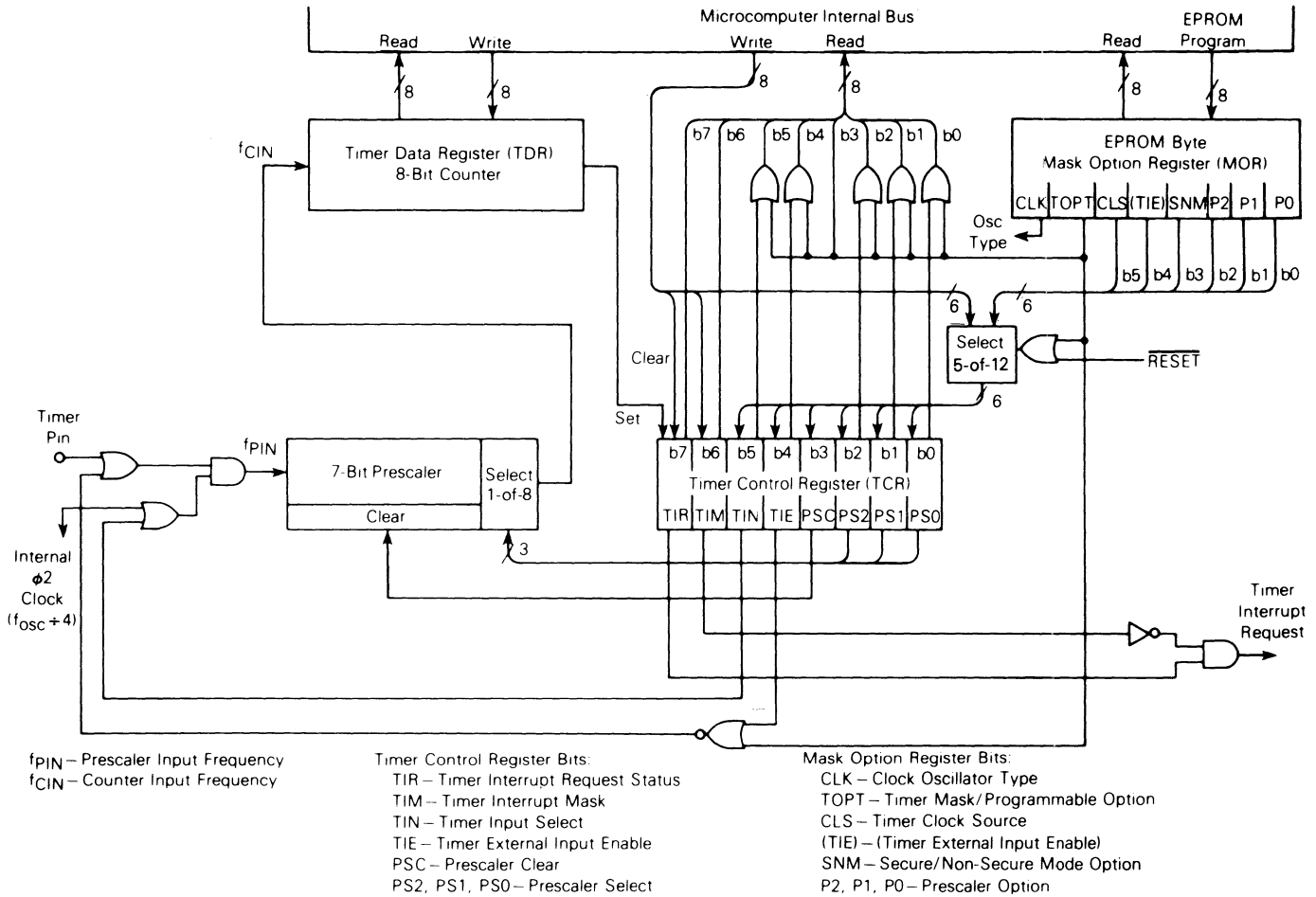


Figure 9. Timer Block Diagram

Timer Input Mode 1

When TIE and TIN are both programmed to zero, the timer input is from the internal clock (phase two) and the timer input pin is disabled. The internal clock mode can be used for periodic interrupt generation as well as a reference for frequency and event measurement.

Timer Input Mode 2

When TIE = 1 and TIN = 0, the internal clock and the timer input signals are ANDed to form the timer input. This mode can be used to measure external pulse widths. The active high, external pulse gates in the internal clock for the duration of the external pulse. The accuracy of the count is ±1.

Timer Input Mode 3

When TIE = 0 and TIN = 1, no prescaler input frequency is applied to the prescaler and the timer is disabled.

Timer Input Mode 4

When TIE and TIN are both one, the timer input is from the external clock. The external clock can be used to count external events as well as to provide an external frequency for generating periodic interrupts. Frequency of external input must be $\leq f_{OSC}/8$.

MOR CONTROLLED MODE

This mode is selected when TOPT (bit 6) in the MOR is programmed to logic one. The timer circuits are the same as described in **SOFTWARE CONTROLLED MODE**. The logic levels of TCR bits 0, 1, 2, and 5 are determined during EPROM programming by the same bits in the MOR. Therefore bits 0, 1, 2, and 5 in the MOR control the prescaler division and the timer clock selection. TIE (bit 4) and PSC (bit 3) in the TCR are set to a logic one when in the MOR controlled mode. TIM (bit 6) and TIR (bit 7) are controlled by the counter and software.

TIMER CONTROL REGISTER (TCR) $009

This is an 8-bit register that controls various functions such as configuring operation mode, setting ratio of the prescaler, and generating timer interrupt request signal. All bits are read/write except bit 3. The configuration of the TCR is determined by the TOPT (bit 6) in the MOR. When TOPT = 1, the TCR emulates the MC6805U2; when TOPT = 0, the TCR is controlled by software.

TCR with MOR TOPT = 1

7	6	5	4	3	2	1	0
TIR	TIM	*	1	PSC	*	*	*

TCR with MOR TOPT = 0

7	6	5	4	3	2	1	0
TIR	TIM	TIN	TIE	PSC	PS2	PS1	PS0
RESET:							
0	1	U	U	U	U	U	U

*The value of corresponding bits in MOR is written during $\overline{\text{RESET}}$ rising edge. These bits always read 'one'.

TIR — Timer Interrupt Request
Used to indicate the timer interrupt when it is logic one
1 = Set when the timer data register changes to all zeros
0 = Cleared by external reset, power-on reset, or under program control

TIM — Timer Interrupt Mask
Used to inhibit the timer interrupt
1 = Interrupt inhibited
0 = Interrupt enabled

TIN — External or Internal
Selects input clock source
1 = External clock selected
0 = Internal clock selected ($f_{OSC}/4$)

TIE — TIMER External Enable
Used to enable external TIMER pin. When TOPT = 1, TIE is always a logical "one".
1 = Enables external timer pin
0 = Disables external timer pin

PSC — Prescaler Clear
Write only bit. Writing a 1 to this bit resets the prescaler to zero. A read of this location always indicates a zero when TOPT = 0. When TOPT = 1, this bit will read a logical "one" and has no effect on the prescaler.

PS2, PS1, PS0 — Prescaler Clear
Decoded to select one of eight outputs of the prescaler

Prescaler

PS2	PS1	PS0	Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

NOTES

When changing the PS bits in software, the PSC bit should be written to a "one" in the same write cycle to clear the prescaler. Changing the PS bits without clearing the prescaler may cause prescaler truncation.

MASK OPTION REGISTER (MOR) $F38

The MOR is implemented in EPROM. This register contains all zeros prior to programming and is not affected by reset. The MOR bits are described in the following paragraphs.

7	6	5	4	3	2	1	0
CLK	TOPT	CLS		SNM	P2	P1	P0

CLK — Clock (oscillator type)
1 = Resistor Capacitor (RC)
0 = Crystal

TOPT — Timer Option
1 = MC6805U2 type timer/prescaler. All bits except 6 and 7, of the TCR are invisible to the user. Bits 5, 2, 1, and 0 of the MOR determine the equivalent MC6805U2 mask options.
0 = All TCR bits are implemented as a software programmable timer. The state of MOR bits 5, 4, 2, 1, and 0 sets the initial values of their respective TCR bits.

CLS — Timer/Prescaler Clock Source
1 = External TIMER pin
0 = Internal clock

Bit 4
Not used if TOPT = 1. Sets the initial value of TIE in the TCR if TOPT = 0.
1 = Not used
0 = Sets initial value of TIE in the TCR

SNM — Secure. Mode.
1 = EPROM contents cannot be access externally
0 = EPROM not programmed

P2, P1, P0
The logical levels of these bits, when decoded, select one of eight outputs on the timer prescaler.

Prescaler

P2	P1	P0	Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

PROGRAMMING CONTROL REGISTER (PCR) $00B

The PCR is an 8-bit register which provides the necessary control bits to program the EPROM. The bootstrap program manipulates the PCR when programming so the user need not be concerned with PCR in most applications.

TCR with MOR TOPT = 1

7	6	5	4	3	2	1	0
1	1	1	1	1	$\overline{VPON}$	$\overline{PGE}$	$\overline{PLE}$

RESET:

U	U	U	U	U	U	1	1

$\overline{PLE}$ — Programming Latch Enabe
Controls address and data being latched into the EPROM. Set during reset, but may be cleared anytime.
1 = Read EPROM
0 = Latch address and data on EPROM

$\overline{PGE}$ — Program Enable
Enables programming of EPROM. Must be set when changing the address and data. Set during reset.
1 = Inhibit EPROM programming
0 = Enable EPROM programming (if $\overline{PLE}$ is low)

$\overline{VPON}$ — V_{PP} On
A read-only bit that indicates high voltage at the V_{PP} pin. When set to "one", disconnects $\overline{PGE}$ and $\overline{PLE}$ from the chip.
1 = No high voltage on V_{PP} pin
0 = High voltage on V_{PP} pin

NOTE

VPON being "zero" does not indicate that the V_{PP} level is correct for programming. It is used as a safety interlock for the user in the normal operating mode.

$\overline{VPON}$	$\overline{PGE}$	$\overline{PLE}$	Programming Conditions
0	0	0	Programming mode (program EPROM byte)
1	0	0	$\overline{PGE}$ and $\overline{PLE}$ disabled from system
0	1	0	Programming disabled (latch address and data in EPROM)
1	1	0	$\overline{PGE}$ and $\overline{PLE}$ disabled from system
0	0	1	Invalid state; $\overline{PGE} = 0$ if $\overline{PLE} = 0$
1	0	1	Invalid state; $\overline{PGE} = 0$ if $\overline{PLE} = 0$
0	1	1	"High voltage" on V_{PP}
1	1	1	$\overline{PGE}$ and $\overline{PLE}$ disabled from system (operating mode)

EPROM PROGRAMMING

ERASING THE EPROM

The EPROM can be erased by exposure to high-intensity ultraviolet (UV) light with a wavelength of 2537 angstroms. The recommended integrated dose (UV intensity × exposure time) is 25Ws/cm^2. The lamps should be used without software filters and the MCU should be positioned about one inch from the UV tubes. Ultraviolet erasure clears all bits of the MCU EPROM to the "zero" state. Data then can be entered by programming "ones" into the desired bit locations.

PROGRAMMING

The MCU bootstrap program can be used to program the MCU EPROM. The alternate vectoring used to implement the self check is used to start execution of the bootstrap program.

A MCM2532 UV EPROM (other industry standard EPROMs may be used) must first be programmed with the same information that is to be transferred to the MCU EPROM. The MC68705U5 is programmed the same as the MC68705U3. Refer to application note, *MC68705P3/R3/U3 8-bit EPROM Microcomputer Programming Module* (AN-857 Rev.2) for schematic diagrams and instructions on programming the MCU EPROM.

EMULATION

The MC68705U5 emulates the MC6805U2 and MC6805U3 "exactly". The MC6805U2 and MC6805U3

mask features are implemented in the mask option register EPROM byte. The following identify the few minor exceptions to the exactness of the emulation.

1. The MC6805U2 "future ROM" areas are implemented in the MC68705U5 and these 1728 bytes must be left unprogrammed to accurately simulate the MC6805U2.
2. The reserved ROM areas have different data stored in them. In the MC6805U2 this area is used for self check, and in the MC68705U5 this area is used for the bootstrap program.
3. The MC6805U2 reads all ones in the 48 byte "future RAM" area. This area is not implemented on the MC6805U2/U3 mask ROM version but is implemented on the MC68705U5.
4. The MC68705U5 V_{PP} (pin 7) line is tied to V_{CC} during normal operations. On MC6805U2, this pin is grounded during normal operations; on the MC6805U3, this pin is not connected.

3

INSTRUCTION SET

The MCU has a set of 59 basic instructions which can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following instruction list.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to the following listing of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (2's Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two-byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Higher	BHI
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
(Branch if Higher or Same)	(BHS)
Branch if Carry Set	BCS
(Branch if Lower)	(BLO)
Branch if Not Equal	BNE
Branch if Equal	BEQ
Branch if Half Carry Clear	BHCC
Branch if Half Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Bit is Clear	BMC
Branch if Interrupt Mask Bit is Set	BMS
Branch if Interrupt Line is Low	BIL
Branch if Interrupt Line is High	BIH
Branch to Subroutine	BSR

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the first 256 bytes of the memory space where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear and bit test, and branch functions are all implemented with a single instruction. For test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch if Bit n is Set	BRSET n (n=0 . . . 7)
Branch if Bit n is Clear	BRCLR n (n=0 . . . 7)
Set Bit n	BSET n (n=0 . . . 7)
Clear Bit n	BCLR n (n=0 . . . 7)

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No-Operation	NOP

OPCODE MAP SUMMARY

Table 2 is an opcode map for the instructions used on the MCU.

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single-byte instructions, while the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. Two-byte direct-addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction.

RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −126 to +129 from the opcode address.

INDEXED, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. Thus, this addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this two-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

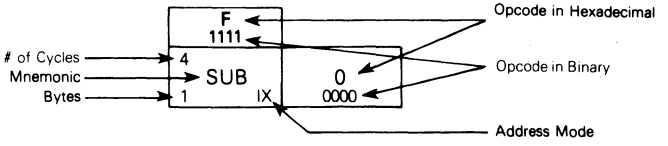
Table 2. Opcode Map

	Bit Manipulation		Branch	Read-Modify-Write					Control		Register/Memory						
	BTB 0 0000	BSC 1 0001	REL 2 0010	DIR 3 0011	INH 4 0100	INH 5 0101	IX1 6 0110	IX 7 0111	INH 8 1000	INH 9 1001	IMM A 1010	DIR B 1011	EXT C 1100	IX2 D 1101	IX1 E 1110	IX F 1111	
0 0000	10 BRSET0 3 BTB	7 BSET0 2 BSC	4 BRA 2 REL	6 NEG 2 DIR	4 NEG 1 INH	4 NEG 1 INH	7 NEG 2 IX1	6 NEG 1 IX	9 RTI 1 INH		2 SUB 2 IMM	4 SUB 2 DIR	5 SUB 3 EXT	6 SUB 3 IX2	5 SUB 2 IX1	4 SUB 1 IX	0 0000
1 0001	10 BRCLR0 3 BTB	7 BCLR0 2 BSC	4 BRN 2 REL						6 RTS 1 INH		2 CMP 2 IMM	4 CMP 2 DIR	5 CMP 3 EXT	6 CMP 3 IX2	5 CMP 2 IX1	4 CMP 1 IX	1 0001
2 0010	10 BRSET1 3 BTB	7 BSET1 2 BSC	4 BHI 2 REL								2 SBC 2 IMM	4 SBC 2 DIR	5 SBC 3 EXT	6 SBC 3 IX2	5 SBC 2 IX1	4 SBC 1 IX	2 0010
3 0011	10 BRCLR1 3 BTB	7 BCLR1 2 BSC	4 BLS 2 REL	6 COM 2 DIR	4 COMA 1 INH	4 COMX 1 INH	7 COM 2 IX1	6 COM 1 IX	11 SWI 1 INH		2 CPX 2 IMM	4 CPX 2 DIR	5 CPX 3 EXT	6 CPX 3 IX2	5 CPX 2 IX1	4 CPX 1 IX	3 0011
4 0100	10 BRSET2 3 BTB	7 BSET2 2 BSC	4 BCC 2 REL	6 LSR 2 DTR	4 LSRA 1 INH	4 LSRX 1 INH	7 LSR 2 IX1	6 LSR 1 IX			2 AND 2 IMM	4 AND 2 DIR	5 AND 3 EXT	6 AND 3 IX2	5 AND 2 IX1	4 AND 1 IX	4 0100
5 0101	10 BRCLR2 3 BTB	7 BCLR2 2 BSC	4 BCS 2 REL								2 BIT 2 IMM	4 BIT 2 DIR	5 BIT 3 EXT	6 BIT 3 IX2	5 BIT 2 IX1	4 BIT 1 IX	5 0101
6 0110	10 BRSET3 3 BTB	7 BSET3 2 BSC	4 BNE 2 REL	6 ROR 2 DIR	4 RORA 1 INH	4 RORX 1 INH	7 ROR 2 IX1	6 ROR 1 IX			2 LDA 2 IMM	4 LDA 2 DIR	5 LDA 3 EXT	6 LDA 3 IX2	5 LDA 2 IX1	4 LDA 1 IX	6 0110
7 0111	10 BRCLR3 3 BTB	7 BCLR3 2 BSC	4 BEQ 2 REL	6 ASR 2 DIR	4 ASRA 1 INH	4 ASRX 1 INH	7 ASR 2 IX1	6 ASR 1 IX		2 TAX 1 INH		5 STA 2 DIR	6 STA 3 EXT	7 STA 3 IX2	6 STA 2 IX1	5 STA 1 IX	7 0111
8 1000	10 BRSET4 3 BTB	7 BSET4 2 BSC	4 BHCC 2 REL	6 LSL 2 DIR	4 LSLA 1 INH	4 LSLX 1 INH	7 LSL 2 IX1	6 LSL 1 IX		2 CLC 1 INH	2 EOR 2 IMM	4 EOR 2 DIR	5 EOR 3 EXT	6 EOR 3 IX2	5 EOR 2 IX1	4 EOR 1 IX	8 1000
9 1001	10 BRCLR4 3 BTB	7 BCLR4 2 BSC	4 BHCS 2 REL	6 ROL 2 DIR	4 ROLA 1 INH	4 ROLX 1 INH	7 ROL 2 IX1	6 ROL 1 IX		2 SEC 1 INH	2 ADC 2 IMM	4 ADC 2 DIR	5 ADC 3 EXT	6 ADC 3 IX2	5 ADC 2 IX1	4 ADC 1 IX	9 1001
A 1010	10 BRSET5 3 BTB	7 BSET5 2 BSC	4 BPL 2 REL	6 DEC 2 DIR	4 DECA 1 INH	4 DECX 1 INH	7 DEC 2 IX1	6 DEC 1 IX		2 CLI 1 INH	2 ORA 2 IMM	4 ORA 2 DIR	5 ORA 3 EXT	6 ORA 3 IX2	5 ORA 2 IX1	4 ORA 1 IX	A 1010
B 1011	10 BRCLR5 3 BTB	7 BCLR5 2 BSC	4 BMI 2 REL							2 SEI 1 INH	2 ADD 2 IMM	4 ADD 2 DIR	5 ADD 3 EXT	6 ADD 3 IX2	5 ADD 2 IX1	4 ADD 1 IX	B 1011
C 1100	10 BRSET6 3 BTB	7 BSET6 2 BSC	4 BMC 2 REL	6 INC 2 DIR	4 INCA 1 INH	4 INCX 1 INH	7 INC 2 IX1	6 INC 1 IX		2 RSP 1 INH		3 JMP 2 DIR	4 JMP 3 EXT	5 JMP 3 IX2	4 JMP 2 IX1	3 JMP 1 IX	C 1100
D 1101	10 BRCLR6 3 BTB	7 BCLR6 2 BSC	4 BMS 2 REL	6 TST 2 DIR	4 TSTA 1 INH	4 TSTX 1 INH	7 TST 2 IX1	6 TST 1 IX		2 NOP 1 INH	8 BSR 2 REL	7 JSR 2 DIR	8 JSR 3 EXT	9 JSR 3 IX2	8 JSR 2 IX1	7 JSR 1 IX	D 1101
E 1110	10 BRSET7 3 BTB	7 BSET7 2 BSC	4 BIL 2 REL								2 LDX 2 IMM	4 LDX 2 DIR	5 LDX 3 EXT	6 LDX 3 IX2	5 LDX 2 IX1	4 LDX 1 IX	E 1110
F 1111	10 BRCLR7 3 BTB	7 BCLR7 2 BSC	4 BIH 2 REL	6 CLR 2 DIR	4 CLRA 1 INH	4 CLRX 1 INH	7 CLR 2 IX1	6 CLR 1 IX		2 TXA 1 INH		5 STX 2 DIR	6 STX 3 EXT	7 STX 3 IX2	6 STX 2 IX1	5 STX 1 IX	F 1111
Hi / Low																	Hi / Low

Abbreviations for Address Modes

INH	Inherent
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This addressing mode can be used in a manner similar to indexed, 8-bit offset except that this three-byte instruction allows tables to be anywhere in memory.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte to which the specified bit is to be set or cleared. Thus, any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single two-byte instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit to be tested, and its condition (set or clear), is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

3

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS (Voltages Referenced to V_{SS})

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage EPROM Programming Voltage (V_{PP} Pin) TIMER Pin — Normal Mode TIMER Pin — Bootstrap Programming Mode All Others	 V_{PP} V_{in} V_{in} V_{in}	 −0.3 to +22.0 −0.3 to +7.0 −0.3 to +15.0 −0.3 to +7.0	V
Operating Temperature Range MC68705U5 MC68705U5C	T_A	T_L to T_H 0 to +70 −40 to +85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature Cerdip	T_J	 175	°C/W

These devices contain circuity to protect the inputs against damage due to high static voltages or electrical fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high-impedance circuit. For proper operation, it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ and $V_{out}) \leq V_{CC}$. Reliability of operation is enhanced if unused inputs except EXTAL are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Cerdip	θ_{JA}	 60	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A

PROGRAMMING OPERATION ELECTRICAL CHARACTERISTICS

(V_{CC} = 5.25 Vdc ±0.5%, V_{SS} = 0 Vdc, T_A = 20°C to 30°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Programming Voltage	V_{PP}	20.0	21.0	22.0	V
V_{PP} Supply Current V_{PP} = 5.25 V V_{PP} = 21.0 V	I_{PP}	 — —	 — —	 8 30	mA
Programming Oscillator Frequency	f_{oscp}	0.9	1.0	1.1	MHz
Bootstrap Programming Mode Voltage (TIMER Pin) @ I_{IHTP} = 100 μA Maximum	V_{IHTP}	9.0	12.0	15.0	V

ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage $\overline{RESET}$ (4.99 ≤ V_{CC} ≤ 5.51) (V_{CC} < 4.75) $\overline{INT}$ (4.99 ≤ V_{CC} ≤ 5.51) (V_{CC} < 4.75) All Other	V_{IH}	 4.0 V_{CC} − 0.5 4.0 V_{CC} − 0.5 2.0	 — — ** ** —	 V_{CC} V_{CC} V_{CC} V_{CC} V_{CC}	V
Input High Voltage (TIMER Pin) Timer Mode Bootstrap Programming Mode	V_{IH}	 2.0 9.0	 — 12.0	 V_{CC} + 1.0 15.0	V
Input Low Voltage $\overline{RESET}$ $\overline{INT}$ All Other	V_{IL}	 V_{SS} V_{SS} V_{SS}	 — ** —	 0.8 1.5 0.8	V
Internal Power Dissipation (No Port Loading, V_{CC} = 5.25 V T_A = 0°C for Steady-State Operation T_A = 48°C	P_{INT}	— —	520 580	740 800	mW
Input Capacitance XTAL All Other	C_{in}	 — —	 25 10	 — —	pF
$\overline{INT}$ Zero-Crossing Input Voltage — Through a Capacitor	V_{INT}	2.0	—	4.0	$V_{ac\ p\text{-}p}$
$\overline{RESET}$ Hysteresis Voltage Out of Reset Voltage Into Reset Voltage	 V_{IRES+} V_{IRES-}	 2.1 0.8	 — —	 4.0 2.0	V
Programming Voltage (V_{PP} Pin) Programming EPROM Operating Mode	V_{PP}*	 20.0 4.75	 21.0 V_{CC}	 22.0 5.75	V
Input Current $\overline{TIMER}$ (V_{in} = 0.4 V) $\overline{INT}$ (V_{in} = 0.4 V) EXTAL (V_{in} = 2.4 V to V_{CC} Crystal Option) (V_{in} = 0.4 V Crystal Option) $\overline{RESET}$ (V_{in} = 0.8 V) (External Capacitor Changing Current)	 I_{in} I_{RES}	 — — — — −4.0	 — 20 — — —	 20 50 10 −1600 −40	μA

*V_{PP} (pin 7) is connected to V_{CC} in the normal operating mode.
**Due to internal biasing, this input (when not used) floats to approximately 2.0 V.

SWITCHING CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Oscillator Frequency Normal	f_{osc}	0.4	—	4.2	MHz
Instruction Cycle Time ($4/f_{osc}$)	t_{cyc}	0.950	—	10	μs
$\overline{INT}$, $\overline{INT2}$, or Timer Pulse Width	t_{WL}, t_{WH}	$t_{cyc}+250$	—	—	ns
$\overline{RESET}$ Pulse Width	t_{RWL}	$t_{cyc}+250$	—	—	ns
$\overline{RESET}$ Delay Time (External Cap = 1.0 μF)	t_{RHL}	100	—	—	ms
$\overline{INT}$ Zero-Crossing Detection Input Frequency	f_{INT}	0.03	—	1.0	kHz
External Clock Duty Cycle (EXTAL)	—	40	50	60	%
Crystal Oscillator Start-Up Time	—	—	—	100	ms

PORT ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, T_A = 0° to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Port A					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Output High Voltage, I_{Load} = −10 μA	V_{OH}	$V_{CC}-1.0$	—	—	V
Input High Voltage, I_{Load} = −300 μA (Max)	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage, I_{Load} = −500 μA (Max)	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current (V_{in} = 2.0 V to V_{CC})	I_{IH}	—	—	−300	μA
Hi-Z State Input Current (V_{in} = 0.4 V)	I_{IL}	—	—	−500	μA
Port B					
Output Low Voltage, I_{Load} = 3.2 mA	V_{OL}	—	—	0.4	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.0	V
Output High Voltage, I_{Load} = −200 μA	V_{OH}	2.4	—	—	V
Darlington Current Drive (Source), V_O = 1.5 V	I_{OH}	−1.0	—	−10	mA
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port C					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	—	—	0.4	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.4	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	<2	10	μA
Port D (Input Only)					
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	—	0.8	V
Input Current	I_{in}	—	<1	5	μA

3

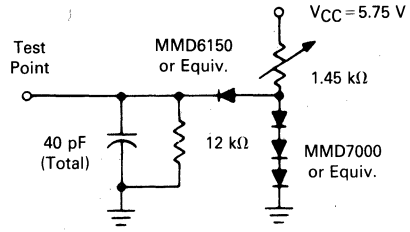


Figure 10. TTL Equivalent Test Load (Port B)

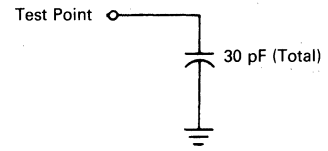


Figure 11. CMOS Equivalent Test Load (Port A)

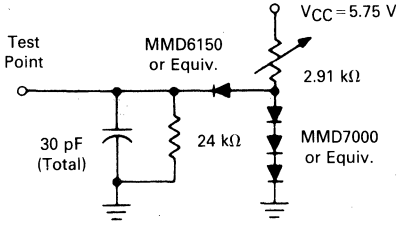


Figure 12. TTL Equivalent Test Load (Ports A and C)

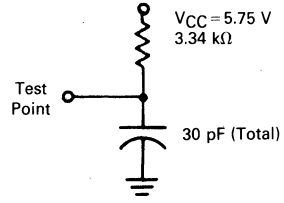


Figure 13. Open-Drain Equivalent Test Load (Port C)

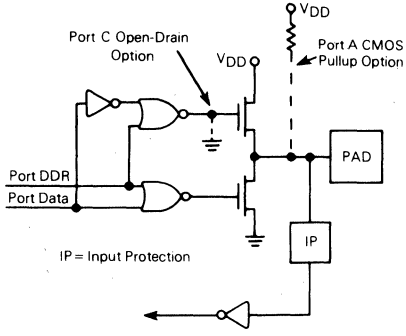


Figure 14. Ports A and C Logic Diagram

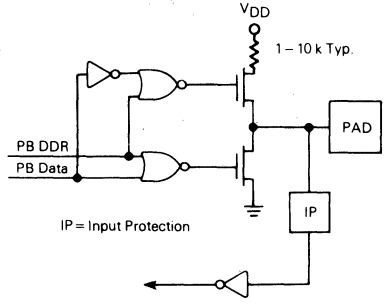


Figure 15. Port B Logic Diagram

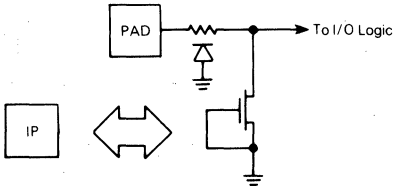


Figure 16. Typical Input Protection

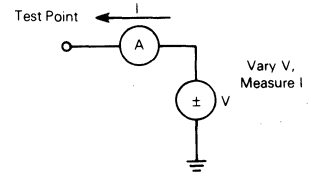
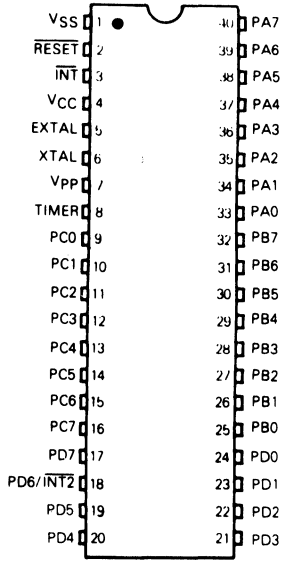


Figure 17. I/O Characteristic Measurement Circuit

MECHANICAL DATA

This section contains the pin assignments and package dimensions for the MC68705U5.

PIN ASSIGNMENTS



ORDERING INFORMATION

The following table provides generic information pertaining to the package type, temperature, and MC order numbers for the MC68705U5.

Table 3. Generic Information

Package Type	Temperature	Order Number
Cerdip	0°C to 70°C	MC68705U5S
S Suffix	−40°C to +85°C	MC68705U5CS

MC68HC05A6

Product Preview

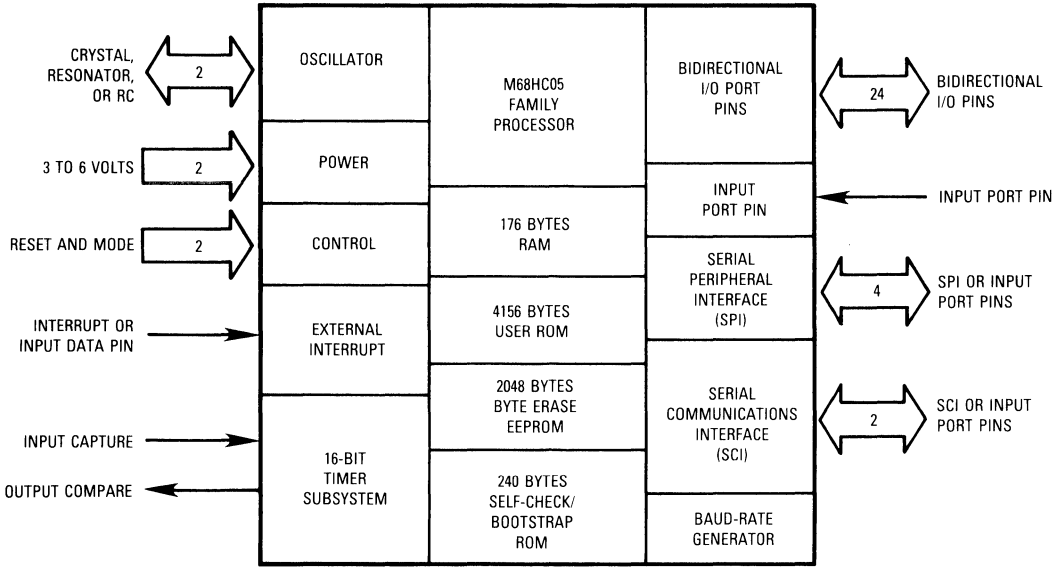
8-Bit Microcontroller Unit

The MC68HC05A6 is an advanced 8-bit microcontroller unit (MCU) with highly sophisticated on-chip peripheral capabilities. This device is similar to the MC68HC05C4 with some differences including 2048 bytes of EEPROM and 4156 bytes of user ROM.

The following are some of the hardware and software features of the MC68HC05A6.

- HCMOS Technology
- Fully Static Operation
- 4156 Bytes of User ROM
- 176 Bytes of RAM
- 2048 Bytes of EEPROM
- 240 Bytes of Self-Check Bootstrap Loader ROM
- 24 Bidirectional I/O Lines
- 16-Bit Timer Subsystem
- Serial Communications Interface (SCI)
- Serial Peripheral Interface (SPI)
- Interrupts: External, Timer, SCI, and SPI
- Master Reset and Power-On Reset
- Single 3- to 6-Volt Supply
- On-Chip Oscillator with RC or Crystal/Ceramic Resonator Mask Option
- 2.1 MHz Internal Operation Frequency at 5 Volts
- True Bit Manipulation
- Memory Mapped I/O
- Two Power Saving Standby Modes
- Multiply Instruction
- 40-Pin Dip, 44-Pin PLCC Package
- EEPROM Programming Bootstrap and Charge Pump On-Chip

BLOCK DIAGRAM



3

MC68HC05B4

Technical Summary
8-Bit Microcontroller Unit

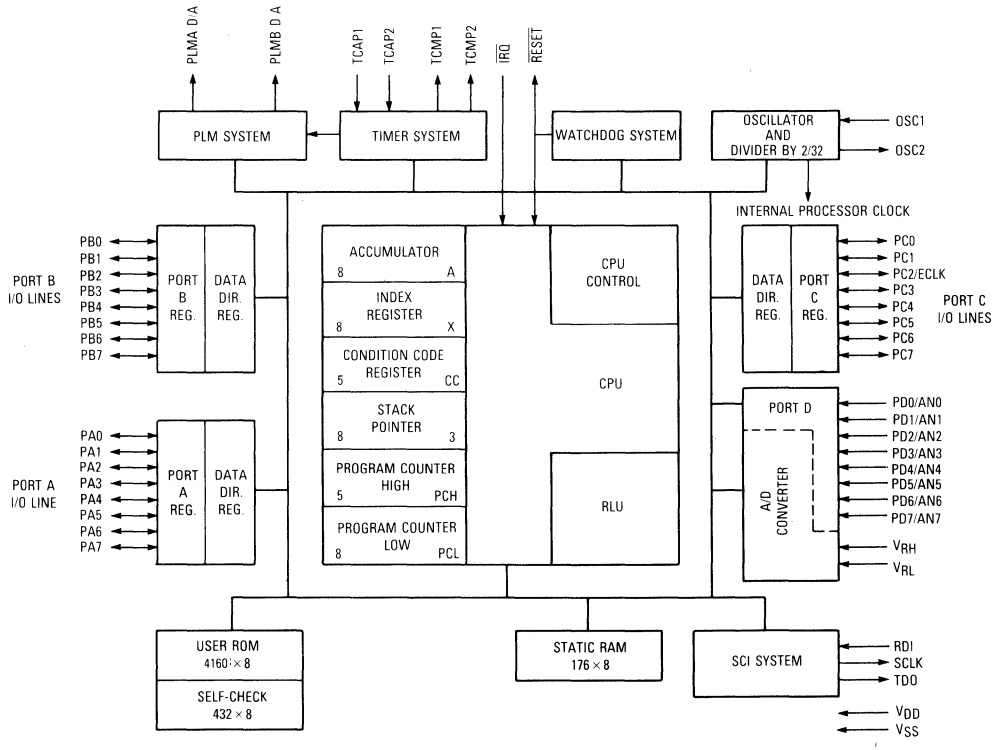
The MC68HC05B4 (HCMOS) microcontroller unit (MCU) is a member of the M68HC05 Family of microcontrollers. This high-performance, low-power MCU has parallel I/O capability with pins programmable as input or output. This publication contains condensed information on the MCU; for more detailed information, contact your local Motorola sales office.

The following block diagram depicts the hardware features; additional features available on the MCU are shown below and at the top of page 2.

- On-Chip Oscillator with Crystal/Ceramic Resonator
- Memory-Mapped I/O
- 176 Bytes of On-Chip RAM
- 4160 Bytes of User ROM
- 24 Bidirectional I/O Lines and 8 Input-Only Lines

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

Features — continued

- Serial Communications Interface (SCI) System
- 8-Channel A/D Converter
- Watchdog System
- Self-Check Mode
- Power-Saving STOP and WAIT Modes
- Single 3.0- to 6.0-Volt Supply
- Fully Static Operation
- Two Pulse-Length Modulation Systems (D/A)
- 2-Channel Pulse Length Modulator
- Slow Mode Option Divides the Basic Clock Frequency by 16
- 16-Bit Timer with Two Input Input Capture and Two Output Compare Functions

SIGNAL DESCRIPTION

The signal descriptions of the MCU are discussed in the following paragraphs.

V_{DD} AND V_{SS}

Power is supplied to the microcontroller using these two pins. V_{DD} is the positive supply, and V_{SS} is ground.

$\overline{IRQ}$

This pin is a programmable option that provides four different choices of interrupt triggering sensitivity. Refer to **INTERRUPTS** for more detail. Note that the voltage level on this pin affects the mode of operation.

OSC1, OSC2

These pins provide control input for an on-chip clock oscillator circuit. A crystal, a ceramic resonator, or an external signal connects to these pins providing a system clock. The oscillator frequency is two times the internal bus rate (or 32 times as a software option).

Crystal

The circuit shown in Figure 1(b) is recommended when using a crystal. Using an external CMOS oscillator is recommended when crystals outside the specified ranges are to be used. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and start-up stabilization time. Refer to **ELECTRICAL SPECIFICATIONS** for V_{DD} specifications.

Ceramic Resonator

A ceramic resonator may be used in place of the crystal in cost-sensitive applications. The circuit in Figure 1(b) is recommended when using a ceramic resonator. Figure 1(a) lists the recommended capacitance and resistance values. The manufacturer of the resonator considered should be consulted for specific information on resonator operation.

External Clock

An external clock should be applied to the OSC1 input with the OSC2 input not connected, as shown in Figure 1(d).

INPUT CAPTURE (TCAP1)

This pin controls the input capture 1 feature for the on-chip programmable timer. Note that the voltage level on this pin affects the mode of operation.

INPUT CAPTURE (TCAP2)

This pin controls the input capture 2 feature for the on-chip programmable timer.

OUTPUT COMPARE (TCMP1)

This pin provides an output for the output compare 1 feature of the on-chip timer.

OUTPUT COMPARE (TCMP2)

This pin provides an output for the output compare 2 feature on the on-chip timer.

$\overline{RESET}$

This pin is used to reset the MCU and provide an orderly start-up procedure by pulling $\overline{RESET}$ low. The voltage level on this pin affects the mode of operation (see Table 2, **Mode of Operation Selection**).

INPUT/OUTPUT PORTS (PA7–PA0, PB7–PB0, PC7–PC0)

These 24 lines are arranged into three 8-bit ports (A, B, and C). These ports are programmable as either inputs or outputs under software control of the data direction registers. Refer to **PROGRAMMING** for additional information.

FIXED INPUT PORT (PD0/AN0–PD7/AN7)

These eight lines comprise port D, a fixed input port. If the A/D function is enabled, it affects this port. Port D accepts the eight analog inputs when the A/D is enabled. Port D can be used for digital input during a conversion sequence, but this may inject noise on the analog signals, reducing the conversion accuracy. Also, a digital read of port D with levels other than V_{DD} or V_{SS} on the pins results in greater power dissipation during the read cycle. Refer to **PROGRAMMING** for additional information.

NOTE

In the 48-pin dual-in-line package, the fixed input port (D) of the MC68HC05B4 is reduced to six pins

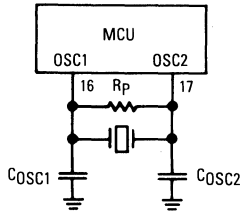
Crystal

	2 MHz	4 MHz	Units
R_{SMAX}	400	75	Ω
C_0	5	7	pF
C_1	0.008	0.012	μF
C_{OSC1}	15-40	15-30	pF
C_{OSC2}	15-30	15-25	pF
R_P	10	10	MΩ
Q	30	40	K

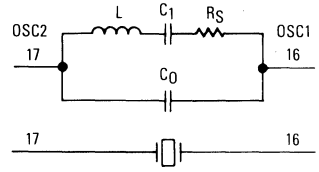
Ceramic Resonator

	2-4 MHz	Units
R_S (typical)	10	Ω
C_0	40	pF
C_1	4, 3	μF
C_{OSC1}	30	pF
C_{OSC2}	30	pF
R_P	1-10	MΩ
Q	1250	—

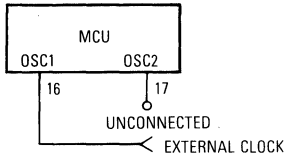
(a) Crystal/Ceramic Resonator Parameters



(b) Crystal/Ceramic Resonator Oscillator Connections



(c) Equivalent Crystal Circuit



(d) External Clock Source Connections

Figure 1. Oscillator Connections

(PD5–PD0, AN5–AN0). This change has no effect on either programming or operation of port D or the A/D converter.

PLMA

This pin is the output of the pulse-length modulation converter A. See **PULSE-LENGTH D/A CONVERTERS** for further information.

PLMB

This pin is the output of the pulse-length modulation converter B. See **PULSE-LENGTH D/A CONVERTERS** for further information.

RDI (Receive Data In)

This pin is the input of the SCI receiver. See **Serial Communications Interface** for more information.

TDO (Transmit Data Out)

This pin is the output of the SCI transmitter. See **Serial Communications Interface** for more information.

SCLK

This pin is the clock output pin of the SCI transmitter. See **Serial Communications Interface** for more information.

V_{RH}

This pin is the positive reference voltage for the A/D converter.

V_{RL}

This pin is the negative reference voltage for the A/D converter.

INPUT/OUTPUT PROGRAMMING

Input/output port programming, fixed input port programming, and serial port programming are discussed in the following paragraphs.

INPUT/OUTPUT PORT PROGRAMMING

Any port pin is programmable as either an input or an output under software control of the corresponding data direction register (DDR). Each port bit can be selected as output or input by writing the corresponding bit in the port DDR to a logic one for output and logic zero for input. On reset, all DDRs are initialized to logic zero to put the ports in the input mode. The port output registers are not initialized on reset but may be written to before setting the DDR bits to avoid undefined levels.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This port write may be used to initialize the data registers and avoid undefined outputs. Refer to Figure 2 for typical port circuitry and to Table 1 for a list of the I/O pin functions.

Table 1. I/O Pin Functions

R/$\overline{W}$*	DDR	I/O Pin Functions
0	0	The I/O pin is in input mode. Data is written into the output data latch.
0	1	Data is written into the output data latch and output to the I/O pin.
1	0	The state of the I/O pin is read.
1	1	The I/O pin is in an output mode. The output data latch is read.

*R $\overline{W}$ is an internal signal.

Under software control, the PC2 pin can become the CPU clock output. If this option is selected, the corresponding DDR bit is automatically set, and bit 2 of port C always reads the output data latch. The other port C pins are not affected by this feature.

E Clock Control Register (CTL/ECLK) $07

7	6	5	4	3	2	1	0
0	0	0	0	ECLK	0	0	0
RESET: 0	0	0	0	0	0	0	0

ECLK — ECLK Control
1 – I/O port function of PC2 is forced to output mode, and PC2 outputs the ECLK CPU clock.
0 – PC2 functions as a regular I/O pin.

FIXED INPUT PORT PROGRAMMING

Port D is a fixed input port that monitors the external pins whenever the A/D is disabled. After reset, all the bits become digital inputs because all special function drivers are disabled. Port D is always a digital input, whether the A/D is on or off.

NOTE

Any unused inputs and I/O ports should be tied to an appropriate logic level (e.g., either V_{DD} or V_{SS}).

SERIAL PORT (SCI) PROGRAMMING

The SCI uses two or three pins for its functions: RDI for its receive data input, TDO for its transmit data output, and SCLK to output the transmitter clock, if needed.

MEMORY

The MCU is capable of addressing 8192 bytes of memory and I/O registers, as shown in Figure 3. The locations consist of user ROM, user RAM, self-check ROM, control registers, and I/O. The user-defined reset and interrupt vectors are located from $1FF0 to $1FFF.

The shared stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer decrements during pushes and increments

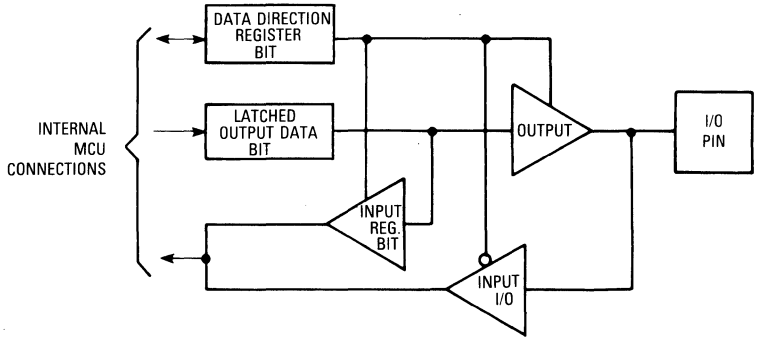


Figure 2. Typical Port I/O Circuit

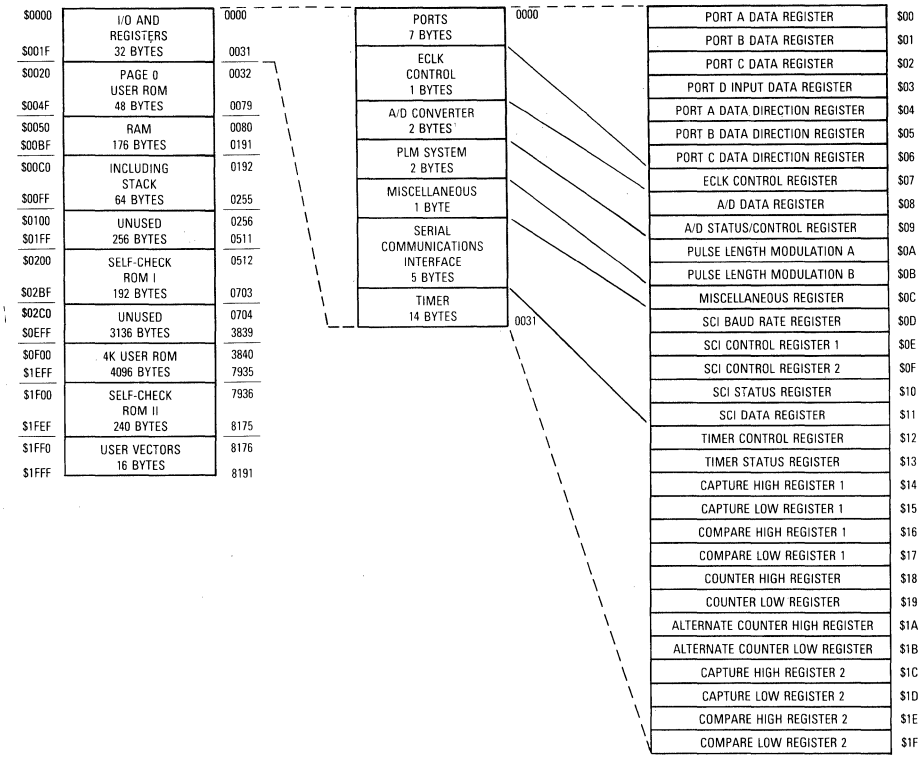


Figure 3. Memory Map

during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

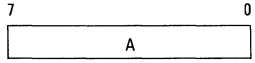
Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

REGISTERS

The MCU contains the registers described in the following paragraphs.

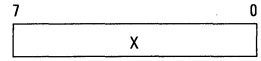
ACCUMULATOR (A)

The accumulator is a general-purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



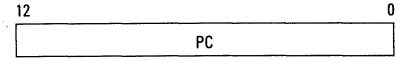
INDEX REGISTER (X)

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16-bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



PROGRAM COUNTER (PC)

The program counter is a 13-bit register that contains the address of the next instruction to be executed.

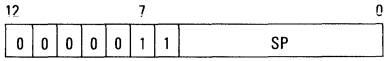


STACK POINTER (SP)

The stack pointer is a 13-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set to location $00FF. The stack pointer

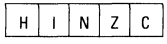
is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

When accessing memory, the seven most significant bits are permanently set to 0000011. These seven bits are appended to the six least significant register bits to produce an address within the range of $00FF to $00C0. Subroutines and interrupts may use up to 64 (decimal) locations. If 64 locations are exceeded, the stack pointer wraps around and loses the previously stored information. A subroutine call occupies two locations on the stack; an interrupt uses five locations.



CONDITION CODE REGISTER (CCR)

The CCR is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program, and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.



Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, the timer and external interrupt is masked (disabled). If an interrupt occurs while this bit is set, the interrupt is latched and processed as soon as the interrupt bit is cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic one).

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/Borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions and during shifts and rotates.

SELF-CHECK

The self-check capability provides the ability to determine if the device is functional. Table 2 shows how self-check mode is entered. Self-check is performed using the circuit shown in Figure 4. Port C pins PC3–PC0 are monitored for the self-check results. After reset, the following tests are performed automatically:

I/O — Exercise of ports A, B, C, and D
RAM — Counter test for each RAM byte
ROM — Exclusive OR with odd ones parity result
Timer — Tracks counter register and checks ICF1, ICF2, OCF1, OCF2, and TOV flag
Interrupts — Tests external, timer, and SCI interrupts
SCI — Transmission test; checks RDRF, TDRE, TC, and FE flags
A/D — Checks A/D on internal channels: V_{RL}, V_{RH}, and $(V_{RL} + V_{RH})/2$
PLM — Checks basic PLM function
Watchdog System — Checks watchdog function

Self-check results (using the LEDs as monitors) are shown in Table 3. The following subroutines are available to the user and do not require any external hardware.

Table 2. Mode of Operation Selection

$\overline{\text{RESET}}$ Pin	$\overline{\text{IRQ}}$ Pin	TCAP1 Pin	Mode
_/‾	V_{SS} to V_{DD}	V_{SS} to V_{DD}	Normal
_/‾	+9 Volts	V_{DD}	Self-Check
V_{SS}	V_{SS} to V_{DD}	V_{SS} to V_{DD}	Reset Condition

Table 3. Self-Check Results

PC3	PC2	PC1	PC0	Remarks
1	0	0	1	Bad Port
0	1	1	0	Bad Port
1	0	1	0	Bad RAM
1	0	1	1	Bad ROM
1	1	0	0	Bad Timer
1	1	0	1	Bad SCI
1	1	1	0	Bad A/D
0	0	0	0	Not Used
0	0	0	1	Bad PLM
0	0	1	0	Bad Interrupts
0	0	1	1	Bad Watchdog
Flashing				Good Device
All Others				Bad Device, Bad Port, etc.

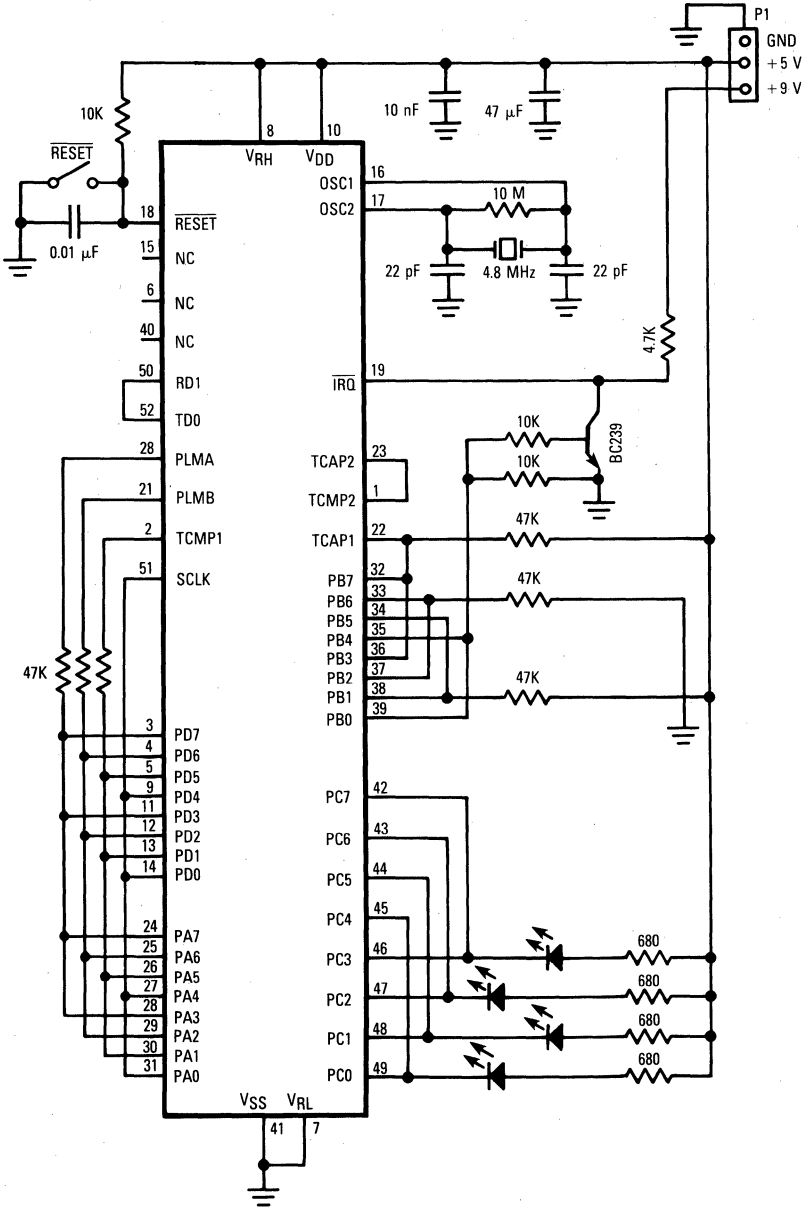
0 indicates LED is on; 1 indicates LED is off.

RAM CHECK SUBROUTINE

This subroutine returns with the Z bit cleared if any error is detected; otherwise, the Z bit is set. The stack pointer must be set to $FF. The RAM check subroutine is called at location $021E. A counter test is done on each location from address $50 to $FD. Each location is made to count from $00 to $00 again. Locations $FE and $FF are assumed to contain the return address. Upon return to the user's program, if the test passed, X = $00, A = $00, and RAM locations $0050 and $00FD contain $01.

NOTE

The watchdog system is turned on when calling this subroutine.



NOTE: Pin numbers are valid for 52-pin PLCC package only.

Figure 4. Self-Check Circuit Schematic Diagram

A/D CONVERTER CHECK SUBROUTINE

This subroutine returns with the Z bit cleared if any error is detected; otherwise, the Z bit is set. The subroutine is called at location $1FAA with X = $00 and A/D STAT/CTRL (address $09) = $20 (ADON = 1 for more than 100 μs and channel PD0 selected). Conversion is done on three of the internal channels: V_{RH}, V_{RL}, and $(V_{RL} + V_{RH})/2$. The result of these conversions is verified at ±1 LSB. Upon return to the user's program, if the test passed, X = $09, A = $00 or $01.

ROM CHECKSUM SUBROUTINE

This subroutine returns with the Z bit cleared if any error is detected; otherwise, the Z bit is set. The ROM checksum subroutine is called at location $0232 with RAM location $0053 equal to $01 and A = 0. A short routine is set up and executed in RAM to compute a checksum of the entire ROM pattern. RAM locations $0050 through $0053 are overwritten. Upon return to the user's program, if the test passed, X = 0, A = 0.

NOTE

The A/D and the watchdog system are turned on when calling this subroutine.

RESETS

The MCU can be reset two ways: by initial power-up (POR) and by the external reset input ($\overline{\text{RESET}}$). The $\overline{\text{RESET}}$ input consists mainly of a Schmitt trigger that senses the $\overline{\text{RESET}}$ line logic level.

POWER-ON RESET (POR)

An internal reset is generated on power-up to allow the internal clock generator to stabilize. The power-on reset is strictly for power turn-on conditions and should not be used to detect a drop in the power supply voltage. There is a delay (t_{PORL}) after the oscillator becomes active. If the $\overline{\text{RESET}}$ pin is low at the end of t_{PORL} the MCU will remain in the reset condition until $\overline{\text{RESET}}$ goes high. A mask option allows t_{PORL} to be either 16 or 4064 internal processor clock cycles (t_{CYC}).

EXTERNAL RESET INPUT

The MCU is reset when a logic zero is applied to the $\overline{\text{RESET}}$ input for a period of one and one-half machine cycles (t_{CYC}).

Miscellaneous Register (0C)

7	6	5	4	3	2	1	0
POR	INTP	INTN	INTE	SFA	SFB	SM	WDOG
RESET: U	0	0	1	0	0	0	0

POR — Power-On Reset
- 1 = The reset occurring is a power-on, not external, reset
- 0 = Power-on reset not in progress

INTP — External Interrupt Positive
Allows a choice of $\overline{\text{IRQ}}$ sensitivity, with INTN. See Table 4.

INTN — External Interrupt Negative
Allows a choice of $\overline{\text{IRQ}}$ sensitivity, with INTP. See Table 4.

INTE — External Interrupt Enable
Allows the user to enable or disable the external interrupt function

SFA — Slow/Fast Selection for PLMA
- 1 = Slow speed used for PLMA (4096 times the timer clock period)
- 0 = Fast speed used for PLMA (256 times the timer clock period). See **PULSE-LENGTH D/A CONVERTERS**

SFB — Slow/Fast Selection for PLMB
- 1 = Slow speed used for PLMB (4096 times the timer clock period)
- 0 = Fast speed used for PLMB (256 times the timer clock period). See **PULSE-LENGTH D/A CONVERTERS**

SM — Slow Mode
- 1 = System runs at 1/16th the normal clock rate ($f_{OSC}/32$)
- 0 = System runs at normal clock rate ($f_{OSC}/2$)

WDOG — Watchdog Counter System
- 1 = Watchdog counter system enabled
- 0 = Watchdog counter system disabled

NOTE

The reset generated by the watchdog timer is a system reset; thus, the watchdog is disabled after a watchdog reset.

Table 4. External Interrupt Options

INTP	INTN	External Interrupt Options
0	0	Negative Edge and Low-Level Sensitive
0	1	Negative Edge Only
1	0	Positive Edge Only
1	1	Positive and Negative Edge Sensitive

Slow Mode

The slow mode function is controlled by the SM bit in the miscellaneous register (0C). In slow mode (SM = 1), an extra divide-by-sixteen circuit is added between the oscillator and the internal clock driver. This slows all functions by a factor of 16 (including SCI, A/D, and timer), which is particularly useful in WAIT mode. SM is cleared by external or power-on reset and by STOP mode.

NOTE

If slow mode is enabled while using the A/D, the internal A/D RC oscillator should be turned on.

Watchdog System

The watchdog counter is driven by the 1024 prescaler in the timer and, unless the counter is reset, generates a system reset when it reaches its maximum count (1024 × 8).

A mask option is available that provides two methods of enabling the watchdog timer. In the first option, the watchdog system is controlled by the WDOG bit in the miscellaneous register (0C). Writing a one to the bit starts the watchdog or, if it is already started, resets the counter to zero. Writing a zero has no effect; the WDOG bit can only be cleared by external or power-on reset. In the second option, the watchdog timer is always enabled following reset.

A second mask option determines the watchdog timer function during WAIT. The watchdog timer can remain active during WAIT, and can cause a reset if the device remains in WAIT longer than the watchdog timeout period. Alternatively, the watchdog timer suspends operation during WAIT and resets its count, resuming normal operation following reset.

INTERRUPTS

The MCU can be interrupted four different ways: the three maskable hardware interrupts ($\overline{IRQ}$, SCI, and timer) and the nonmaskable software interrupt instruction (SWI).

Interrupts cause the processor to save register contents on the stack and to set the interrupt mask (I bit) to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack and normal processing to resume. The stacking order is shown in Figure 5.

Unlike $\overline{RESET}$, hardware interrupts do not cause the current instruction execution to be halted but are considered pending until the current instruction is complete.

NOTE

The current instruction is the one already fetched and being operated on.

When the current instruction is complete, the processor checks all pending hardware interrupts. If unmasked (I bit clear) and if the corresponding interrupt enable bit is set, the processor proceeds with interrupt processing; otherwise, the next instruction is fetched and executed.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction, regardless of the I-bit state.

Refer to Figure 6 for the reset and interrupt instruction processing sequence.

TIMER INTERRUPT

There are three different timer interrupt flags that cause a timer interrupt whenever they are set and enabled. The interrupt flags are in the timer status register (TSR), and the enable bits are in the timer control register (TCR). Refer to **TIMER** for more information.

EXTERNAL INTERRUPT

If the interrupt mask bit (I bit) of the CCR is set, all interrupts are disabled. Clearing the I bit enables the external interrupt. The external interrupt is internally synchronized and then latched on the falling edge of $\overline{IRQ}$. The action of the external interrupt is identical to the timer interrupt with the exception that the interrupt request input at $\overline{IRQ}$ is latched internally and the service routine address is specified by the contents of $1FFA and $1FFB.

Four options are available for interrupt triggering sensitivity:

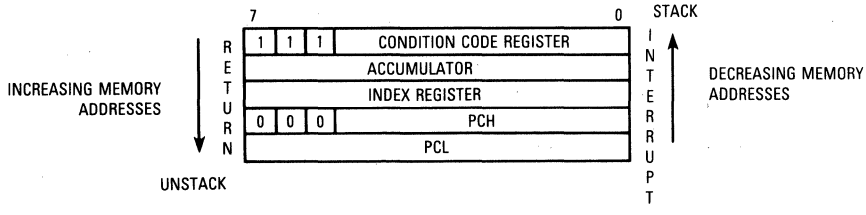
- Negative edge and low level
- Negative edge only
- Positive edge only
- Positive and negative edge

See **Miscellaneous Register (0C)** for further information.

Figure 7 shows a mode timing diagram for the interrupt line. The timing diagram shows two treatments of the interrupt line to the processor. The first method shows a single pulse on the interrupt line spaced far enough apart to be serviced. The minimum time between pulses is a function of the length of the interrupt service. Once a pulse occurs, the next pulse should not occur until an RTI occurs. This time (t_{ILIL}) is obtained by adding 21 instruction cycles to the total number of cycles it takes to complete the service routine (not including the RTI instruction). The second method shows many interrupt lines "wire-ORed" to form the interrupts at the processor. If the interrupt line remains low after servicing an interrupt, then the next interrupt is recognized.

NOTE

The internal interrupt latch is cleared in the first part of the interrupt service routine; therefore, one external interrupt pulse could be latched and serviced as soon as the I bit is cleared.



NOTE: Since the Stack Pointer decrements during pushes, the PCL is stacked first, followed by PCH, etc. Pulling from the stack is in the reverse order.

Figure 5. Interrupt Stacking Order

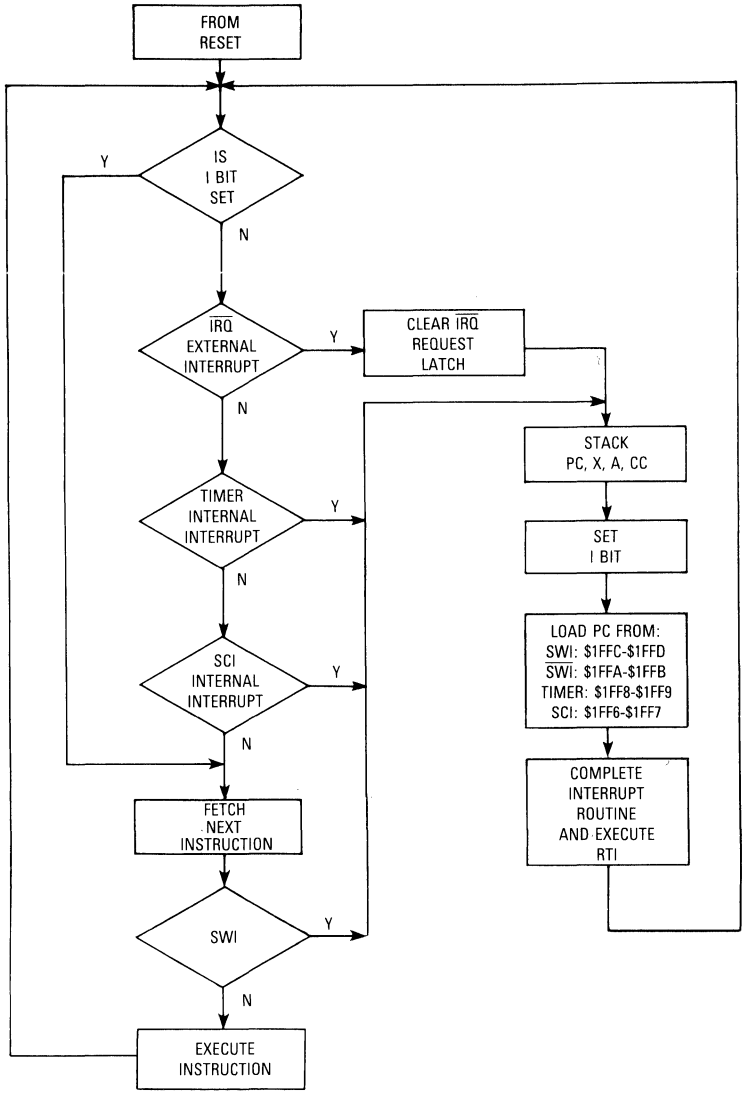


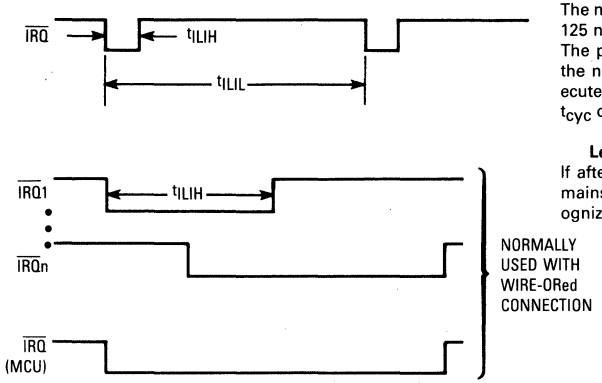
Figure 6. Reset and Interrupt Processing Flowchart

SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. If the I bit is zero, SWI executes after the other interrupts. The SWI operation is similar to the hardware interrupts. The interrupt service routine address is specified by the contents of memory locations $1FFC and $1FFD.

SCI INTERRUPTS

An interrupt in the SCI occurs when one of the interrupt flag bits in the serial communications status register is set, provided the I bit in the CCR is clear and the enable bit in the serial communications control register 2 is set. Software in the serial interrupt service routine must determine the cause and priority of the SCI interrupt by



Edge-Sensitive Trigger Condition
The minimum pulse width (t_{ILIH}) is either 125 ns (V_{DD} = 5 V) or 250 ns (V_{DD} = 3 V). The period t_{ILIL} should not be less than the number of t_{cyc} cycles it takes to execute the interrupt service routine plus 21 t_{cyc} cycles.

Level-Sensitive Trigger Condition
If after servicing an interrupt the $\overline{IRQ}$ remains low, then the next interrupt is recognized.

Figure 7. External Interrupt Mode Diagram

3

examining the interrupt flags and status bits in the SCI status register.

LOW-POWER MODES

STOP

The STOP instruction places the MCU in its lowest power consumption mode. In the STOP mode, the internal oscillator is turned off, halting all internal processing including timer, SCI, and A/D operation (refer to Figure 8).

During the STOP mode, the TCR bits are altered to remove any pending timer interrupt request and to disable any further timer interrupts. The timer prescaler is cleared. The I bit in the CCR is cleared to enable external interrupts. All other registers and memory remain unaltered. All input/output lines remain unchanged. The processor can be brought out of the STOP mode only by an external interrupt or reset.

SCI during STOP Mode

When the MCU enters the STOP mode, the baud rate generator stops, halting all SCI activity. If the STOP instruction is executed during a transmitter transfer, that transfer is halted. If a low input to the $\overline{IRQ}$ pin is used to exit STOP mode, the transfer resumes. If the SCI receiver is receiving data and the STOP mode is entered, received data sampling stops because the baud rate generator stops, and all subsequent data is lost. For these reasons, all SCI transfers should be in the idle state when the STOP instruction is executed.

Watchdog during STOP Mode

The STOP instruction is inhibited when the watchdog system is enabled. If a STOP instruction is executed while the watchdog is enabled, a reset occurs that resets the entire MCU.

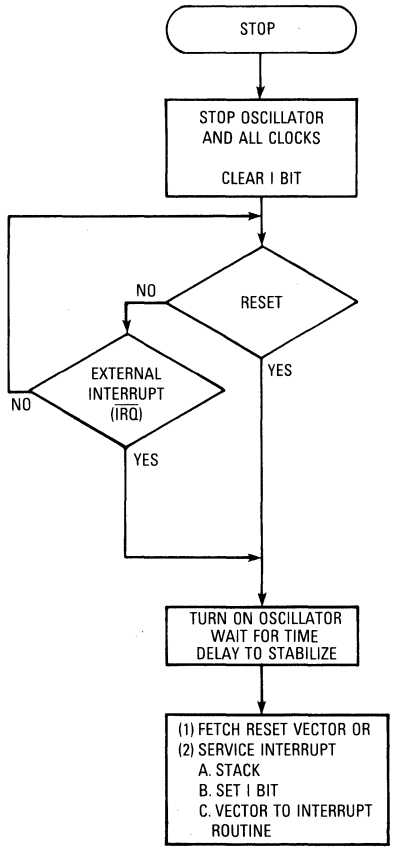


Figure 8. STOP Function Flowchart

PLM during STOP Mode

When the MCU enters stop mode, the PLM outputs remain at their particular level. If power-on or external reset causes the exit from stop mode, the register values are forced to $00.

A/D Converter during STOP Mode

When stop mode is entered with the A/D converter turned on, the A/D clocks are stopped and the A/D converter is disabled for the duration of stop mode, including the t_{PORL} startup time. If the A/D RC oscillator is used, it will also be disabled.

When leaving STOP mode, after the t_{PORL} startup time, the A/D converter and A/D RC oscillator resume regular operation. However, a time t_{ADON} is required for the current sources to stabilize. During t_{ADON}, A/D conversion results may be inaccurate.

WAIT

The WAIT instruction places the MCU in a low-power consumption mode, but the WAIT mode consumes more power than the STOP mode. All CPU action and the watchdog system are suspended, but the timer, SCI, PLM, and A/D remain active (refer to Figure 9). An interrupt from the timer, SCI, or an IRQ can cause the MCU to exit the WAIT mode.

During the WAIT mode, the I bit in the CCR is cleared to enable interrupts. All other registers, memory, and input/output lines remain in their previous state. The timer may be enabled to allow a periodic exit from the WAIT mode.

To achieve proper operation and reduce power consumption, the following points should be set as desired before entering wait mode:

- Timer interrupt enable bits
- A/D control bits
- SCI enable bits and interrupt enable bits

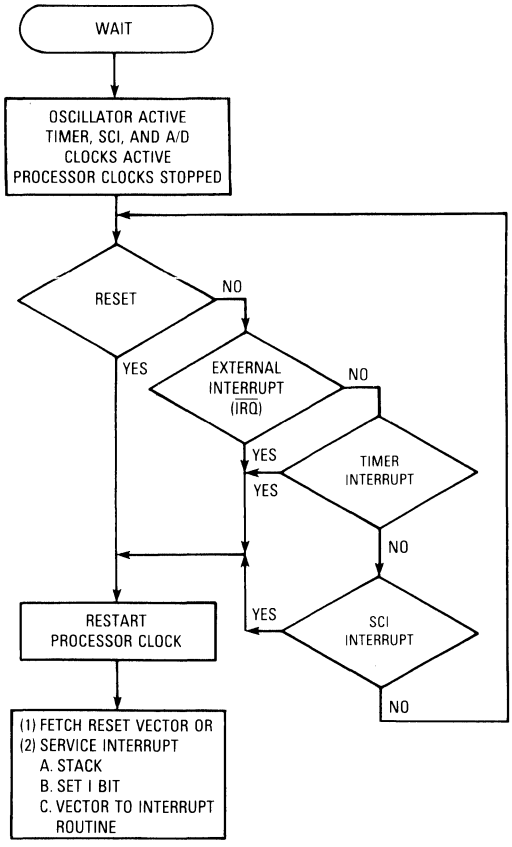


Figure 9. WAIT Function Flowchart

TIMER

The timer consists of a 16-bit, software-programmable counter driven by a fixed divide-by-four prescaler. This timer can be used for many purposes, including input waveform measurements of two input signals while simultaneously generating two output waveforms. Pulse widths can vary from several microseconds to many seconds. The programmable timer works in conjunction with the PLM system to execute two 8-bit D/A PLM conversions, with a choice of two repetition rates. Refer to Figure 10 for a timer block diagram.

Because the timer has a 16-bit architecture, each specific functional segment (capability) is represented by two registers. These registers contain the high and low byte of that functional segment. Generally, accessing the low byte of a specific timer function allows full control of that function; however, an access of the high byte inhibits that specific timer function until the low byte is also accessed.

NOTE

The I bit in the CCR should be set while manipulating both the high and low byte register of a specific timer function to ensure that an interrupt does not occur.

COUNTER

The key element in the programmable timer is a 16-bit, free-running counter or counter register, preceded by a prescaler that divides the internal processor clock by four. The prescaler gives the timer a resolution of 2.0 microseconds if the internal bus clock is 2.0 MHz. The counter is incremented during the low portion of the internal bus clock. Software can read the counter at any time without affecting its value.

The double-byte, free-running counter can be read from either of two locations, $18–$19 (counter register) or $1A–$1B (counter alternate register). A read from only the least significant byte (LSB) of the free-running counter ($19, $1B) receives the count value at the time of the read.

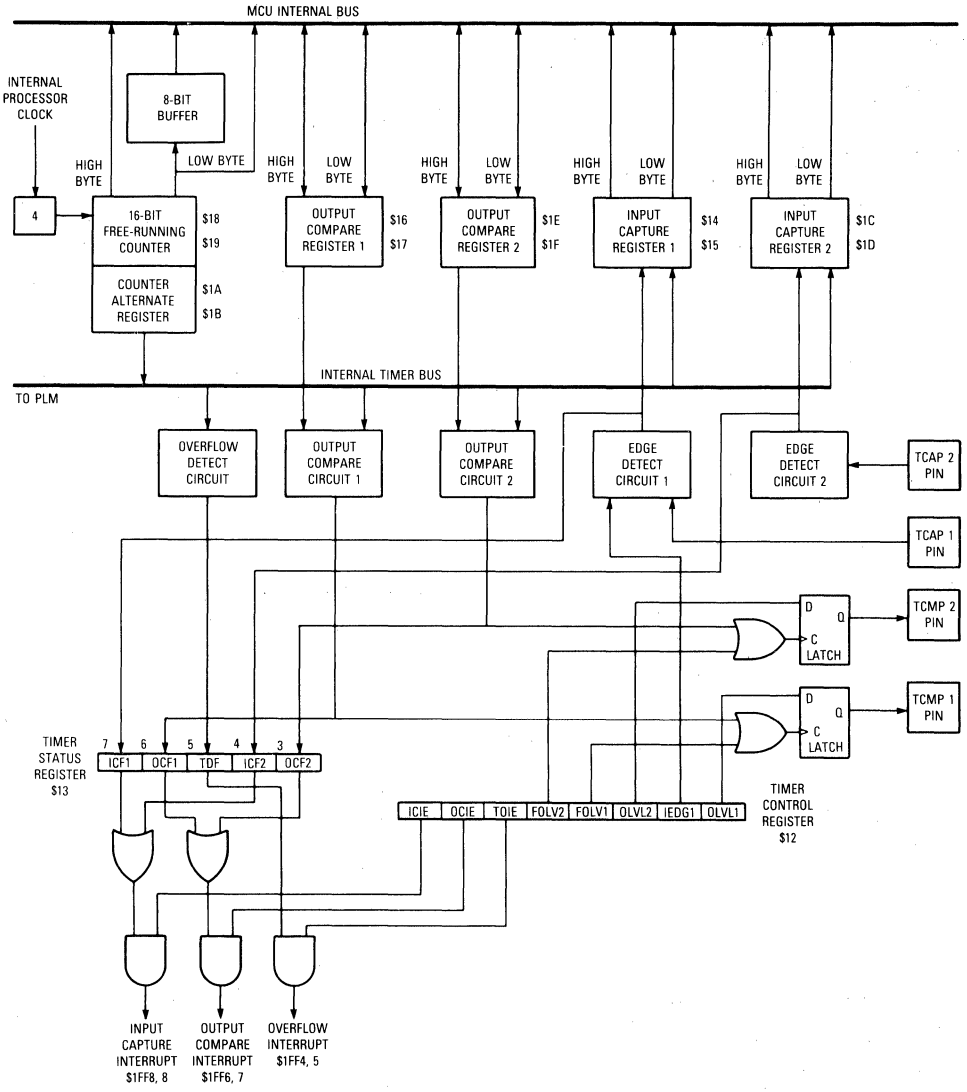


Figure 10. Timer Block Diagram

If a read of the free-running counter or counter alternate register first addresses the most significant byte (MSB) ($18, $1A), the LSB ($19, $1B) is transferred to a buffer. This buffer value remains fixed after the first MSB read, even if the user reads the MSB several times. This buffer is accessed when reading the free-running counter or counter alternate register LSB ($19 or $1B) and, thus, completes a read sequence of the total counter value. In reading either the free-running counter or counter alternate register, if the MSB is read, the LSB must also be read to complete the sequence.

The counter alternate register differs from the counter register in one respect: a read of the counter register LSB can clear the timer overflow flag (TOF). Therefore, the counter alternate register can be read at any time without the possibility of missing timer overflow interrupts due to clearing of the TOF.

The free-running counter is configured to $FFFC during reset and is always a read-only register. During a power-on reset, the counter is also preset to $FFFC and begins running after the oscillator start-up delay. Because the free-running counter is 16 bits preceded by a fixed divide-

by-four prescaler, the value in the free-running counter repeats every 262,144 internal bus clock cycles. When the counter rolls over from $FFFF to $0000, the TOF bit is set. An interrupt can also be enabled when counter rollover occurs by setting its interrupt enable bit (TOIE).

NOTE

Since the PLM system uses the timer counter, PLM results will be affected when resetting this counter.

OUTPUT COMPARE REGISTERS

There are two output compare registers: output compare register 1 (OCR1) and output compare register 2 (OCR2). The output compare registers can be used for several purposes, such as controlling an output waveform or indicating when a period of time has elapsed. All bits are readable and writable and are not altered by the timer hardware or reset. If the compare function is not needed, the four bytes of the output compare registers can be used as storage locations.

NOTE

The same output compare interrupt enable bit is used for the two output compares.

Output Compare Register 1

The output compare register 1 (OCR1) is a 16-bit register, which is made up of two 8-bit registers at locations $16 (most significant byte) and $17 (least significant byte).

The output compare register contents are continually compared with the contents of the free-running counter and, if a match is found, the corresponding output compare flag (OCF1, bit 6 of timer status register $13) is set, and the corresponding output level (OLVL1) bit is clocked to pin TCMP1. The output compare register values and the output level bit should be changed after each successful comparison to control an output waveform or establish a new elapsed timeout. An interrupt can also accompany a successful output compare, provided the corresponding interrupt enable bit, OCIE, is set.

After a processor write cycle to the output compare register 1 containing the most significant byte ($16), the output compare 1 function is inhibited until the least significant byte ($17) is also written. The user must write both bytes (locations) if the most significant byte is written first. A write made only to the least significant byte ($17) will not inhibit the compare function. The free-running counter is updated every four internal bus clock cycles. The minimum time required to update the output compare register 1 is a function of the program rather than the internal hardware.

The processor can write to either byte of the output compare register 1 without affecting the other byte. The output level (OLVL1) bit is clocked to the corresponding output level register and then to the TCMP1 pin, regardless of whether the output compare flag (OCF1) is set or clear.

Output Compare Register 2

The output compare register 2 (OCR2) is a 16-bit register, which is made up of two 8-bit registers at locations $1E (most significant byte) and $1F (least significant byte). The function of OCR2 is identical to OCR1, requiring only changes of the register locations and control bits in the timer status register ($13) to make the OCR1 description apply to OCR2.

SOFTWARE FORCE COMPARE

The MCU provides a force compare capability to facilitate fixed frequency generation as well as other applications. Bit 3 (FOLV1 for OCR1) and bit 4 (FOLV2 for OCR2) in the timer control register ($12) implement this force compare. Writing a one to these bits causes the OLVL1 or OLVL2 values to be copied to the respective output registers (TCMP1 or TCMP2 pins). Internal logic allows a single instruction to change OLVL1 and OLVL2 and cause a forced compare with the new values of OLVL1 and OLVL2.

NOTE

A software force compare, which affects the corresponding output pin TCMP1 or TCMP2, does not affect the compare flag; thus, it does not generate an interrupt.

INPUT CAPTURE REGISTERS

There are two input capture registers: input capture register 1 (ICR1) and input capture register 2 (ICR2).

NOTE

The same input capture interrupt enable bit (ICIE) is used for the two input capture registers.

Input Capture Register 1

Two 8-bit registers that make up the 16-bit input capture register 1 (ICR1) are read-only and are used to latch the value of the free-running counter after the corresponding input capture edge detector senses a defined transition. The level transition that triggers the counter transfer is defined by the corresponding input edge bit (IEDG1). Reset does not affect the contents of the input capture register.

The result obtained by an input capture will be one more than the value of the free-running counter on the rising edge of the internal bus clock preceding the external transition. This delay is required for internal synchronization. Resolution is one count of the free-running counter, which is four internal-bus clock cycles.

The free-running counter contents are transferred to the input capture register on each proper signal transition, regardless of whether the input capture flag (ICF1) is set or clear. The input capture register always contains the free-running counter value, which corresponds to the most recent input capture.

After a read of the input capture register 1 ($14) most significant byte, the counter transfer is inhibited until the least significant byte ($15) is also read. This characteristic causes the time used in the input capture software routine and its interaction with the main program to determine the minimum pulse period.

A read of the input capture register 1 least significant byte ($15) does not inhibit the free-running counter transfer since they occur on opposite edges of the internal bus clock.

Input Capture Register 2

The input capture register 2 (ICR2) is a 16-bit register that is composed of two 8-bit registers at locations $1C (most significant byte) and $1D (least significant byte). Input capture register 2 functions identically to input capture register 1, except that only negative edge sensitivity is available. By substituting the appropriate bits in the timer status register ($13) and substituting register locations, the ICR1 description applies to ICR2.

TIMER CONTROL REGISTER (TCR) $12

The TCR is an 8-bit read/write register, illustrated below with a definition of each bit.

7	6	5	4	3	2	1	0
ICIE	OCIE	TOIE	FOLV2	FOLV1	OLVL2	IEDG1	OLVL1
RESET: 0	0	0	0	0	0	U	0

ICIE — Input Capture Interrupt Enable
- 1 = Interrupt enabled
- 0 = Interrupt disabled

OCIE — Output Compare Interrupt Enable
- 1 = Interrupt enabled
- 0 = Interrupt disabled

TOIE — Timer Overflow Interrupt Enable
- 1 = Interrupt enabled
- 0 = Interrupt disabled

FOLV2 — Force Output Compare 2
- 1 = Forces the OLVL2 bit to the corresponding output latch
- 0 = No effect

FOLV1 — Force Output Compare 1
- 1 = Forces the OLVL1 bit to the corresponding output latch
- 0 = No effect

OLVL2 — Output Level 2
- 1 = The value of the output level 2 bit, which is copied to the output level latch by the next successful output compare 2, appears at TCMP2
- 0 = No effect

IEDG1 — Input Edge One

Value of input edge determines which level transition on TCAP1 pin will trigger free-running counter transfer to the input capture register.
- 1 = Positive edge
- 0 = Negative edge

OLVL1 — Output Level One

Value of output level, which is clocked into output level register by the next successful output compare 1, will appear on the TCMP pin.
- 1 = High output
- 0 = Low output

TIMER STATUS REGISTER (TSR) $13

The TSR is a read-only register containing three status flag bits. Bits 4–0 always read zero.

7	6	5	4	3	2	1	0
ICF1	OCF1	TOF	ICF2	OCF2	—	—	—
RESET: U	U	U	U	U	—	—	—

ICF1 — Input Capture Flag One
- 1 = Flag set when selected polarity edge is sensed by input capture edge detector
- 0 = Flag cleared when TSR and input capture 1 low register ($15) are accessed

OCF1 — Output Capture Flag One
- 1 = Flag set when output compare register contents match the free-running counter contents
- 0 = Flag cleared when TSR and output compare 1 low register ($17) are accessed

TOF — Timer Overflow Flag
- 1 = Flag set when free-running counter transition from $FFFF to $0000 occurs
- 0 = Flag cleared when TSR and counter low register ($19) are accessed

ICF2 — Input Capture Flag Two
- 1 = Flag set when selected polarity edge is sensed by input capture 2 edge detector
- 0 = Flag cleared when TSR and input capture 2 low register ($1D) are accessed

OCF2 — Output Capture Flag Two
- 1 = Flag set when output compare register contents match the free-running counter contents
- 0 = Flag cleared when TSR and output compare low register 2 ($1F) are accessed

Bits 0–2 — Not Used

Can read either zero or one.

TIMER DURING WAIT MODE

The CPU clock halts during the WAIT mode, but the timer remains active. An interrupt from the timer causes the processor to exit the WAIT mode.

TIMER DURING STOP MODE

In the STOP mode, the timer stops counting and holds the last count value if STOP is exited by an interrupt. If reset is used, the counter is forced to $FFFC. During STOP, if at least one valid input capture edge occurs at the TCAP pin, the input capture detect circuit is armed. This does not set any timer flags or wake up the MCU, but when the MCU does wake up, there is an active input capture flag and data from the first valid edge that occurred during the STOP mode. If reset is used to exit STOP mode, then no input capture flag or data remains, even if a valid input capture edge occurred.

Accessing the timer status register satisfies the first condition required to clear status bits. The remaining step is to access the register corresponding to the status bit. A problem can occur when using the timer overflow function and reading the free-running counter at random times

to measure an elapsed time. Without incorporating the proper precautions into software, the timer overflow flag could unintentionally be cleared if 1) the timer status register is read or written when TOF is set, and 2) the least significant byte of the free-running counter is read but not for the purpose of servicing the flag.

The counter alternate register at address $1A and $1B contains the same value as the free-running counter (at address $18 and $19); therefore, this alternate register can be read at any time without affecting the timer overflow flag in the timer status register.

SERIAL COMMUNICATIONS INTERFACE

A full-duplex asynchronous SCI is provided with a standard NRZ format and a variety of baud rates. The SCI transmitter and receiver are functionally independent but use the same data format and baud rate prescaler. The terms baud and bit rate are used synonymously in the following description.

SCI TWO-WIRE SYSTEM FEATURES

- Standard NRZ (mark/space) format
- Advanced error detection method includes noise detection for noise duration of up to one-sixteenth bit time
- Full-duplex operation (simultaneous transmit and receive)
- Software programmable for one of 32 different baud rates
- Different baud rates possible for transmit and receive
- Software-selectable word length (eight- or nine-bit words)
- Separate transmitter and receiver enable bits
- SCI may be interrupt driven
- Four separate interrupt conditions

SCI RECEIVER FEATURES

- Receiver wake-up function (idle or address bit)
- Idle line detect
- Framing error detect
- Noise detect
- Overrun detect
- Receiver data register full flag

SCI TRANSMITTER FEATURES

- Transmit data register empty flag
- Transmit complete flag
- Break send

Any SCI two-wire system requires receive data in (RDI) and transmit data out (TDO).

DATA FORMAT

Receive data in (RDI) or transmit data out (TDO) is the serial data presented between the internal data bus and the output pin (TDO) and between the input pin (RDI) and the internal data bus. Data format is as shown for the NRZ in Figure 11.

WAKE-UP FEATURE

In a typical multiprocessor configuration, the software protocol will usually identify the addressee(s) at the beginning of the message. To permit uninterested MPUs to ignore the remainder of the message, a wake-up feature is included, whereby all further SCI receiver flag (and interrupt) processing can be inhibited until its data line returns to the idle state. An SCI receiver is re-enabled by an idle string of at least ten (or eleven) consecutive ones. Software for the transmitter must provide for the required idle string between consecutive messages and prevent it from occurring within messages.

A second wake-up method is available in which sleeping SCI receivers can be awakened by a logic one in the high-order bit of a received character.

RECEIVE DATA IN

Receive data in (RDI) is the serial data which is presented from the input pin via the SCI to the receive data register (RDR). While waiting for a start bit, the receiver samples the input at a rate 16 times higher than the set baud rate. This increased rate is referred to as the RT rate. When the input (idle) line is detected low, it is tested for three more sample times. If at least two of these three samples detect a logic low, a valid start bit is assumed to be detected. If in two or more samples, a logic high is detected, the line is assumed to be idle. The receive clock generator is controlled by the baud rate register (see Figure 13); however, the SCI is synchronized by the start bit independent of the transmitter. Once a valid start bit is detected, the start bit, each data bit, and the stop bit are

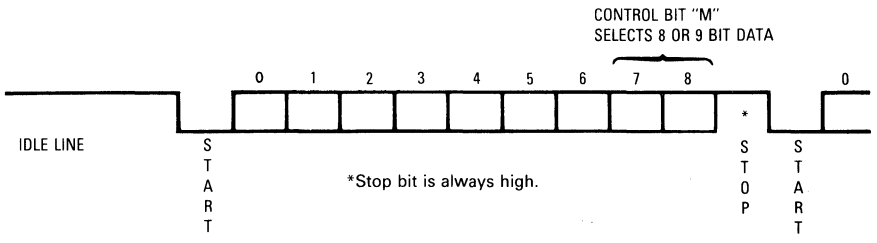


Figure 11. Data Format

each sampled three times. The value of the bit is determined by voting logic, which takes the value of a majority of samples. A noise flag is set when all three samples on a valid start bit, data bit, or stop bit do not agree. A noise flag is also set when the start verification samples do not agree.

START BIT DETECTION FOLLOWING A FRAMING ERROR

If there has been a framing error (FE) without detection of a break (10 zeros for 8-bit format or 11 zeros for a 9-bit format), the circuit continues to operate as if there actually were a stop bit, and the start edge will be placed artificially. The last bit received in the data shift register is inverted to a logic one, and the three logic-one start qualifiers are forced into the sample shift register during the interval when detection of a start bit is anticipated; therefore, the start bit will be accepted no sooner than it is anticipated.

If the receiver detects that a break (RDRF = 1, FE = 1, receiver data register = $00) produced the framing error, the start bit will not be artificially induced, and the receiver must actually receive a logic one before start.

3

SCI SYNCHRONOUS TRANSMISSION

The SCI transmitter allows a one-way synchronous transmission, with the SCLK pin as the clock output. No clock is sent to the SCLK pin during the stop and start bits. The LCL bit (SSCR1) controls whether clocks are active during the last valid data bit (address mark). The CPOL bit selects clock polarity, and the CPHA bit selects the phase of the external clock. During idle, preamble, and send break, the external SCLK clock is not active.

These options allow the SCI to control serial peripherals consisting of shift registers without losing any function of the SCI transmitter. These options do not affect the SCI receiver, which is independent of the transmitter.

The SCLK pin works in conjunction with the TDO pin. When the SCI transmitter is disabled, the SCLK and the TDO pins assume a high-impedance state.

NOTE

THE LBCL, CPOL and CPHA bits must be selected before the transmitter is enabled to ensure that the clocks function correctly. These bits should not be changed while the transmitter is enabled.

TRANSMIT DATA OUT

Transmit data out (TDO) is the serial data presented from the transmit data register (TDR) via the SCI to the output pin. The transmitter generates a bit time by using a derivative of the RT clock, producing a transmission rate equal to one-sixteenth that of the receiver sample clock (if the same baud rate is used for transmit and receive).

FUNCTIONAL DESCRIPTION

A block diagram of the SCI is shown in Figure 12. The user has option bits in the serial communications control register 1 (SCCR1) to determine the SCI wake-up method and data word length. Serial communications control register 2 (SCCR2) provides control bits that individually enable/disable the transmitter or receiver, enable system interrupts, and provide wake-up enable, and send break code bits. The baud rate register bits allow the user to select different baud rates for the transmitter and receiver.

Data transmission is initiated by a write to the serial communications data register (SCDAT). Provided the transmitter is enabled, data stored in the SCDAT is transferred to the transmit data shift register. This data transfer sets the SCI status register (SCSR) transmit data register empty (TDRE) bit and generates an interrupt if the transmit interrupt is enabled. Data transfer to the transmit data shift register is synchronized with the bit rate clock. All data is transmitted LSB first. Upon completion of data transmission, the transmission complete (TC) bit is set (provided no pending data, preamble, or break code is sent), and an interrupt is generated if the transmit complete interrupt is enabled. If the transmitter is disabled, and the data, preamble, or break code has been sent, the TC bit will also be set, which will also generate an interrupt if the TCIE bit is set. If the transmitter is disabled in the middle of a transmission, that character will be completed before the transmitter gives up control of the TDO pin.

When the SCDAT is read, it contains the last data byte received, provided that the receiver is enabled. The SCSR receive data register full (RDRF) bit is set to indicate that a data byte is transferred from the input serial shift register to the SCDAT, which can cause an interrupt if the receiver interrupt is enabled. Data transfer from the input serial shift register to the SCDAT is synchronized by the receiver bit rate clock. The SCSR overrun (OR), noise flag (NF), or FE bits are set if data reception errors occur.

An idle line interrupt is generated if the idle line interrupt is enabled and the SCSR IDLE bit (which detects idle line transmission) is set. This allows a receiver that is not in the wake-up mode to detect the end of a message, the preamble of a new message, or to resynchronize with the transmitter. A valid character must be received before the idle line condition for the IDLE bit to be set and for an idle line interrupt to be generated.

REGISTERS

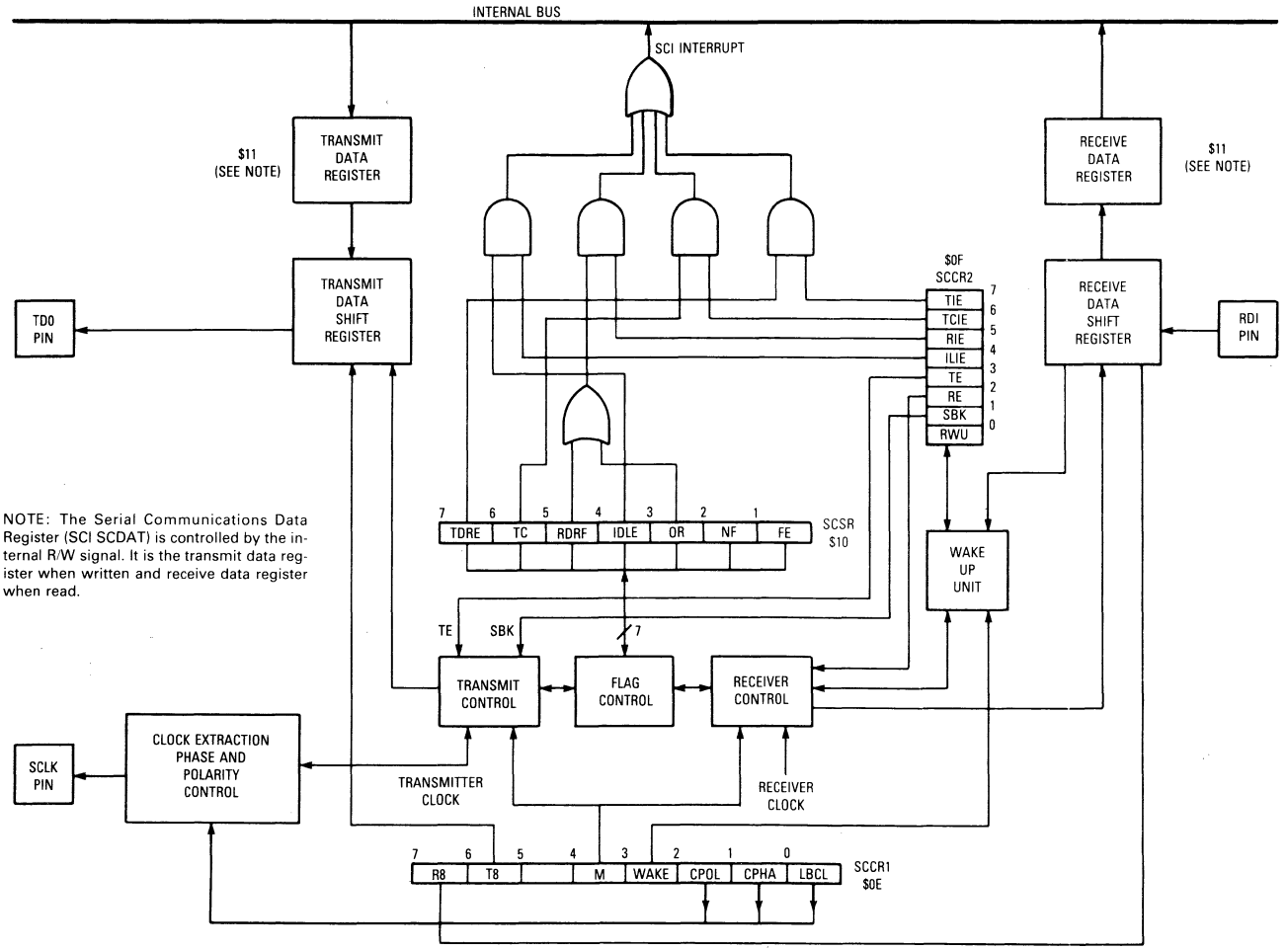
There are five registers used in the SCI; the internal configuration of these registers is discussed in the following paragraphs.

Serial Communications Data Register (SCDAT) $11

The SCDAT is a read/write register used to receive and transmit SCI data.

7	6	5	4	3	2	1	0
SCD7	SCD6	SCD5	SCD4	SCD3	SCD2	SCD1	SCD0
RESET: U	U	U	U	U	U	U	U

As shown in Figure 12, SCDAT functions as two separate registers. The transmit data register (TDR) provides the parallel interface from the internal data bus to the transmit shift register. The receive data register (RDR)



NOTE: The Serial Communications Data Register (SCI SCDAT) is controlled by the internal R/W signal. It is the transmit data register when written and receive data register when read.

Figure 12. SCI Block Diagram

provides the interface from the receive shift register to the internal data bus.

Serial Communications Control Register 1 (SCCR1) $0E

The SCCR1 provides control bits that determine word length, select the wake-up method, and control the options to output the transmitter clocks for synchronous transmissions.

7	6	5	4	3	2	1	0
R8	T8	—	M	WAKE	CPOL	CPHA	LBCL
RESET:							
U	U	—	U	U	U	U	U

R8 — Receive Data Bit 8

R8 bit provides storage location for the ninth bit in the receive data byte (if M = 1).

T8 — Transmit Data Bit 8

T8 bit provides storage location for the ninth bit in the transmit data byte (if M = 1).

M — SCI Character Word Length

1 = one start bit, nine data bits, one stop bit

0 = one start bit, eight data bits, one stop bit

WAKE — Wake-Up Select

Wake bit selects the receiver wake-up method.

1 = Address bit (most significant bit)

0 = Idle line condition

CPOL — Clock Polarity

Selects the clock polarity sent to the SCLK pin.

1 = Steady state high outside the transmission window

0 = Steady state low outside the transmission window

The CPOL bit should not be changed with the transmitter active.

CPHA — Clock Phase

Selects the clock phase sent to the SCLK pin.

1 = SCLK line activated at the beginning of the data bit

0 = SCLK line activated in the middle of the data bit (see Figures 13 and 14)

The CPHA bit should not be changed with the transmitter active.

LBCL — Last Bit Clock

Selects whether the clock associated with the last data bit transmitted is output to the SCLK pin.

1 = Last data bit output

0 = Last data bit not output

The last data bit is the eighth or ninth bit, depending on whether an 8- or 9-bit format is used (see Table 5).

The LCBL bit should not be changed while the transmitter is enabled.

Bit 5 — Not used.

Can be either 1 or 0.

Table 5. SCI Clock on SCLK Pin

Data Format	M Bit	LBCL Bit	Number of Clocks on SCLK Pin
8 Bit	0	0	7
8 Bit	0	1	8
9 Bit	1	0	8
9 Bit	1	1	9

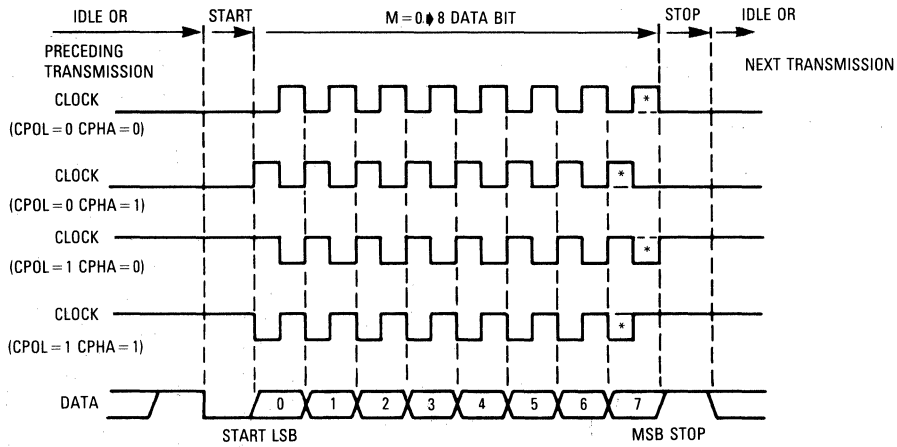


Figure 13. SCI Data Clock Timing Diagram (M = 0)

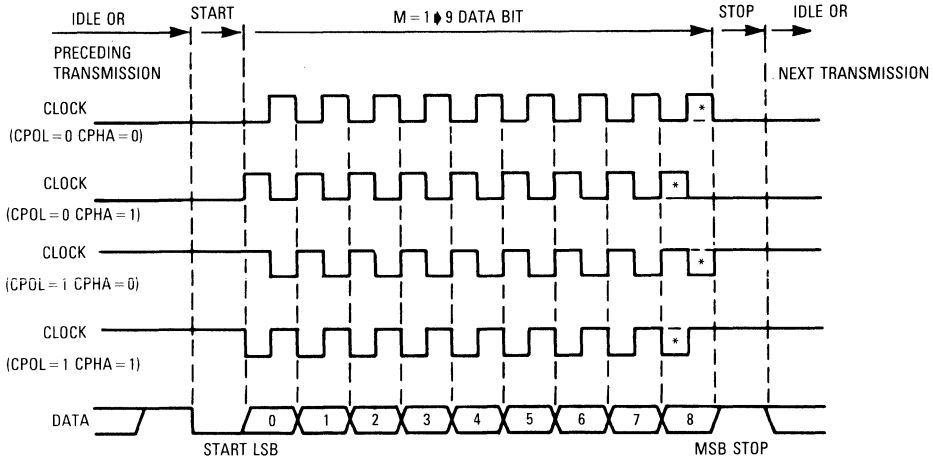


Figure 14. SCI Data CLock Timing Diagram (M = 1)

The address bit is dependent on both the wake-bit and the M-bit level. Additionally, the receiver does not use the wake-up feature unless the RWU control bit in SCCR2 is set.

Wake	M	Receiver Wake-Up
0	X	Detection of an idle line allows the next data byte received to cause the receive data register to fill and produce an RDRF flag.
1	0	Detection of a received one in the eighth data bit allows an RDRF flag and associated error flags.
1	1	Detection of a received one in the ninth data bit allows an RDRF flag and associated error flags.

Serial Communications Control Register 2 (SCCR2) $0F

The SCCR2 provides control of individual SCI functions such as interrupts, transmit/receive enabling, receiver wake-up, and break code.

7	6	5	4	3	2	1	0
TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
RESET: 0	0	0	0	0	0	0	0

TIE — Transmit Interrupt Enable
- 1 = SCI interrupt enabled, provided TDRE is set
- 0 = TDRE interrupt disabled

TCIE — Transmit Complete Interrupt Enable
- 1 = SCI interrupt enabled, provided TC is set
- 0 = TC interrupt disabled

RIE — Receive Interrupt Enable
- 1 = SCI interrupt enabled, provided OR or RDRF is set
- 0 = RDRF and OR interrupts disabled

ILIE — Idle Line Interrupt Enable
- 1 = SCI interrupt enabled, provided IDLE is set
- 0 = Idle interrupt disabled

TE — Transmit Enable
- 1 = Transmit shift register output is applied to the TD0 line, and the corresponding clocks are applied to the SCLK pin. Depending upon the SCCR1 M bit, a preamble of 10 (M = 0) or 11 (M = 1) consecutive ones is transmitted.
- 0 = Transmitter disabled after last byte is loaded in the SCDAT and TDRE is set. After last byte is transmitted, TD0 line becomes a high-impedance line.

RE — Receive Enable
- 1 = Receiver shift register input is applied to the RDI line.
- 0 = Receiver disabled and RDRF, IDLE, OR, NF, and FE status bits are inhibited.

RWU — Receiver Wake-Up
- 1 = Places receiver in sleep mode and enables wake-up function
- 0 = Wake-up function disabled after receiving data word with MSB set (if WAKE = 1)
  Wake-up function also disabled after receiving 10 (M = 0) or 11 (M = 1) consecutive ones (if WAKE = 0)

SBK — Send Break
- 1 = Transmitter continually sends blocks of zeros (sets of 10 or 11) until cleared. Upon completion of break code, transmitter sends one high bit for recognition of valid start bit.
- 0 = Transmitter sends 10 (M = 0) or 11 (M = 1) zeros then reverts to an idle state or continues sending

data. If transmitter is empty and idle, setting and clearing the SBK bit may queue up to two character times of break because the first break transfers immediately to the shift register, and the second is queued into the parallel transmit buffer.

Serial Communications Status Register (SCSR) $10

The SCSR provides inputs to the SCI interrupt logic circuits. Noise flag and framing error bits are also contained in the SCSR.

7	6	5	4	3	2	1	0
TDRE	TC	RDRF	IDLE	OR	NF	FE	—
RESET:							
1	1	0	0	0	0	0	—

TDRE — Transmit Data Register (TDR) Empty
- 1 = TDR contents transferred to the transmit data shift register
- 0 = TDR still contains data. TDRE is cleared by reading the SCSR, followed by a write to the TDR.

TC — Transmit Complete
- 1 = Indicates end of data frame, preamble, or break condition has occurred if:
  1. TE = 1, TDRE = 1, and no pending data, preamble or break is to be transmitted; or
  2. TE = 0 and the data preamble or break (in the transmit shift register) has been transmitted.
- 0 = TC bit cleared by reading the SCSR, followed by a write to the TDR

The TC bit is a status register that indicates one of the above conditions has occurred. It does not inhibit the transmitter in any way.

RDRF — Receive Data Register (RDR) Full
- 1 = Receive data shift register contents transferred to the RDR
- 0 = Receive data shift register transfer did not occur. RDRF is cleared by reading the SCSR, followed by a read of the RDR

IDLE — Idle Line Detect
- 1 = Indicates receiver has detected an idle line
- 0 = IDLE is cleared by reading the SCSR, followed by a read of the RDR. Once IDLE is cleared, IDLE cannot be set until RDI line becomes active and idle again.

OR — Overrun Error
- 1 = Indicates receive data shift register data is ready to be sent to a full RDR (RDRF = 1). Data causing the overrun is lost, and RDR data is not disturbed.
- 0 = OR is cleared by reading the SCSR, followed by a read of the RDR.

NF — Noise Flag
- 1 = Indicates noise is present on the receive bits, including the start and stop bits. NF is not set until RDRF = 1.
- 0 = NF is cleared by reading the SCSR, followed by a read of the RDR.

FE — Framing Error
- 1 = Indicates stop bit not detected in received data character. FE is set the same time RDRF is set. If received byte causes both framing and overrun errors, processor will only recognize the overrun error. Further data transfer into the RDR is inhibited until FE is cleared.
- 0 = FE is cleared by reading the SCSR, followed by a read of the RDR.

Bit 0 — Not used
Can read either one or zero

Baud Rate Register $0D

The baud rate register selects the SCI transmitter and receiver baud rate. The SCP0 and SCP1 prescaler bits are used in conjunction with the SCR2–SCR0 bits to generate the receiver baud rate and in conjunction with the SCT2–SCT0 baud rate bits to generate the transmitter baud rate.

Tables 6 and 7 tabulate the divide chain used to obtain the baud rate clock (transmit or receive clock). The actual divider chain is controlled by the combined SCP1–SCP0 and SCR2–SCR0 or SCT2–SCR0 bits in the baud rate register. The divided frequencies shown in Table 6 represent the final baud rate that results from prescaler clock division only (SCR or SCT bits all zero). Table 7 lists the prescaler output frequency divided by the action of the SCR or SCT bits.

For example, assume that a 9600-Hz baud rate is desired from a 2.4576-MHz system clock crytal. The prescaler bits could be set for either a divide-by-one or divide-by-four. If a divide-by-four prescaler is used, then the SCR and SCT bits must be set for divide-by-two. The same result, using the same crystal frequency, can be obtained with a prescaler divide-by-one and SCR and SCT bit divide-by-eight.

7	6	5	4	3	2	1	0
SCP1	SCP0	SCT2	SCT1	SCT0	SCR2	SCR1	SCR0
RESET:							
0	0	U	U	U	U	U	U

SCP1–SCP0 — SCI Prescaler Bit 1 and 0
These two prescaler bits are used to increase the range of standard baud rates controlled by the SCT2–SCT0 and SCR2–SCR0 bits. Prescaler internal processor clock division versus bit levels are shown in Table 6.

SCT2–SCT0 — SCI Transmit Baud Rate Selection Bits
These three bits, taken in conjunction with bits SCP1–SCP0, are used to select the SCI transmit baud rate. Baud rates versus bit levels are listed in Table 7.

SCR2–SCR0 — SCI Recieve Baud Rate Selection Bits
These three bits, taken in conjunction with bits SCP1–SCP0, are used to select the SCI receive baud rate. Baud rates versus bit levels are listed in Table 7.

Load Program in RAM and Execute

This function is entered if the following conditions are met when reset is released:

$\overline{IRQ}$ is at V_{DD} + 4 V for at least two machine cycles after reset

Table 6. Prescaler Highest Baud Rate Frequency Output

SCP Bit		Clock* Divided By	Crystal Frequency MHz				
1	0		4.194304	4.0	2.4576	2.0	1.8432
0	0	1	131.072 kHz	125.000 kHz	76.80 kHz	62.50 kHz	57.60 kHz
0	1	3	43.691 kHz	41.666 kHz	25.60 kHz	20.833 kHz	19.20 kHz
1	0	4	32.768 kHz	31.250 kHz	19.20 kHz	15.625 kHz	14.40 kHz
1	1	13	10.082 kHz	9600 Hz	5.907 kHz	4800 Hz	4430 Hz

*Refers to the internal processor clock.

NOTE: The divided frequencies shown in Table 6 represent baud rates which are the highest transmit baud rate (Tx) that can be obtained by a specific crystal frequency and only using the prescaler division. Lower baud rates may be obtained by providing a further division using the SCI rate select bits as shown below for some representative prescaler outputs.

Table 7. Transmit Baud Rate Output for a Given Prescaler Output

SCR/T Bits			Divided By	Representative Highest Prescaler Baud Rate Output				
2	1	0		131.072 kHz	32.768 kHz	76.80 kHz	19.20 kHz	9600 Hz
0	0	0	1	131.072 kHz	32.768 kHz	76.80 kHz	19.20 kHz	9600 Hz
0	0	1	2	65.536 kHz	16.384 kHz	38.40 kHz	9600 Hz	4800 Hz
0	1	0	4	32.768 kHz	8.192 kHz	19.20 kHz	4800 Hz	2400 Hz
0	1	1	8	16.384 kHz	4.096 kHz	9600 Hz	2400 Hz	1200 Hz
1	0	0	16	8.192 kHz	2.048 kHz	4800 Hz	1200 Hz	600 Hz
1	0	1	32	4.096 kHz	1.024 kHz	2400 Hz	600 Hz	300 Hz
1	1	0	64	2.048 kHz	512 Hz	1200 Hz	300 Hz	150 Hz
1	1	1	128	1.024 kHz	256 Hz	600 Hz	150 Hz	75 Hz

NOTE: Table 7 illustrates how the SCI select bits can be used to provide lower transmitter or receiver baud rates by further dividing the prescaler output frequency. The five examples are only representative samples. In all cases, the baud rates shown are transmit baud rates (transmit clock), and the receive clock is 16 times higher in frequency than the actual baud rate.

TCAP1 is at V_{DD} for at least two machine cycles after reset

PD3 is at V_{DD} for at least 30 machine cycles after reset

PD4 is at V_{SS} for at least 30 machine cycles after reset

User programs are loaded into RAM using the SCI port and then executed. Data is loaded sequentially, starting at RAM location $50, until the last byte is loaded. Program control is then transferred to the RAM progam starting at location $51. The first byte loaded is the count of the number of bytes in the program plus the count byte. The program starts at the second byte in the RAM. During firmware initialization, the SCI is configured for the NRZ format (idle line, eight data bits, and stop bit). The baud rate is 9600 with a 4-MHz crystal. Figure 15 shows a schematic for the load program in RAM and execute function.

Immediate execution can be avoided by setting the byte count to a value greater than the length of data loaded, which causes the firmware to wait for additional data after loading is complete. Resetting the MCU then allows entering any routine without disturbing the RAM data that was loaded.

Jump to Any Address

This function is entered if the following conditions are met when reset is released:

$\overline{IRQ}$ is at V_{DD} + 4 V for at least two machine cycles after reset

TCAP1 is at V_{DD} for at least two machine cycles after reset

PD3 is at V_{DD} for at least 30 machine cycles after reset

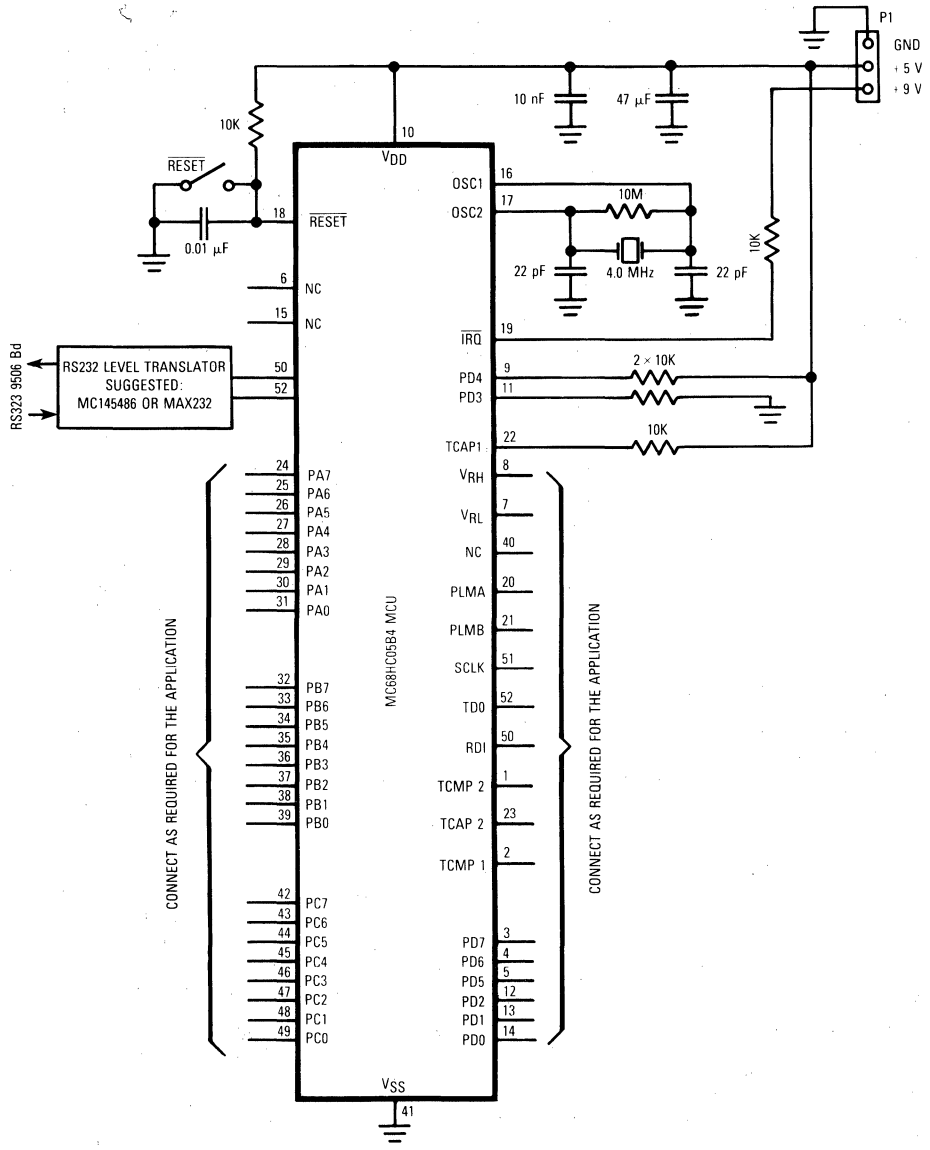
PD4 is at V_{DD} for at least 30 machine cycles after reset

To execute the jump to any address function, port A data input should be $CC, and port B and C should be the MSB and LSB, respectively, of the address desired for the jump. Figure 16 shows a schematic for the jump function.

PULSE-LENGTH D/A CONVERTERS

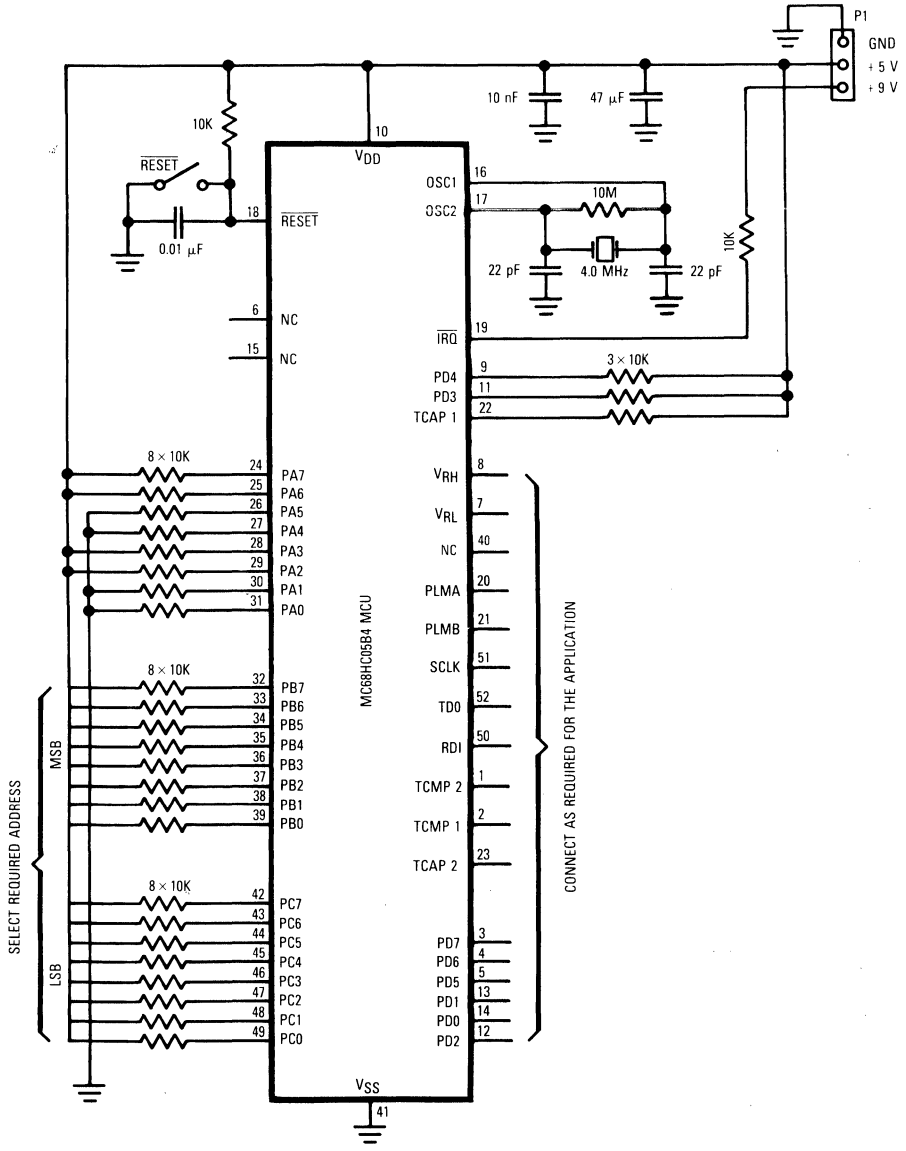
The pulse-length D/A converter (PLM) works in conjunction with the timer to execute two 8-bit conversions with a choice of two repetition rates. The outputs are pulse-length modulated signals whose duty-cycle ratio may be modified. These signals can be used directly as PLMS, or the filtered average values can be used as general-purpose analog outputs.

Registers PLMA and PLMB contain the pulse-length values for the two PLMs. A value of $00 results in a continuously low output from the D/A. A value of $80 results in a 50-percent duty-cycle output, and a value of $FF gives an output that is a logic 1 for 255/256 of the cycle. When the MCU writes to the PLMA or PLMB register, the D/A picks up the new value at the end of a complete conversion cycle so that a monotonic change in the dc component of the output results. This monotonic change avoids overshoots or vicious starts (a vicious start is an output that gives totally erroneous output during the first



NOTE: Pin numbers are valid for 52-pin PLCC package only.

Figure 15. Load Program in RAM and Execute Diagram



NOTE: Pin numbers are valid for 52-Pin PLCC Package only.

Figure 16. Jump to Any Address Diagram

cycle following an update of the registers). WAIT mode does not affect the output waveform of the D/A converters.

NOTE

Since the PLM system uses the timer counter, PLM results will be affected while resetting the timer counter.

Figure 17 shows a block diagram of the PLM system.

PLMA (0A)

7	6	5	4	3	2	1	0
PLMA7	PLMA6	PLMA5	PLMA4	PLMA3	PLMA2	PLMA1	PLMA0
RESET:							
0	0	0	0	0	0	0	0

PLMB (0B)

7	6	5	4	3	2	1	0
PLMB7	PLMB6	PLMB5	PLMB4	PLMB3	PLMB2	PLMB1	PLMB0
RESET:							
0	0	0	0	0	0	0	0

Miscellaneous (0C)

7	6	5	4	3	2	1	0
—	—	—	—	SFA	SFB	—	—
RESET:							
—	—	—	—	0	0	—	—

SFA — Slow/Fast Control for PLMA Clock
1 = Slow speed of PLMA used (4096 times the timer clock period)
0 = Fast speed of PLMA used (256 times the timer clock period)

SFB — Slow/Fast Control for PLMB Clock
1 = Slow speed of PLMB used (4096 times the timer clock period)
0 = Fast speed of PLMB used (256 times the timer clock period)

NOTE

The highest speed of the PLM system corresponds to the frequency of the TOF bit being set, multiplied by 256. The slowest speed of the PLM system corresponds to the frequency of the TOF bit being set, multipled by 16.

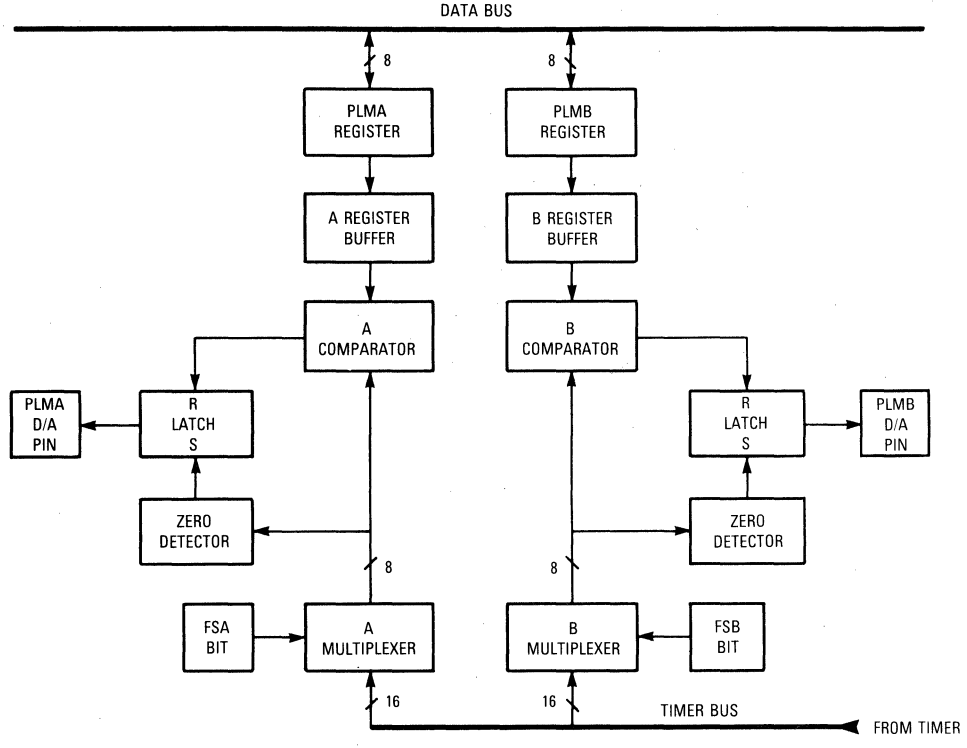


Figure 17. PLM Block Diagram

The SFA and SFB bits are not double buffered; therefore, these bits must be selected before writing to either PLM register to avoid temporary wrong values from the PLM outputs. Figure 18 shows some examples of the PLM output waveforms.

A/D CONVERTER

The A/D converter system consists of an 8-bit successive approximation converter and a 16-channel multiplexer. Eight of the channels are available for output, and the other eight channels are dedicated to internal test functions. There is one 8-bit result data register (address $08) and one 8-bit status/control register (address $09).

NOTE

In the 48-pin dual-in-line package, the fixed input port (D) of the MC68HC05B4 is reduced to six pins (PD5–PD0, AN5–AN0). This change has not effect on either programming or operating of port D or the A/D converter.

The reference supply for the converter uses dedicated input pins instead of the power supply lines, because drops caused by loading in the power supply lines would degrade the accuracy of the A/D conversion. An internal RC oscillator is available if the bus speed is low enough to degrade the A/D accuracy. An ADON bit allows the A/D to be switched off to reduce power consumption, which is particularly useful in the WAIT mode.

For ratiometric conversions, the source of each analog input should use V_{RH} as the supply voltage and be referenced to V_{RL}. An input voltage greater than or equal to V_{RH} converts as $FF (full scale) with no overflow indication. An input voltage equal to V_{RL} converts as $00. The conversion is monotonic with no missing codes.

A/D STATUS/CONTROL REGISTER ($09)

7	6	5	4	3	2	1	0
COCO	ADRC	ADON	0	CH3	CH2	CH1	CH0
RESET:							
0	0	0	0	0	0	0	0

COCO — Conversion Complete
- 1 = Conversion is complete; a new result can be read from the result data register ($08).
- 0 = No conversion since last reset

ADRC — A/D RC Oscillator Control
- 1 = A/D uses RC clock
- 0 = A/D uses CPU clock

When the RC oscillator is turned on, it requires a time t_{adrc} to stabilize, and results can be inaccurate during this time.

ADON — A/D On
- 1 = A/D enabled
- 0 = A/D disabled

When the A/D is turned on, it requires a time t_{ADON} for the current sources to stabilize, and results can be inaccurate during this time.

CH3–CH0 — Channel 3 through Channel 0

These bits select the A/D channel assignment (see Table 8).

NOTE

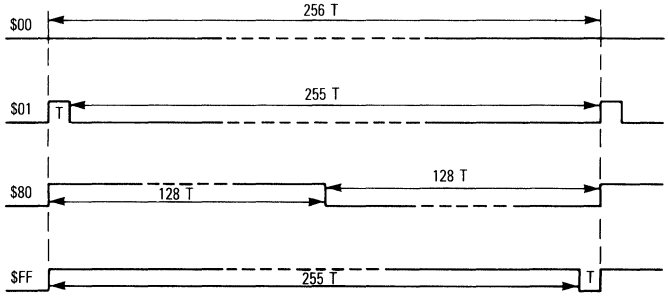
Using one or more pins of PD7/AN7–PD0/AN0 as analog inputs does not affect the ability to use port D inputs as digital inputs. However, using port D for digital inputs during an analog conversion sequence may inject noise on the analog inputs and reduce the accuracy of the A/D result.

Performing a digital read of port D with levels other than V_{DD} or V_{SS} on the inputs causes greater than normal power dissipation during the read and may give erroneous results.

INSTRUCTION SET

The MCU instruction set can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

This MCU uses all the instructions available in the M146805 CMOS Family plus one more: the unsigned



T = 4 CPU Clocks Fast Mode
T = 64 CPU Clocks Slow Mode

Figure 18. PLM Output Waveform Examples

Table 8. A/D Channel Assignments

CH3	CH2	CH1	CH0	Channel Selected
0	0	0	0	AN0, Port D Bit 0
0	0	0	1	AN1, Port D Bit 1
0	0	1	0	AN2, Port D Bit 2
0	0	1	1	AN3, Port D Bit 3
0	1	0	0	AN4, Port D Bit 4
0	1	0	1	AN5, Port D Bit 5
0	1	1	0	AN6, Port D Bit 6
0	1	1	1	AN7, Port D Bit 7
1	0	0	0	V_{RH} Pin (High)
1	0	0	1	$((V_{RH}) + (V_{RL}))/2$
1	0	1	0	V_{RL} Pin (Low)
1	0	1	1	V_{RL} Pin (Low)
1	1	0	0	V_{RL} Pin (Low)
1	1	0	1	V_{RL} Pin (Low)
1	1	1	0	V_{RL} Pin (Low)
1	1	1	1	V_{RL} Pin (Low)

3

multiply (MUL) instruction. This instruction allows unsigned multiplication of the contents of the accumulator (A) and the index register (X). The high-order product is then stored in the index register, and the low-order product is stored in the accumulator. A detailed definition of the MUL instruction is shown below.

Operation	X:A ◊ X × A		
Description	Multiplies the eight bits in the index register by the eight bits in the accumulator to obtain a 16-bit unsigned number in the concatenated accumulator and index register		
Condition Codes	H: Cleared I: Not affected N: Not affected Z: Not affected C: Cleared		
Source Form(s)	MUL		
Addressing Mode	Cycles	Bytes	Opcode
Inherent	11	1	$42

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following instruction list.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two-byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Higher	BHI
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
Branch if Higher or Same	BHS
Branch if Carry Set	BCS
Branch if Lower	BLO
Branch if Not Equal	BNE
Branch if Equal	BEQ
Branch if Half Carry Clear	BHCC
Branch if Half Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Bit is Clear	BMC
Branch if Interrupt Mask Bit is Set	BMS
Branch if Interrupt Line is Low	BIL
Branch if Interrupt Line is High	BIH
Branch to Subroutine	BSR

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any writable bit which resides in the first 256 bytes of the memory space where all port registers, port DDRs, timer, timer control, ROM, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear and bit test, and branch functions are all implemented with a single instruction. For test and branch instructions,

the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch if Bit n is Set	BRSET n (n 0 . . . 7)
Branch if Bit n is Clear	BRCLR n (n 0 . . . 7)
Set Bit n	BSET n (n 0 . . . 7)
Clear Bit n	BCLR n (n 0 . . . 7)

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (Twos Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST
Multiply	MUL

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP

Function	Mnemonic
No-Operation	NOP
Stop	STOP
Wait	WAIT

OPCODE MAP SUMMARY

Table 9 is an opcode map for the instructions used on the MCU.

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single byte instructions; the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. One- or two-byte direct addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction. When using the Motorola assembler, the user need not specify whether an instruction uses direct or extended addressing. The assembler automatically selects the shortest form of the instruction.

RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added

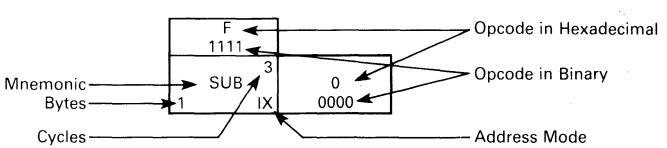
Table 9. Opcode Map

	Bit Manipulation		Branch	Read-Modify-Write					Control		Register/Memory						
	BTB	BSC	REL	DIR	INH	INH	IX1	IX	INH	INH	IMM	DIR	EXT	IX2	IX1	IX	
HI / LOW	0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111	8 1000	9 1001	A 1010	B 1011	C 1100	D 1101	E 1110	F 1111	HI / LOW
0 0000	BRSET0 5 3 BTB	BSET0 5 2 BSC	BRA 3 2 REL	NEG 5 2 DIR	NEGA 3 1 INH	NEGX 3 1 INH	NEG 6 2 IX1	NEG 5 1 IX	RTI 9 1 INH		SUB 2 2 IMM	SUB 3 2 DIR	SUB 4 3 EXT	SUB 5 3 IX2	SUB 4 2 IX1	SUB 3 1 IX	0 0000
1 0001	BRCLR0 5 3 BTB	BCLR0 5 2 BSC	BRN 3 2 REL						RTS 6 1 INH		CMP 2 2 IMM	CMP 3 2 DIR	CMP 4 3 EXT	CMP 5 3 IX2	CMP 4 2 IX1	CMP 3 1 IX	1 0001
2 0010	BRSET1 5 3 BTB	BSET1 5 2 BSC	BHI 3 2 REL		MUL 11 1 INH						SBC 2 2 IMM	SBC 3 2 DIR	SBC 4 3 EXT	SBC 5 3 IX2	SBC 4 2 IX1	SBC 3 1 IX	2 0010
3 0011	BRCLR1 5 3 BTB	BCLR1 5 2 BSC	BLS 3 2 REL	COM 5 2 DIR	COMA 3 1 INH	COMX 3 1 INH	COM 6 2 IX1	COM 5 1 IX	SWI 10 1 INH		CPX 2 2 IMM	CPX 3 2 DIR	CPX 4 3 EXT	CPX 5 3 IX2	CPX 4 2 IX1	CPX 3 1 IX	3 0011
4 0100	BRSET2 5 3 BTB	BSET2 5 2 BSC	BCC 3 2 REL	LSR 5 2 DIR	LSRA 3 1 INH	LSRX 3 1 INH	LSR 6 2 IX1	LSR 5 1 IX			AND 2 2 IMM	AND 3 2 DIR	AND 4 3 EXT	AND 5 3 IX2	AND 4 2 IX1	AND 3 1 IX	4 0100
5 0101	BRCLR2 5 3 BTB	BCLR2 5 2 BSC	BCS 3 2 REL								BIT 2 2 IMM	BIT 3 2 DIR	BIT 4 3 EXT	BIT 5 3 IX2	BIT 4 2 IX1	BIT 3 1 IX	5 0101
6 0110	BRSET3 5 3 BTB	BSET3 5 2 BSC	BNE 3 2 REL	ROR 5 2 DIR	RORA 3 1 INH	RORX 3 1 INH	ROR 6 2 IX1	ROR 5 1 IX			LDA 2 2 IMM	LDA 3 2 DIR	LDA 4 3 EXT	LDA 5 3 IX2	LDA 4 2 IX1	LDA 3 1 IX	6 0110
7 0111	BRCLR3 5 3 BTB	BCLR3 5 2 BSC	BEQ 3 2 REL	ASR 5 2 DIR	ASRA 3 1 INH	ASRX 3 1 INH	ASR 6 2 IX1	ASR 5 1 IX		TAX 2 1 INH		STA 4 2 DIR	STA 5 3 EXT	STA 6 3 IX2	STA 5 2 IX1	STA 4 1 IX	7 0111
8 1000	BRSET4 5 3 BTB	BSET4 5 2 BSC	BHCC 3 2 REL	LSL 5 2 DIR	LSLA 3 1 INH	LSLX 3 1 INH	LSL 6 2 IX1	LSL 5 1 IX		CLC 2 1 INH	EOR 2 2 IMM	EOR 3 2 DIR	EOR 4 3 EXT	EOR 5 3 IX2	EOR 4 2 IX1	EOR 3 1 IX	8 1000
9 1001	BRCLR4 5 3 BTB	BCLR4 5 2 BSC	BHCS 3 2 REL	ROL 5 2 DIR	ROLA 3 1 INH	ROLX 3 1 INH	ROL 6 2 IX1	ROL 5 1 IX		SEC 2 1 INH	ADC 2 2 IMM	ADC 3 2 DIR	ADC 4 3 EXT	ADC 5 3 IX2	ADC 4 2 IX1	ADC 3 1 IX	9 1001
A 1010	BRSET5 5 3 BTB	BSET5 5 2 BSC	BPL 3 2 REL	DEC 5 2 DIR	DECA 3 1 INH	DECX 3 1 INH	DEC 6 2 IX1	DEC 5 1 IX		CLI 2 1 INH	ORA 2 2 IMM	ORA 3 2 DIR	ORA 4 3 EXT	ORA 5 3 IX2	ORA 4 2 IX1	ORA 3 1 IX	A 1010
B 1011	BRCLR5 5 3 BTB	BCLR5 5 2 BSC	BMI 3 2 REL							SEI 2 1 INH	ADD 2 2 IMM	ADD 3 2 DIR	ADD 4 3 EXT	ADD 5 3 IX2	ADD 4 2 IX1	ADD 3 1 IX	B 1011
C 1100	BRSET6 5 3 BTB	BSET6 5 2 BSC	BMC 3 2 REL	INC 5 2 DIR	INCA 3 1 INH	INCX 3 1 INH	INC 6 2 IX1	INC 5 1 IX		RSP 2 1 INH		JMP 2 2 DIR	JMP 3 3 EXT	JMP 4 3 IX2	JMP 3 2 IX1	JMP 2 1 IX	C 1100
D 1101	BRCLR6 5 3 BTB	BCLR6 5 2 BSC	BMS 3 2 REL	TST 4 2 DIR	TSTA 3 1 INH	TSTX 3 1 INH	TST 5 2 IX1	TST 4 1 IX		NOP 2 1 INH	BSR 6 2 REL	JSR 5 2 DIR	JSR 6 3 EXT	JSR 7 3 IX2	JSR 6 2 IX1	JSR 5 1 IX	D 1101
E 1110	BRSET7 5 3 BTB	BSET7 5 2 BSC	BIL 3 2 REL						STOP 2 1 INH		LDX 2 2 IMM	LDX 3 2 DIR	LDX 4 3 EXT	LDX 5 3 IX2	LDX 4 2 IX1	LDX 3 1 IX	E 1110
F 1111	BRCLR7 5 3 BTB	BCLR7 5 2 BSC	BIH 3 2 REL	CLR 5 2 DIR	CLRA 3 1 INH	CLRX 3 1 INH	CLR 6 2 IX1	CLR 5 1 IX	WAIT 2 1 INH	TXA 2 1 INH		STX 4 2 DIR	STX 5 3 EXT	STX 6 3 IX2	STX 5 2 IX1	STX 4 1 IX	F 1111

Abbreviations for Address Modes

INH	Inherent	REL	Relative
A	Accumulator	BSC	Bit Set/Clear
X	Index Register	BTB	Bit Test and Branch
IMM	Immediate	IX	Indexed (No Offset)
DIR	Direct	IX1	Indexed, 1 Byte (8-Bit) Offset
EXT	Extended	IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −126 to +129 from the opcode address. The programmer need not calculate the offset when using the Motorola assembler, since it calculates the proper offset and checks to see that it is within the span of the branch.

INDEXED, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. This addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this two-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This address mode can be used in a manner similar to indexed, 8-bit offset except that this three-byte instruction allows tables to be anywhere in memory. As with direct and extended addressing, the Motorola assembler determines the shortest form of indexed addressing.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode, and the byte following the opcode specifies the direct addressing of the byte in which the specified bit is to be set or cleared. Any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single two-byte instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit that is to be tested and its condition (set or clear), is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

3

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS (Voltages referenced to V_{SS})

Rating	Symbol	Value	Unit
Supply Voltage	V_{DD}	−0.5 to +7.0	V
Input Voltage	V_{in}	V_{SS} −0.5 to V_{DD} +0.5	V
Self-Check Mode ($\overline{IRQ}$ Pin Only)	V_{in}	V_{SS} −0.5 to $2 \times V_{DD}$ +0.5	V
Current Drain Per Pin Excluding V_{DD} and V_{SS}	I	25	mA
Operating Temperature Range MC68HC05B4P, FN (Standard) MC68HC05B4CP, CFN (Extended) MC68HC05B4MP, MFN (Automotive)	T_A	T_L to T_H 0 to +70 −40 to +85 −40 to +125	°C
Storage Temperature Range	T_{stg}	−65 to +150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation, it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{DD}$. Reliability of operation is enhanced if unused inputs are connected to an appropriate logic voltage level (e.g., either V_{SS} or V_{DD}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic Plastic Leaded Chip Carrier (PLCC)	θ_{JA}	 40 50	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{I/O}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
$P_{I/O}$ = Power Dissipation on Input and Output Pins — User Determined

For most applications $P_{I/O} < P_{INT}$ and can be neglected.

The following is an approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected):

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

V_{DD} = 4.5 V

Pins	R1	R2	C
PA7–PA0, PB7–PB0, PC7–PC0, TCMP1 TCMP2	3.26 kΩ	2.38 kΩ	50 pF
TDO, SCLK, PLMA, PLMB	1.9 kΩ	2.26 kΩ	200 pF

V_{DD} = 3.0 V

Pins	R1	R2	C
PA7–PA0, PB7–PB0, PC7–PC0, TCMP1, TCMP2	10.91 kΩ	6.32 kΩ	50 pF
TDO, SCLK, PLMA, PLMB	6 kΩ	6 kΩ	200 pF

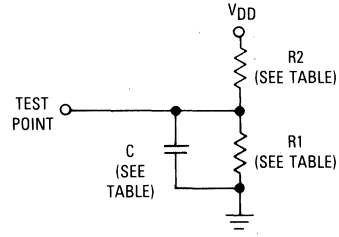


Figure 19. Equivalent Test Load

DC ELECTRICAL CHARACTERISTICS (V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = T_L to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage, I_{Load} = 10.0 μA	V_{OL} V_{OH}	— V_{DD} – 0.1	— —	0.1 —	V
Output High Voltage (I_{Load} = 0.8 mA) PA7–PA0, PB7–PB0, PC7–PC0, TCMP1, TCMP2 (I_{Load} = 1.6 mA) TDO, SCLK, PLMA, PLMB	V_{OH}	 V_{DD} – 0.8 V_{DD} – 0.8	 V_{DD} – 0.4 V_{DD} – 0.4	 — —	V
Output Low Voltage (I_{Load} = 1.6 mA) PA7–PA0, PB7–PB0, PC7–PC0, TCMP1, TCMP2, PLMA, PLMB, TDO, SCLK $\overline{RESET}$	V_{OL}	 — —	 0.1 0.4	 0.4 1.0	V
Input High Voltage PA7–PA0, PB7–PB0, PC7–PC0, PD7–PD0, TCAP1, TCAP2, $\overline{IRQ}$, $\overline{RESET}$, OSC1, RDI	V_{IH}	0.7 × V_{DD}	—	V_{DD}	V
Input Low Voltage PA7–PA0, PB7–PB0, PC7–PC0, PD7–PD0, TCAP1, TCAP2, $\overline{IRQ}$, $\overline{RESET}$, OSC1, RDI	V_{IL}	V_{SS}	—	0.2 × V_{DD}	V
Supply Current (see Notes) RUN (SM = 0) RUN (SM = 1, t_{cyc} = 8 μs) WAIT (SM = 0) WAIT (SM = 1, t_{cyc} = 8 μs) STOP 0 to 70 (Standard) – 40 to 85 (Extended) – 40 to 125 (Automotive)	I_{DD}	 — — — — — — —	 3.5 0.5 1 0.35 2 — —	 9 2 4 1 10 20 50	 mA mA mA mA μA μA μA
I/O Ports Hi-Z Leakage Current PA7–PA0, PB7–PB0, PC7–PC0, TDO, $\overline{RESET}$, SCLK	I_{IL}	—	± 0.2	± 1	μA
Input Current $\overline{IRQ}$, TCAP1, TCAP2, OSC1, RDI PD7/AN7–PD0/AN0 (A/D off) PD7/AN7–PD0/AN0 (A/D on)	I_{in}	 — — —	 ± 0.2 ± 0.2 ± 10	 ± 1 ± 1 TBD	μA
Capacitance Ports (as Input or Output), $\overline{RESET}$ TDO, SCLK $\overline{IRQ}$, TCAP1, TCAP2, OSC1, RDI PD7/AN7–PD0/AN0 (A/D off) PD7/AN7–PD0/AN0 (A/D on)	 C_{out} C_{out} C_{in} C_{in} C_{in}	 — — — — —	 — — — 12 22	 12 12 8 TBD TBD	pF

NOTES:
1. All values shown reflect average measurements.
2. Typical values at midpoint of voltage range, 25°C only.
3. Wait I_{DD}: Only timer system active (TE = RE = 0). If SCI active (TE = RE = 1) add 10% current draw.
4. Run (Operating) I_{DD}, Wait I_{DD}: Measured using external square wave clock source (f_{osc} = 4.0 MHz), all inputs 0.2 V from rail; no dc loads, less than 50 pF on all outputs, C_L = 20 pF on OSC2.
5. Wait, Stop I_{DD}: All ports configured as inputs, V_{IL} = 0.2 V, V_{IH} = V_{DD} – 0.2 V.
6. Wait I_{DD} is affected linearly by the OSC2 capacitance.

TBD = To be decided.

DC ELECTRICAL CHARACTERISTICS (V_{DD} = 3.3 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage, $I_{Load} \leq 10.0\ \mu A$	V_{OL} V_{OH}	— $V_{DD}-0.1$	— —	0.1 —	V
Output High Voltage (I_{Load} = 0.2 mA) PA7–PA0, PB7–PB0, PC7–PC0, TCMP1, TCMP2 (I_{Load} = 0.4 mA) TDO, SCLK, PLMA, PLMB	V_{OH}	 $V_{DD}-0.3$ $V_{DD}-0.3$	 $V_{DD}-0.1$ $V_{DD}-0.1$	 — —	V
Output Low Voltage (I_{Load} = 0.4 mA) PA7–PA0, PB7–PB0, PC7–PC7, TCMP1, TCMP2, PLMA, PLMB, TDO, SCLK RESET	V_{OL}	 — —	 0.1 0.2	 0.3 0.6	V
Input High Voltage PA7–PA0, PB7–PB0, PC7–PC0, PD7–PD0, TCAP1, TCAP2, IRQ, RESET, OSC1, RDI	V_{IH}	$0.7 \times V_{DD}$	—	V_{DD}	V
Input Low Voltage PA7–PA0, PB7–PB0, PC7–PC0, PD7–PD0, TCAP1, TCAP2, IRQ, RESET, OSC1, RDI	V_{IL}	V_{SS}	—	$0.2 \times V_{DD}$	V
Supply Current (see Notes) RUN (SM = 0) RUN (SM = 1, t_{cyc} = 8 µs) WAIT (SM = 0) WAIT (SM = 1, t_{cyc} = 8 µs) STOP 0 to 70 (Standard) −40 to 85 (Extended) −40 to 125 (Automotive)	I_{DD}	 — — — — — — —	 1.2 0.2 0.4 0.15 1 — —	 5 1 2 0.5 10 10 30	 mA mA mA mA µA µA µA
I/O Ports Hi-Z Leakage Current PA7–PA0, PB7–PB0, PC7–PC0, TDO, RESET, SCLK	I_{IL}	—	±0.2	±1	µA
Input Current IRQ, TCAP1, TCAP2, OSC1, RDI PD7/AN7–PD0/AN0 (A/D off) PD7/AN7–PD0/AN0 (A/D on)	I_{in}	 — — —	 ±0.2 ±0.2 ±10	 ±1 ±1 TBD	µA
Capacitance Ports (as Input or Output), RESET, TDO TDO, SCLK IRQ, TCAP1, TCAP2, OSC1, RDI PD7/AN7–PD0/AN0 (A/D off) PD7/AN7–PD0/AN0 (A/D on)	 C_{out} C_{out} C_{in} C_{in} C_{in}	 — — — — —	 — — — 12 22	 12 12 8 TBD TBD	pF

NOTES:
1. All values shown reflect average measurements.
2. Typical values at midpoint of voltage range, 25°C only.
3. Wait I_{DD}: Only timer system active (TE = RE = 0). If SCI active (TE = RE = 1) add 10% current draw.
4. Run (Operating) I_{DD}, Wait I_{DD}: Measured using external square wave clock source (f_{osc} = 4.0 MHz), all inputs 0.2 V from rail; no dc loads, less than 50 pF on all outputs, C_L = 20 pF on OSC2.
5. Wait, Stop I_{DD}: All ports configured as inputs, V_{IL} = 0.2 V, $V_{IH} = V_{DD} - 0.2$ V.
6. Wait I_{DD} is affected linearly by the OSC2 capacitance.

TBD = To be decided.

A/D CONVERTER CHARACTERISTICS (V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc)

Characteristic	Parameter	Min	Max	Unit
Resolution	Number of bits resolved by the A/D	8	—	Bit
Non-Linearity	Maximum deviation from the best straight line through the A/D transfer characteristics ($V_{RH} = V_{DD}$ and $V_{RL} = 0$ V)	—	± ½	LSB
Quantization Error	Uncertainty due to converter resolution	—	± ½	LSB
Absolute Accuracy	Difference between the actual input voltage and the full-scale equivalent of the binary code output code for all errors	—	± 1	LSB
Conversion Range	Analog input voltage range	V_{RL}	V_{RH}	V
V_{RH}	Maximum analog reference voltage	V_{RL}	V_{DD} + 0.1	V
V_{RL}	Minimum analog reference voltage	V_{SS} − 0.1	V_{RH}	V
Conversion Time	Total time to perform a single analog-to-digital conversion a. External Clock (XTAL, EXTAL) b. Internal RC oscillator	 — —	 32 32	 t_{cyc} µs
Monotonicity	Conversion result never decreases with an increase in input voltage and has no missing codes	Guaranteed		
Zero-Input Reading	Conversion result when $V_{in} = V_{RL}$	00	—	Hex
Full-Scale Reading	Conversion result when $V_{in} = V_{RH}$	—	FF	Hex
Sample Acquisition Time (see Note 1)	Analog input acquisition sampling a. External Clock (XTAL, EXTAL) b. Internal RC oscillator	 — —	 12 12	 t_{cyc} µs
Sample/Hold Capacitance	Input capacitance on PD7/AN7–PD0/AN0	—	12	pF
Input Leakage (see Note 2)	Input leakage on A/D pins PD7/AN7–PD0/AN0, V_{RL}, V_{RH}	— —	1 1	µA

NOTES:
1. Source impedances greater than 10K ohm will adversely affect internal RC charging time during input sampling.
2. The external system error caused by input leakage current is approximately equal to the product of R source and input current.

3

CONTROL TIMING (V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation Crystal Option External Clock Option	f_{osc}	 — dc	 4.2 4.2	MHz
Internal Operating Frequency Crystal (f_{osc}/2) External Clock (f_{osc}/2)	f_{op}	 — dc	 2.1 2.1	MHz
Cycle Time (see Figure 21)	t_{cyc}	480	—	ns
Crystal Oscillator Startup Time (see Figure 21)	t_{OXOV}	—	100	ms
Stop Recovery Startup Time (Crystal Oscillator)	t_{ILCH}	—	100	ms
External RESET Input Pulse Width (see Figure 21)	t_{RL}	1.5	—	t_{cyc}
Power-On RESET Output Pulse Width 4064 Cycle Option 16 Cycle Option	t_{PORL}	 4064 16	 — —	t_{cyc}
Watchdog RESET Output Pulse Width	t_{DOGL}	1.5	—	t_{cyc}
Watchdog Time-Out	t_{DOG}	6144	7168	t_{cyc}
Timer Resolution** Input Capture Pulse Width (see Figure 20) Input Capture Pulse Period (see Figure 20)	 t_{RESL} t_{TH}, t_{TL} t_{TL}, t_{TL}	 4.0 125 ***	 — — —	 t_{cyc} ns t_{cyc}
Interrupt Pulse Width (Edge-Triggered)	t_{ILIH}	125	—	ns
Interrupt Pulse Period	t_{ILIL}	*	—	t_{cyc}
OSC1 Pulse Width	t_{OH}, t_{OL}	90	—	ns

NOTES:

*The minimum period t_{ILIL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus 21 t_{cyc}.

**Since a 2-bit prescaler in the timer must count four internal cycles (t_{cyc}), this is the limiting minimum factor in determining the timer resolution.

***The minimum period t_{TLTL} should not be less than the number of cycle times it takes to execute the capture interrupt service routine plus 24 t_{cyc}.

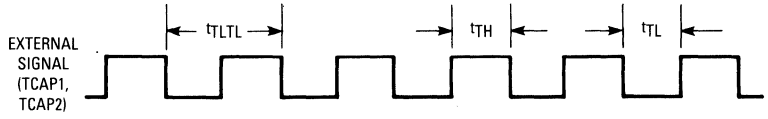


Figure 20. Timer Relationship

CONTROL TIMING (V_{DD} = 3.3 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation Crystal Option External Clock Option	f_{osc}	 — dc	 4.2 4.2	MHz
Internal Operating Frequency Crystal (f_{osc}/2) External Clock (f_{osc}/2)	f_{op}	 — dc	 2.1 2.1	MHz
Cycle Time (see Figure 21)	t_{cyc}	1000	—	ns
Crystal Oscillator Startup Time (see Figure 21)	t_{OXOV}	—	100	ms
Stop Recovery Startup Time (Crystal Oscillator)	t_{ILCH}	—	100	ms
External RESET Input Pulse Width (see Figure 21)	t_{RL}	1.5	—	t_{cyc}
Power-On RESET Output Pulse Width 4064 Cycle Option 16 Cycle Option	t_{PORL}	 4064 16	 — —	t_{cyc}
Watchdog RESET Otuput Pulse Width	t_{DOGL}	1.5	—	t_{cyc}
Watchdog Time-Out	t_{DOG}	6144	7168	t_{cyc}
Timer Resolution** Input Capture Pulse Width (see Figure 20) Input Capture Pulse Period (see Figure 20)	 t_{RESL} t_{TH}, t_{TL} t_{TL}, t_{TL}	 4.0 250 ***	 — — —	 t_{cyc} ns t_{cyc}
Interrupt Pulse Width (Edge-Triggered)	t_{ILIH}	250	—	ns
Interrupt Pulse Period	t_{ILIL}	*	—	t_{cyc}
OSC1 Pulse Width	t_{OH}, t_{OL}	200	—	ns

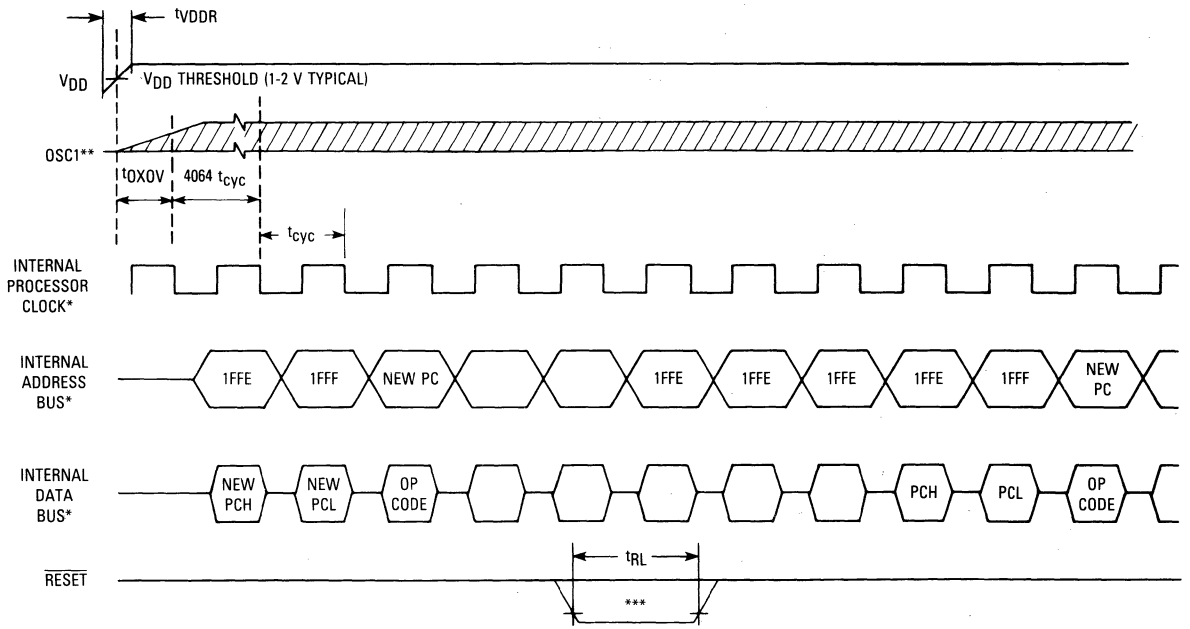
NOTES:

*The minimum period t_{ILIL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus 21 t_{cyc}.

**Since a 2-bit prescaler in the timer must count four internal cycles (t_{cyc}), this is the limiting minimum factor in determining the timer resolution.

***The minimum period t_{TLTL} should not be less than the number of cycle times it takes to execute the capture interrupt service routine plus 24 t_{cyc}.

3



*Internal timing signal and bus information not available externally.
**OSC1 line is not meant to represent frequency. It is only used to represent time.
***The next rising edge of the internal processor clock following the rising edge of $\overline{\text{RESET}}$ initiates the reset sequence.

Figure 21. Power-On Reset and $\overline{\text{RESET}}$

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola in the following media:

MS™-DOS/PC-DOS disk file (360K)
EPROM(s) 2764, MCM68764, MCM68766, or EEPROM MC68HC805C4

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, a sales person, or a Motorola representative.

FLEXIBLE DISKS

A flexible disk (MS-DOS/PC-DOS disk file), programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. The diskette should be clearly labeled with the customer's name, data, project or product name, and the name of the file containing the pattern.

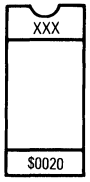
MS-DOS is Microsoft's Disk Operating System. PC-DOS is the IBM® Personal Computer (PC) Disk Operating System. Disk media submitted must be a standard density (360K) double-sided 5 1/4 inch compatible floppy diskette. The diskette must contain object file code in Motorola's S-record format. The S-record format is a character-based object file format generated by M6805 cross assemblers and linkers on IBM PC style machines.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to expedite the process in case of any difficulty with the pattern file.

EPROMs

A 2764, 68764, or 68766 type EPROM, programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. Since all program and data space information will fit on one 2764, 68764, or 68766 EPROM device, the EPROM must be programmed as described in the following paragraphs.

For an MC68HC805B6 MCU start the page zero, user ROM at EEPROM address $0020 through $004F. Start the user ROM at EEPROM address $0800 through $1EFF with vectors from $1FF0 to $1FFF. All unused bytes, including the user's space, must be set to zero. For shipment to Motorola, EPROMs should be placed in a conductive IC carrier and packed securely. Styrofoam is not acceptable for shipment.



xxx = Customer ID

Verification Media

All original pattern media (EPROMs or floppy disks) are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked, and the verification form should be completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for the creation of the customer mask. To aid in the verification process, Motorola will program *customer supplied* blank EPROM(s) or DOS disks from the data file used to create the custom mask.

ROM VERIFICATION UNITS (RVUs)

Ten MCUs containing the customer's ROM pattern will be sent for program verification. These units will have been made using the custom mask, but are for the purpose of ROM verification only. For expediency, the MCUs are unmarked, packaged in ceramic, and tested with five volts at room temperature. These RVUs are free with the minimum order quantity, but are not production parts. RVUs are not backed or guaranteed by Motorola Quality Assurance.

ORDERING INFORMATION

The following table provides ordering information pertaining to the package type, temperature, and MC order numbers for the MC68HC05B4 device.

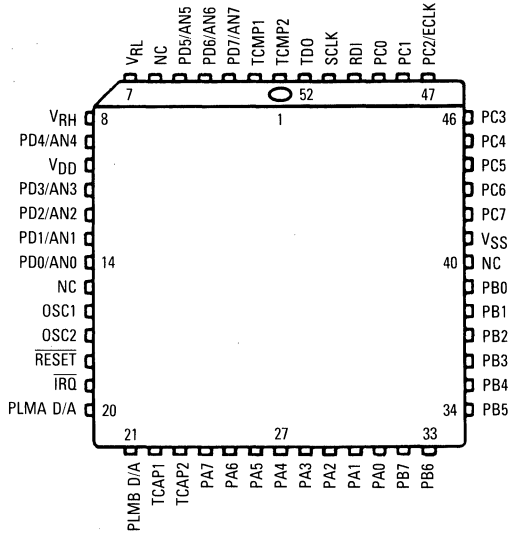
Package Type	Temperature	MC Order Number
Plastic (P Suffix)	0°C to +70°C	MC68HC05B4P
	−40°C to +85°C	MC68HC05B4CP
	−40°C to +125°C	MC68HC05B4MP
PLCC (FN Suffix)	0°C to +70°C	MC68HC05B4FN
	−40°C to +85°C	MC68HC05B4CFN
	−40°C to +125°C	MC68HC05B4MFN

3

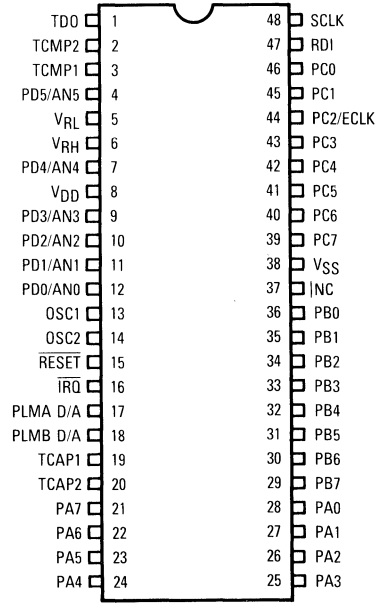
MS is a trademark of Microsoft, Inc.
IBM is a registered trademark of International Business Machines Corporation.

PIN ASSIGNMENTS

52-Pin PLCC



48-Pin Dual-in-Line Package



3

MC68HC05B6

Technical Summary
8-Bit Microcontroller Unit

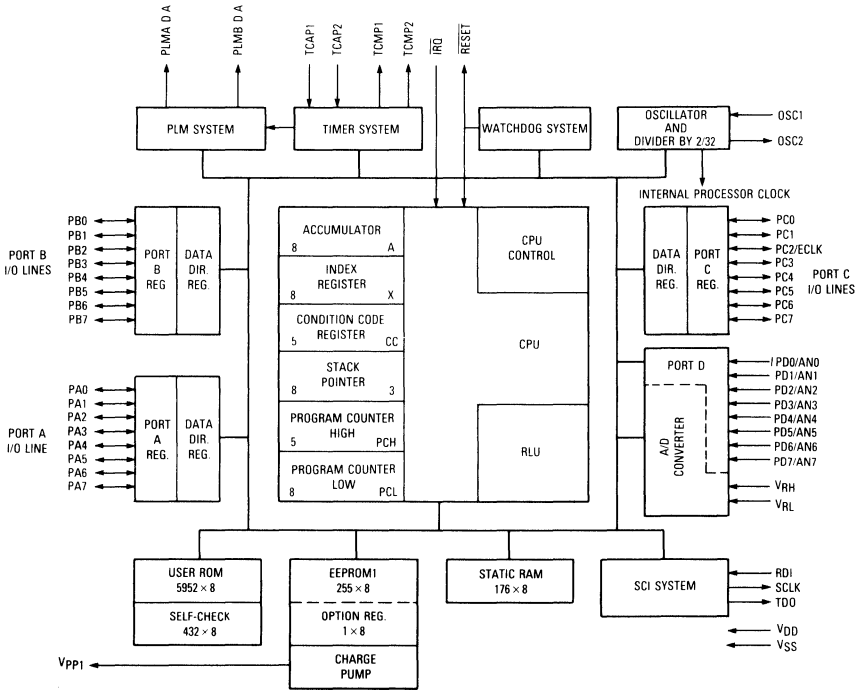
The MC68HC05B6 (HCMOS) microcontroller unit (MCU) is a member of the M68HC05 Family of microcontrollers. This high-performance, low-power MCU has parallel I/O capability with pins programmable as input or output. This publication contains condensed information on the MCU; for more detailed information, contact your local Motorola sales office.

The following block diagram depicts the hardware features; additional features available on the MCU are shown below and at the top of page 2.

- On-Chip Oscillator with Crystal/Ceramic Resonator
- Memory-Mapped I/O
- 176 Bytes of On-Chip RAM
- 256 Bytes of On-Chip EEPROM

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

FEATURES (Continued)

- 5952 Bytes of User ROM
- 24 Bidirectional I/O Lines and 8 Input-Only Lines
- Serial Communications Interface (SCI) System
- 8-Channel A/D Converter
- Watchdog System
- Self-Check Mode
- Power-Saving STOP and WAIT Modes
- Single 3.0- to 6.0-Volt Supply
- Fully Static Operation
- Two Pulse-Length Modulation Systems (D/A)
- 16-Bit Timer with Two Input Capture and Two Output Functions
- Slow Mode Option Divides the Basic Clock Frequency by 16

SIGNAL DESCRIPTION

The signal descriptions of the MCU are discussed in the following paragraphs.

V_{DD} AND V_{SS}

Power is supplied to the microcontroller using these two pins. V_{DD} is the positive supply, and V_{SS} is ground.

$\overline{IRQ}$

This pin is a programmable option that provides four different choices of interrupt triggering sensitivity. Refer to **INTERRUPTS** for more detail. Note that the voltage level on this pin affects the mode of operation.

OSC1, OSC2

These pins provide control input for an on-chip clock oscillator circuit. A crystal, a ceramic resonator, or an external signal connects to these pins providing a system clock. The oscillator frequency is two times the internal bus rate (or 32 times as a software option).

Crystal

The circuit shown in Figure 1(b) is recommended when using a crystal. Using an external CMOS oscillator is recommended when crystals outside the specified ranges are to be used. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and start-up stabilization time. Refer to **ELECTRICAL SPECIFICATIONS** for V_{DD} specifications.

Ceramic Resonator

A ceramic resonator may be used in place of the crystal in cost-sensitive applications. The circuit in Figure 1(b) is recommended when using a ceramic resonator. Figure 1(a) lists the recommended capacitance and resistance values. The manufacturer of the resonator considered should be consulted for specific information on resonator operation.

External Clock

An external clock should be applied to the OSC1 input with the OSC2 input not connected, as shown in Figure 1(d).

INPUT CAPTURE (TCAP1)

This pin controls the input capture 1 feature for the on-chip programmable timer (see Table 2).

INPUT CAPTURE (TCAP2)

This pin controls the input capture 2 feature for the on-chip programmable timer.

OUTPUT COMPARE (TCMP1)

This pin provides an output for the output compare 1 feature of the on-chip timer.

OUTPUT COMPARE (TCMP2)

This pin provides an output for the output compare 2 feature on the on-chip timer.

$\overline{RESET}$

This pin is used to reset the MCU and provide an orderly start-up procedure by pulling $\overline{RESET}$ low. The voltage on this pin affects the mode of operation (see Table 2, **Mode of Operation Selection**).

INPUT/OUTPUT PORTS (PA7–PA0, PB7–PB0, PC7–PC0)

These 24 lines are arranged into three 8-bit ports (A, B, and C). These ports are programmable as either inputs or outputs under software control of the data direction registers. Refer to **PROGRAMMING** for additional information.

FIXED INPUT PORT (PD7/AN7–PD0/AN0)

These eight lines comprise port D, a fixed input port. Enabling the A/D function affects this port. Port D accepts the eight analog inputs when the A/D is enabled. Port D can be used for digital input during a conversion sequence, but this may inject noise on the analog signals, reducing the conversion accuracy. Also, a digital read of

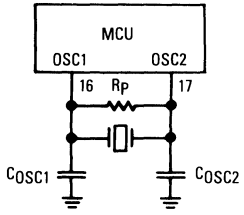
Crystal

	2 MHz	4 MHz	Units
R_{SMAX}	400	75	Ω
C_0	5	7	pF
C_1	0.008	0.012	μF
C_{OSC1}	15-40	15-30	pF
C_{OSC2}	15-30	15-25	pF
R_P	10	10	MΩ
Q	30	40	K

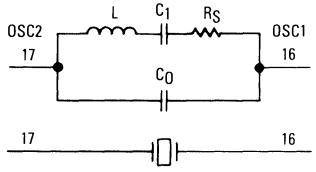
Ceramic Resonator

	2-4 MHz	Units
R_S (typical)	10	Ω
C_0	40	pF
C_1	4, 3	μF
C_{OSC1}	30	pF
C_{OSC2}	30	pF
R_P	1-10	MΩ
Q	1250	—

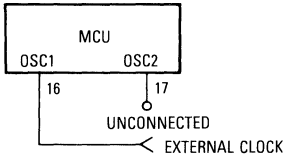
(a) Crystal/Ceramic Resonator Parameters



(b) Crystal/Ceramic Resonator Oscillator Connections



(c) Equivalent Crystal Circuit



(d) External Clock Source Connections

Figure 1. Oscillator Connections

port D with levels other than V_{DD} or V_{SS} on the pins results in greater power dissipation during the read cycle. Refer to **PROGRAMMING** for additional information.

NOTE

In the 48-pin dual-in-line package, the fixed input port (D) of the MC68HC05B6 is reduced to six pins (PD5–PD0, AN5–AN0). This change has no effect on either programming or operation of port D or the A/D converter.

PLMA

This pin is the output of the pulse-length modulation converter A. See **PULSE-LENGTH D/A CONVERTERS** for further information.

PLMB

This pin is the output of the pulse-length modulation converter B. See **PULSE-LENGTH D/A CONVERTERS** for further information.

RDI (Receive Data In)

This pin is the input of the SCI. See **Serial Communications Interface** for more information.

TDO (Transmit Data Out)

This pin is the output of the SCI. See **Serial Communications Interface** for more information.

SCLK

This pin is the clock output pin of the SCI transmitter. See **Serial Communications Interface** for more information.

V_{PP1}

This pin is the EEPROM programming voltage output. See **EEPROM** for further information.

V_{RH}

This pin is the positive reference voltage for the A/D converter.

V_{RL}

This pin is the negative reference voltage for the A/D converter.

INPUT/OUTPUT PROGRAMMING

Input/output port programming, fixed input port programming, and serial port programming are discussed in the following paragraphs.

INPUT/OUTPUT PORT PROGRAMMING

Any port pin is programmable as either an input or an output under software control of the corresponding data direction register (DDR). Each port bit can be selected as output or input by writing the corresponding bit in the port DDR to a logic one for output and logic zero for input. On reset, all DDRs are initialized to logic zero to put the ports in the input mode. The port output registers are not initialized on reset but may be written to before setting the DDR bits to avoid undefined levels.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This port write may be used to initialize the data registers and avoid undefined outputs. Refer to Figure 2 for typical port circuitry and to Table 1 for a list of the I/O pin functions.

Table 1. I/O Pin Functions

R/$\overline{W}$*	DDR	I/O Pin Functions
0	0	The I/O pin is in input mode. Data is written into the output data latch.
0	1	Data is written into the output data latch and output to the I/O pin.
1	0	The state of the I/O pin is read.
1	1	The I/O pin is in an output mode. The output data latch is read.

*R/$\overline{W}$ is an internal signal.

Under software control, the PC2 pin can become the CPU clock output. If this option is selected, the corresponding DDR bit is automatically set, and bit 2 of port C always reads the output data latch. The other port C pins are not affected by this feature.

Control Register (CTL/ECLK) $07

7	6	5	4	3	2	1	0
0	0	0	0	ECLK	—	—	—
RESET:							
0	0	0	0	0	0	0	0

ECLK — ECLK Control
1 = I/O port function of PC2 is forced to output mode, and PC2 outputs the ECLK CPU clock.
0 = PC2 functions as a regular I/O pin.

FIXED INPUT PORT PROGRAMMING

Port D is a fixed input port that monitors the external pins whenever the A/D is disabled. After reset, all eight bits become digital inputs because all special function drivers are disabled. Port D is always at digital input, whether the A/D is on or off.

NOTE

Any unused inputs and I/O ports should be tied to an appropriate logic level (e.g., either V_{DD} or V_{SS}).

SERIAL PORT (SCI) PROGRAMMING

The SCI uses two or three pins for its functions: RDI for its receive data input, TDO for its transmit data output, and SCLK to output the transmitter clock, if needed.

MEMORY

The MCU is capable of addressing 8192 bytes of memory and I/O registers, as shown in Figure 3. The locations consist of user ROM, user RAM, EEPROM, self-check ROM, control registers, and I/O. The user-defined reset and interrupt vectors are located from $1FF0 to $1FFF.

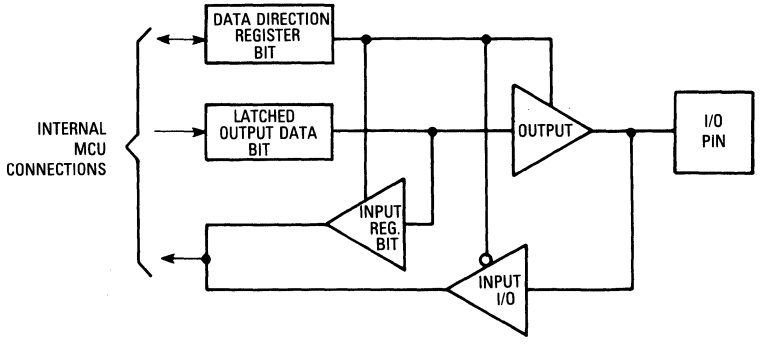


Figure 2. Typical Port I/O Circuit

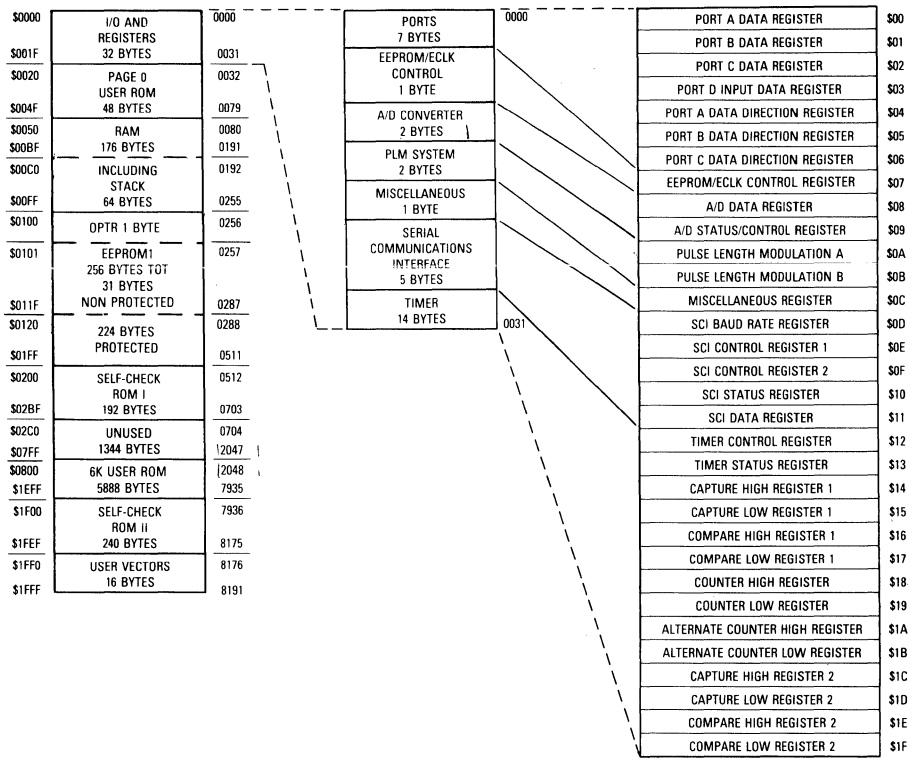


Figure 3. Memory Map

The shared stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer decrements during pushes and increments during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

EEPROM

The MCU has 256 bytes of byte-erasable EEPROM (255 bytes general purpose and 1-byte option register), located at addresses $0100–$01FF. An internal charge pump, connected to the V_{PP} pin, avoids the necessity of supplying a high voltage for erase and programming. The V_{PP} pin should be left open.

CAUTION

An external high voltage should **not** be applied to this pin.

To provide a higher degree of security for stored data, there is no bulk or row erase.

EEPROM Read Operation

To read data from EEPROM, the E1LAT bit must be zero. When E1LAT is zero, the E1PGM and E1ERA bits are forced to zero, and the 256-byte EEPROM is read as if it were a normal ROM. The V_{PP} charge pump generator is off since E1PGM is zero. If a read is performed while E1LAT is set, data will be read as $FF.

NOTE

When not performing a programming or erase operation on the EEPROM, remain in read mode (E1LAT = 0).

EEPROM Erase Operation

To erase a byte of EEPROM, set E1LAT and E1ERA to one, write to the address to be erased, and set E1PGM for a time t_{ERA1}. After the required erase time, E1LAT must be cleared, which resets E1ERA and E1PGM. To erase a second word, E1LAT must be cleared before it is set, or the erase will have not effect. This procedure is

done to increase the security of the stored data. While an erase is being performed, any access to the EEPROM will not be successful. Data written in an erase operation is not used; therefore, its value is not significant. User programs must be running from ROM or RAM since the EEPROM has its address and data buses latched.

EEPROM Programming Operation

To program a byte of EEPROM, set the E1LAT bit, write data to the desired address, and set the E1PGM bit for a time t_{PROG}. After the required programming delay, E1LAT must be cleared, which also resets E1PGM. While a programming operation is being performed, any access to the EEPROM will not be successful.

NOTE

To program a byte correctly, the byte must have been previously erased.

To program a second word, E1LAT must be cleared before it is set, or the programming will have no effect. This procedure is done to increase the security of the stored data. User programs must be running from RAM or ROM since the EEPROM will have its data buses latched.

Control Register (CTL/ECLK) $07

7	6	5	4	3	2	1	0
—	—	—	—	—	E1ERA	E1LAT	E1PGM
RESET:							
0	0	0	0	0	U	0	U

E1ERA — EEPROM Erase
- 1 = An erase will take place if E1LAT and E1PGM are both one.
- 0 = A programming operation will take place if E1LAT and E1PGM are both one.

If E1LAT = 0, E1ERA is held to zero. Once an EEPROM address is selected, E1ERA cannot be changed.

E1LAT — EEPROM Latch Enable
- 1 = Address and data can be latched into the EEPROM for programming or erase operation if E1PGM = 0.
- 0 = Data can be read from the EEPROM, and the E1ERA and E1PGM bits are cleared.

After the programming or erase time, the E1LAT bit must be reset in order to reset the E1ERA and E1PGM bits.

E1PGM — EEPROM Program Mode
- 1 = Charge pump generator is on, and the resulting high voltage is applied to the EEPROM array.
- 0 = Charge pump generator is off.

E1PGM cannot be set before the data is selected; it can only be reset by resetting E1LAT.

The charge pump is not affected by the WAIT mode; thus, WAIT can be used for the erase or programming delay time. If STOP mode is entered, the EEPROM is set to read mode.

The V_{PP1} charge pump generator is normally supplied by the CPU clock, but for very low clocking frequencies, the A/D RC oscillator should be used. See **A/D CONVERTER** for more information.

Options Register (OPTR) $0100

7	6	5	4	3	2	1	0
—	—	—	—	—	—	EE1P	SEC
RESET:							
U	U	U	U	U	U	U	U

EE1P — EEPROM Protect
- 1 = EEPROM not protected.
- 0 = EEPROM addresses from $0120 to $01FF are read only, and attempts to write to this area will be unsuccessful.

When this bit is erased to one, protection remains until the next external or power-up reset occurs.

SEC — High-Security Bit
- 1 = Security not active.
- 0 = EEPROM contents protected because access to test mode is inhibited.

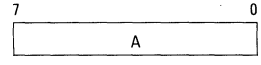
The SEC bit can only be erased to one externally by entering self-check mode, which erases the entire EEPROM. When SEC is changed, the new value has no effect until the next external or power-on reset.

REGISTERS

The MCU contains the registers described in the following paragraphs.

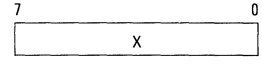
ACCUMULATOR (A)

The accumulator is a general-purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



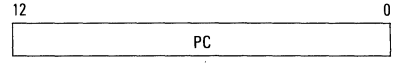
INDEX REGISTER (X)

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16-bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



PROGRAM COUNTER (PC)

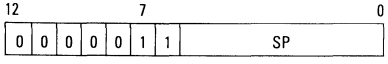
The program counter is a 13-bit register that contains the address of the next instruction to be executed.



STACK POINTER (SP)

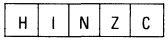
The stack pointer is a 13-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set to location $00FF. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

When accessing memory, the seven most significant bits are permanently set to 0000011. These seven bits are appended to the six least significant register bits to produce an address within the range of $00FF to $00C0. Subroutines and interrupts may use up to 64 (decimal) locations. If 64 locations are exceeded, the stack pointer wraps around and loses the previously stored information. A subroutine call occupies two locations on the stack; an interrupt uses five locations.



CONDITION CODE REGISTER (CCR)

The CCR is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program, and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.



Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, the timer and external interrupt is masked (disabled). If an interrupt occurs while this bit is set, the interrupt is latched and processed as soon as the interrupt bit is cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic one).

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/Borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions and during shifts and rotates.

SELF-CHECK

The self-check capability provides the ability to determine if the device is functional. Table 2 shows how self-check mode is entered. Self-check is performed using the circuit shown in Figure 4. Port C pins PC3–PC0 are monitored for the self-check results. After reset, the following tests are performed automatically:

- I/O — Exercise of ports A, B, C, and D
- RAM — Counter test for each RAM byte
- ROM — Exclusive OR with odd ones parity result
- Timer — Tracks counter register and checks ICF1, ICF2, OCF1, OCF2, and TOV flag
- Interrupts — Tests external, timer, and SCI interrupts
- SCI — Transmission test; checks RDRF, TDRE, TC, and FE flags
- A/D — Checks A/D on internal channels: V_{RL}, V_{RH}, and $(V_{RL}+V_{RH})/2$
- EEPROM — Optional. Performs write/erase of the 256-byte EEPROM and then deactivates the security bit.
- PLM — Checks basic PLM function
- Watchdog System — Checks watchdog function

Self-check results (using the LEDs as monitors) are shown in Table 3. The following subroutines are available to the user and do not require any external hardware.

Table 2. Mode of Operation Selection

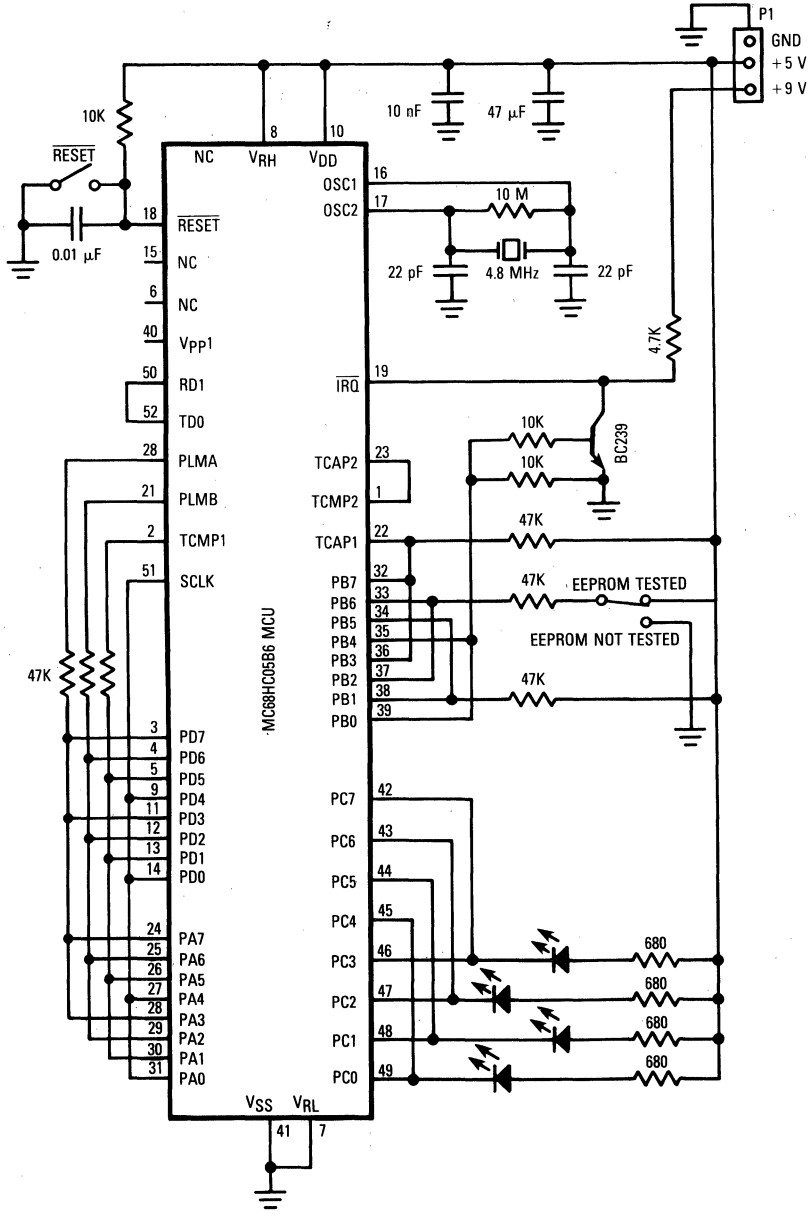
$\overline{\text{RESET}}$ Pin	$\overline{\text{IRQ}}$ Pin	TCAP1 Pin	Mode
⎍	V_{SS} to V_{DD}	V_{SS} to V_{DD}	Normal
⎍	+9 Volts	V_{DD}	Self-Check
V_{SS}	V_{SS} to V_{DD}	V_{SS} to V_{DD}	Reset Condition

Table 3. Self-Check Results

PC3	PC2	PC1	PC0	Remarks
1	0	0	1	Bad Port
0	1	1	0	Bad Port
1	0	1	0	Bad RAM
1	0	1	1	Bad ROM
1	1	0	0	Bad Timer
1	1	0	1	Bad SCI
1	1	1	0	Bad A/D
0	0	0	0	Bad EEPROM
0	0	0	1	Bad PLM
0	0	1	0	Bad Interrupts
0	0	1	1	Bad Watchdog
Flashing				Good Device
All Others				Bad Device, Bad Port, etc.

0 indicates LED is on; 1 indicates LED is off.

3



NOTE: Pin numbers are valid for 52-pin PLCC package only.

Figure 4. Self-Check Circuit Schematic Diagram

RAM CHECK SUBROUTINE

This subroutine returns with the Z bit cleared if any error is detected; otherwise, the Z bit is set. The stack pointer must be set to $FF. The stack pointer must be set to $FF. The RAM check subroutine is called at location $021E. A counter test is done on each location from address $50 to $FD. Each location is made to count from $00 to $00 again. Locations $FE and $FF are assumed to contain the return address. Upon return to the user's program, if the test passed, X = $00, A = $00, and RAM locations $0050 and $00FD contain $01.

NOTE

The watchdog system is turned on when calling this subroutine.

A/D CONVERTER CHECK SUBROUTINE

This subroutine returns with the Z bit cleared if any error is detected; otherwise, the Z bit is set. The subroutine is called at location $1FAA with X = $00 and A/D STAT/CTRL (address $09) = $20 (ADON = 1 for more than 100 μs and channel PD0 selected). Conversion is done on three of the internal channels: V_{RH}, V_{RL}, and $(V_{RL} + V_{RH})/2$. The result of these conversions is verified at ±1 LSB. Upon return to the user's program, if the test passed, X = $09, A = $00 or $01.

ROM CHECKSUM SUBROUTINE

This subroutine returns with the Z bit cleared if any error is detected; otherwise, the Z bit is set. The ROM checksum subroutine is called at location $0232 with RAM location $0053 equal to $01 and A = 0. A short routine is set up and executed in RAM to compute a checksum of the entire ROM pattern. RAM locations $0050 through $0053 are overwritten. Upon return to the user's program, if the test passed, X = 0, A = 0.

NOTE

The A/D and the watchdog system are turned on when calling this subroutine.

RESETS

The MCU can be reset two ways: by initial power-up (POR) and by the external reset input ($\overline{\text{RESET}}$). The $\overline{\text{RESET}}$ input consists mainly of a Schmitt trigger that senses the $\overline{\text{RESET}}$ line logic level.

POWER-ON RESET

An internal reset is generated on power-up to allow the internal clock generator to stabilize. The power-on reset is strictly for power turn-on conditions and should not be used to detect a drop in the power supply voltage. There is a delay (t_{PORL}) after the oscillator becomes active. If the $\overline{\text{RESET}}$ pin is low at the end of t_{PORL}, the MCU will remain in the reset condition until $\overline{\text{RESET}}$ goes high. A mask option allows t_{PORL} to be either 16 or 4064 internal processor clock cycles (t_{CYC}).

EXTERNAL RESET INPUT

The MCU is reset when a logic zero is applied to the $\overline{\text{RESET}}$ input for a period of one and one-half machine cycles (t_{CYC}).

Miscellaneous Register (0C)

7	6	5	4	3	2	1	0
POR	INTP	INTN	INTE	SFA	SFB	SM	WDOG
RESET: U	0	0	1	0	0	0	0

POR — Power-On Reset
- 1 = The reset occurring is a power-on, not external, reset
- 0 = Power-on reset not in progress

INTP — External Interrupt Positive
Allows a choice of $\overline{\text{IRQ}}$ sensitivity, with INTN. See Table 4.

INTN — External Interrupt Negative
Allows a choice of $\overline{\text{IRQ}}$ sensitivity, with INTP. See Table 4.

INTE — External Interrupt Enable
Allows the user to enable or disable the external interrupt function.

SFA — Slow/Fast Selection for PLMA
- 1 = Slow speed used for PLMA (4096 times the timer clock period)
- 0 = Fast speed used for PLMA (256 times the timer clock period). See **PULSE-LENGTH D/A CONVERTERS**

SFB — Slow/Fast Selection for PLMB
- 1 = Slow speed used for PLMB (4096 times the timer clock period)
- 0 = Fast speed used for PLMB (256 times the timer clock period). See **PULSE-LENGTH D/A CONVERTERS**

SM — Slow Mode
- 1 = System runs at 1/16th the normal clock rate ($f_{OSC}/32$)
- 0 = System runs at normal clock rate ($f_{OSC}/2$)

WDOG — Watchdog Counter System
- 1 = Watchdog counter system enabled
- 0 = Watchdog counter system disabled

NOTE

The reset generated by the watchdog timer is a system reset; thus, the watchdog is disabled after a watchdog reset.

Table 4. External Interrupt Options

INTP	INTN	External Interrupt Options
0	0	Negative Edge and Low-Level Sensitive
0	1	Negative Edge Only
1	0	Positive Edge Only
1	1	Positive and Negative Edge Sensitive

Slow Mode

The slow mode function is controlled by the SM bit in the miscellaneous register (0C). In slow mode (SM = 1), an extra divide-by-sixteen circuit is added between the oscillator and the internal clock driver. This slows all functions by a factor of 16 (including SCI, A/D, and timer), which is particularly useful in WAIT mode. SM is cleared by external or power-on reset and by STOP mode.

NOTE

If slow mode is enabled while using the A/D, the internal A/D RC oscillator should be turned on.

Watchdog System

The watchdog counter is driven by the 1024 prescaler in the timer and, unless the counter is reset, generates a system reset when it reaches its maximum count (1024 × 8).

A mask option is available that provides two methods of enabling the watchdog timer. In the first option, the watchdog system is controlled by the WDOG bit in the miscellaneous register (0C). Writing a one to the bit starts the watchdog or, if it is already started, resets the counter to zero. Writing a zero has no effect; the WDOG bit can only be cleared by external or power-on reset. In the second option, the watchdog timer is always enabled following reset.

A second mask option determines the watchdog timer function during WAIT. The watchdog timer can remain active during WAIT, and can cause a reset if the device remains in WAIT longer than the watchdog timeout period. Alternatively, the watchdog timer suspends operation during WAIT and resets its count, resuming normal operation following reset.

INTERRUPTS

The MCU can be interrupted four different ways: the three maskable hardware interrupts ($\overline{IRQ}$, SCI, and timer) and the nonmaskable software interrupt instruction (SWI).

Interrupts cause the processor to save register contents on the stack and to set the interrupt mask (I bit) to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack and normal processing to resume. The stacking order is shown in Figure 5.

Unlike $\overline{RESET}$, hardware interrupts do not cause the current instruction execution to be halted but are considered pending until the current instruction is complete.

NOTE

The current instruction is the one already fetched and being operated on.

When the current instruction is complete, the processor checks all pending hardware interrupts. If unmasked (I bit clear) and if the corresponding interrupt enable bit is set, the processor proceeds with interrupt processing; otherwise, the next instruction is fetched and executed.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction, regardless of the I-bit state. Refer to Figure 6 for the reset and interrupt instruction processing sequence.

TIMER INTERRUPT

There are three different timer interrupt flags that cause a timer interrupt whenever they are set and enabled. The interrupt flags are in the timer status register (TSR), and the enable bits are in the timer control register (TCR). Refer to **TIMER** for more information.

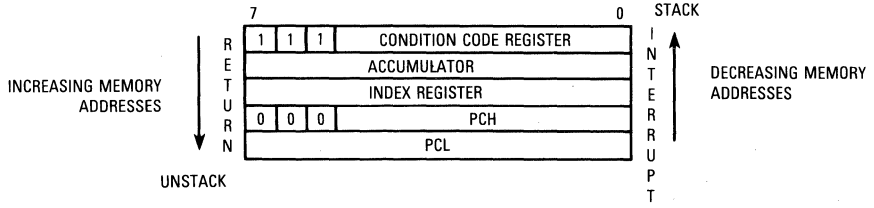
EXTERNAL INTERRUPT

If the interrupt mask bit (I bit) of the CCR is set, all interrupts are disabled. Clearing the I bit enables the external interrupt. The external interrupt is internally synchronized and then latched on the falling edge of $\overline{IRQ}$. The action of the external interrupt is identical to the timer interrupt with the exception that the interrupt request input at $\overline{IRQ}$ is latched internally and the service routine address is specified by the contents of $1FFA and $1FFB.

Four options are available for interrupt triggering sensitivity:

- Negative edge and low level
- Negative edge only
- Positive edge only
- Positive and negative edge

See **Miscellaneous Register (0C)** for further information.



NOTE: Since the Stack Pointer decrements during pushes, the PCL is stacked first, followed by PCH, etc. Pulling from the stack is in the reverse order.

Figure 5. Interrupt Stacking Order

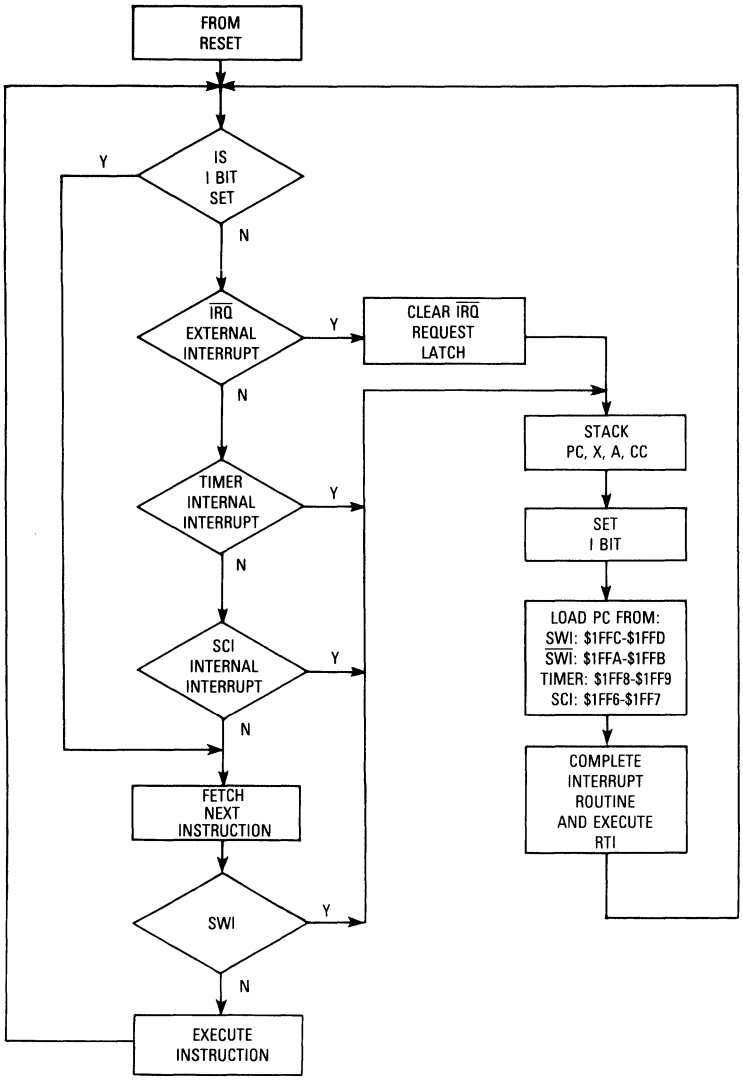


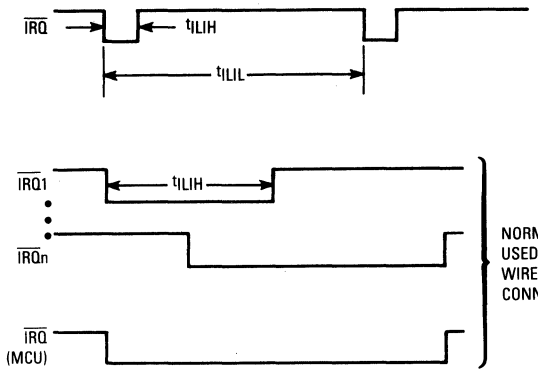
Figure 6. Reset and Interrupt Processing Flowchart

Figure 7 shows a mode timing diagram for the interrupt line. The timing diagram shows two treatments of the interrupt line to the processor. The first method shows a single pulse on the interrupt line spaced far enough apart to be serviced. The minimum time between pulses is a function of the length of the interrupt service. Once a pulse occurs, the next pulse should not occur until an RTI occurs. This time (t_{ILIL}) is obtained by adding 21 instruction cycles to the total number of cycles it takes to complete the service routine (not including the RTI instruction).

The second method shows many interrupt lines "wire-ORed" to form the interrupts at the processor. If the interrupt line remains low after servicing an interrupt, then the next interrupt is recognized.

NOTE

The internal interrupt latch is cleared in the first part of the interrupt service routine; therefore, one external interrupt pulse could be latched and serviced as soon as the I bit is cleared.



Edge-Sensitive Trigger Condition
The minimum pulse width (t_{ILIH}) is either 125 ns (V_{DD} = 5 V) or 250 ns (V_{DD} = 3 V). The period t_{ILIL} should not be less than the number of t_{cyc} cycles it takes to execute the interrupt service routine plus 21 t_{cyc} cycles.

Level-Sensitive Trigger Condition
If after servicing an interrupt the $\overline{IRQ}$ remains low, then the next interrupt is recognized.

Figure 7. External Interrupt Mode Diagram

3

SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. If the I bit is zero, SWI executes after the other interrupts. The SWI operation is similar to the hardware interrupts. The interrupt service routine address is specified by the contents of memory locations $1FFC and $1FFD.

SCI INTERRUPTS

An interrupt in the SCI occurs when one of the interrupt flag bits in the serial communications status register is set, provided the I bit in the CCR is clear and the enable bit in the serial communications control register 2 is set. Software in the serial interrupt service routine must determine the cause and priority of the SCI interrupt by examining the interrupt flags and status bits in the SCI status register.

LOW-POWER MODES

STOP

The STOP instruction places the MCU in its lowest power consumption mode. In the STOP mode, the internal oscillator is turned off, halting all internal processing including timer, SCI, and A/D operation (refer to Figure 8).

During the STOP mode, the TCR bits are altered to remove any pending timer interrupt request and to disable any further timer interrupts. The timer prescaler is cleared. The I bit in the CCR is cleared to enable external interrupts. All other registers and memory remain unaltered. All input/output lines remain unchanged. The processor can be brought out of the STOP mode only by an external interrupt or reset.

SCI during STOP Mode

When the MCU enters the STOP mode, the baud rate generator stops, halting all SCI activity. If the STOP instruction is executed during a transmitter transfer, that

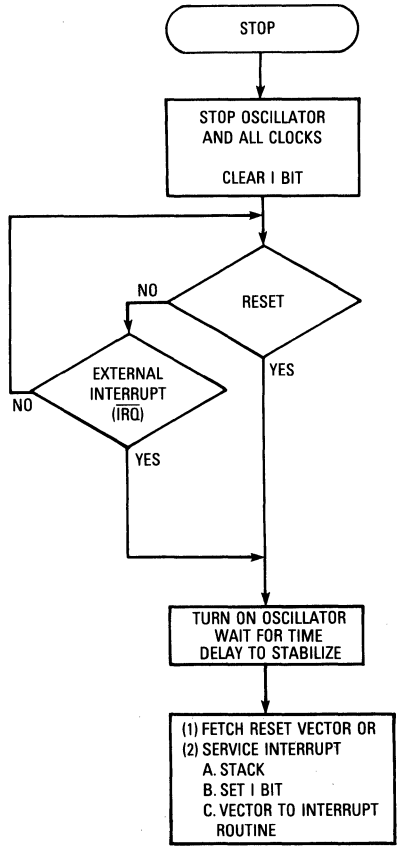


Figure 8. STOP Function Flowchart

transfer is halted. If a low input to the $\overline{IRQ}$ pin is used to exit STOP mode, the transfer resumes. If the SCI receiver is receiving data and the STOP mode is entered, received data sampling stops because the baud rate generator stops, and all subsequent data is lost. For these reasons, all SCI transfers should be in the idle state when the STOP instruction is executed.

Watchdog during STOP Mode

The STOP instruction is inhibited when the watchdog system is enabled. If a STOP instruction is executed while the watchdog is enabled, a reset occurs that resets the entire MCU.

EEPROM during STOP Mode

The EEPROM is set to read, and the V_{PP1} high-voltage charge pump generator is disabled when stop mode is entered.

PLM during STOP Mode

When the MCU enters stop mode, the PLM outputs remain at their particular level. If power-on or external reset causes the exit from stop mode, the register values are forced to $00.

A/D Converter during STOP Mode

When stop mode is entered with the A/D converter turned on, the A/D clocks are stopped and the A/D converter is disabled for the duration of stop mode, including the t_{PORL} startup time. If the A/D RC oscillator is used, it will also be disabled.

When leaving STOP mode, after the t_{PORL} startup time, the A/D converter and A/D RC oscillator resume regular operation. However, a time t_{ADON} is required for the current sources to stabilize. During t_{ADON}, A/D conversion results may be inaccurate.

WAIT

The WAIT instruction places the MCU in a low-power consumption mode, but the WAIT mode consumes more power than the STOP mode. All CPU action and the watchdog system are suspended, but the timer, SCI, PLM, and A/D remain active (refer to Figure 9). An interrupt from the timer, SCI, or an IRQ can cause the MCU to exit the WAIT mode.

During the WAIT mode, the I bit in the CCR is cleared to enable interrupts. All other registers, memory, and input/output lines remain in their previous state. The timer may be enabled to allow a periodic exit from the WAIT mode.

To achieve proper operation and reduce power consumption, the following points should be set as desired before entering wait mode:

- Timer interrupt enable bits
- A/D control bits
- EEPROM control bits
- SCI enable bits and interrupt enable bits

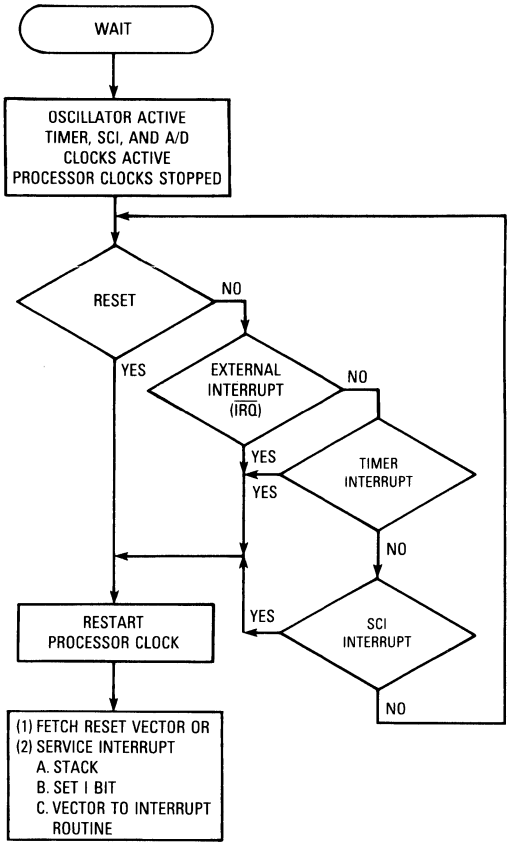


Figure 9. WAIT Function Flowchart

TIMER

The timer consists of a 16-bit, software-programmable counter driven by a fixed divide-by-four prescaler. This timer can be used for many purposes, including input waveform measurements of two input signals while simultaneously generating two output waveforms. Pulse widths can vary from several microseconds to many seconds. The programmable timer works in conjunction with the PLM system to execute two 8-bit D/A PLM conversions, with a choice of two repetition rates. Refer to Figure 10 for a timer block diagram.

Because the timer has a 16-bit architecture, each specific functional segment (capability) is represented by two registers. These registers contain the high and low byte of that functional segment. Generally, accessing the low byte of a specific timer function allows full control of that function; however, an access of the high byte inhibits that specific timer function until the low byte is also accessed.

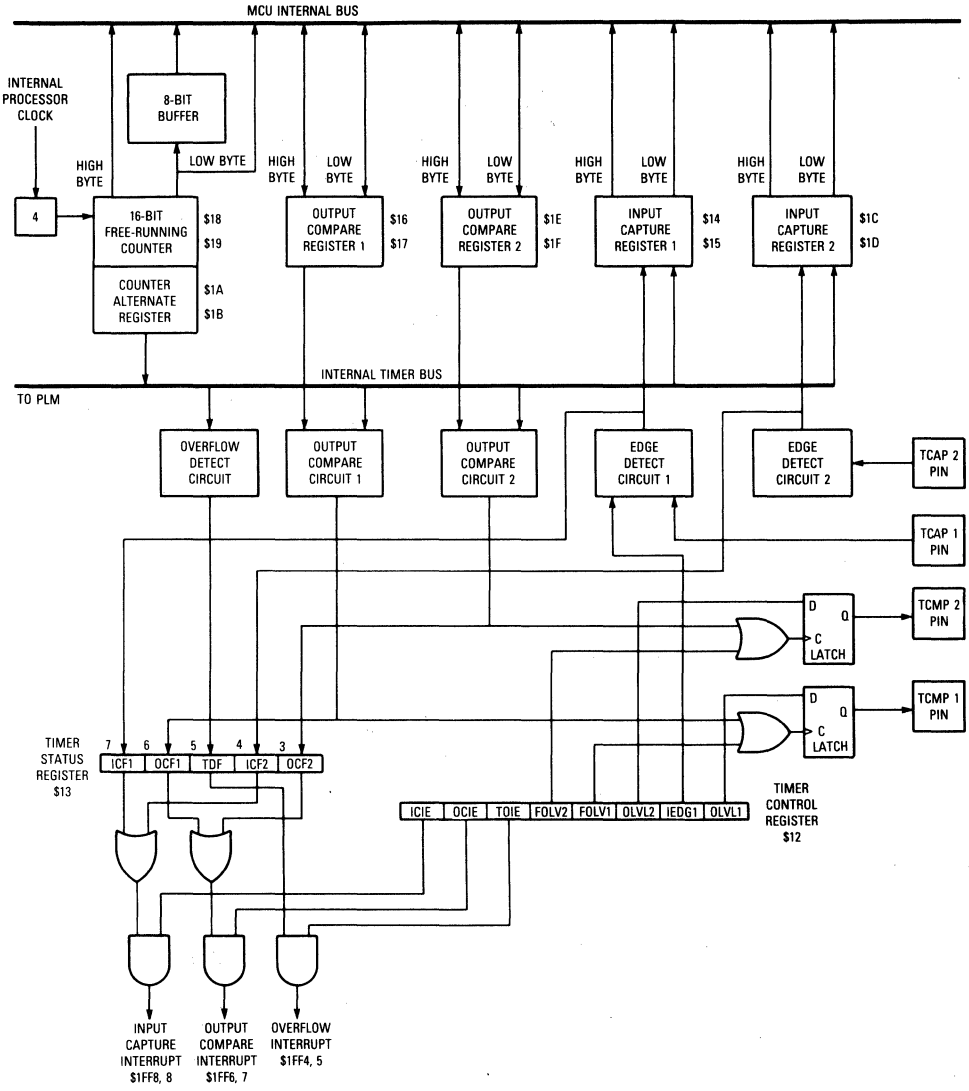


Figure 10. Timer Block Diagram

NOTE

The I bit in the CCR should be set while manipulating both the high and low byte register of a specific timer function to ensure that an interrupt does not occur.

COUNTER

The key element in the programmable timer is a 16-bit, free-running counter or counter register, preceded by a prescaler that divides the internal processor clock by four. The prescaler gives the timer a resolution of 2.0 microseconds if the internal bus clock is 2.0 MHz. The counter is incremented during the low portion of the internal bus clock. Software can read the counter at any time without affecting its value.

The double-byte, free-running counter can be read from either of two locations, $18–$19 (counter register) or $1A–$1B (counter alternate register). A read from only the least significant byte (LSB) of the free-running counter

($19, $1B) receives the count value at the time of the read. If a read of the free-running counter or counter alternate register first addresses the most significant byte (MSB) ($18, $1A), the LSB ($19, $1B) is transferred to a buffer. This buffer value remains fixed after the first MSB read, even if the user reads the MSB several times. This buffer is accessed when reading the free-running counter or counter alternate register LSB ($19 or $1B) and, thus, completes a read sequence of the total counter value. In reading either the free-running counter or counter alternate register, if the MSB is read, the LSB must also be read to complete the sequence.

The counter alternate register differs from the counter register in one respect: a read of the counter register LSB can clear the timer overflow flag (TOF). Therefore, the counter alternate register can be read at any time without the possibility of missing timer overflow interrupts due to clearing of the TOF.

The free-running counter is configured to $FFFC during reset and is always a read-only register. During a power-on reset, the counter is also preset to $FFFC and begins running after the oscillator start-up delay. Because the free-running counter is 16 bits preceded by a fixed divide-by-four prescaler, the value in the free-running counter repeats every 262,144 internal bus clock cycles. When the counter rolls over from $FFFF to $0000, the TOF bit is set. An interrupt can also be enabled when counter rollover occurs by setting its interrupt enable bit (TOIE).

NOTE

Since the PLM system uses the timer counter, PLM results will be affected when resetting this counter.

OUTPUT COMPARE REGISTERS

There are two output compare registers: output compare register 1 (OCR1) and output compare register 2 (OCR2). The output compare registers can be used for several purposes, such as controlling an output waveform or indicating when a period of time has elapsed. All bits are readable and writable and are not altered by the timer hardware or reset. If the compare function is not needed, the four bytes of the output compare registers can be used as storage locations.

NOTE

The same output compare interrupt enable bit is used for the two output compares.

Output Compare Register 1

The output compare register 1 (OCR1) is a 16-bit register, which is made up of two 8-bit registers at locations $16 (most significant byte) and $17 (least significant byte).

The output compare register contents are continually compared with the contents of the free-running counter and, if a match is found, the corresponding output compare flag (OCF1, bit 6 of timer status register $13) is set, and the corresponding output level (OLVL1) bit is clocked to pin TCMP1. The output compare register values and the output level bit should be changed after each successful comparison to control an output waveform or establish a new elapsed timeout. An interrupt can also accompany a successful output compare, provided the corresponding interrupt enable bit, OCIE, is set.

After a processor write cycle to the output compare register 1 containing the most significant byte ($16), the output compare 1 function is inhibited until the least significant byte ($17) is also written. The user must write both bytes (locations) if the most significant byte is written first. A write made only to the least significant byte ($17) will not inhibit the compare function. The free-running counter is updated every four internal bus clock cycles. The minimum time required to update the output compare register 1 is a function of the program rather than the internal hardware.

The processor can write to either byte of the output compare register 1 without affecting the other byte. The output level (OLVL1) bit is clocked to the corresponding output level register and then to the TCMP1 pin, regardless of whether the output compare flag (OCF1) is set or clear.

Output Compare Register 2

The output compare register 2 (OCR2) is a 16-bit register, which is made up of two 8-bit registers at locations $1E (most significant byte) and $1F (least significant byte). The function of OCR2 is identical to OCR1, requiring only changes of the register locations and control bits in the timer status register ($13) to make the OCR1 description apply to OCR2.

SOFTWARE FORCE COMPARE

The MCU provides a force compare capability to facilitate fixed frequency generation as well as other applications. Bit 3 (FOLV1 for OCR1) and bit 4 (FOLV2 for OCR2) in the timer control register ($12) implement this force compare. Writing a one to these bits causes the OLVL1 or OLVL2 values to be copied to the respective output registers (TCMP1 or TCMP2 pins). Internal logic allows a single instruction to change OLVL1 and OLVL2 and cause a forced compare with the new values of OLVL1 and OLVL2.

NOTE

A software force compare, which affects the corresponding output pin TCMP1 or TCMP2, does not affect the compare flag; thus, it does not generate an interrupt.

INPUT CAPTURE REGISTERS

There are two input capture registers: input capture register 1 (ICR1) and input capture register 2 (ICR2).

NOTE

The same input capture interrupt enable bit (ICIE) is used for the two input capture registers.

Input Capture Register 1

Two 8-bit registers that make up the 16-bit input capture register 1 (ICR1) are read-only and are used to latch

the value of the free-running counter after the corresponding input capture edge detector senses a defined transition. The level transition that triggers the counter transfer is defined by the corresponding input edge bit (IEDG1). Reset does not affect the contents of the input capture register.

The result obtained by an input capture will be one more than the value of the free-running counter on the rising edge of the internal bus clock preceding the external transition. This delay is required for internal synchronization. Resolution is one count of the free-running counter, which is four internal-bus clock cycles.

The free-running counter contents are transferred to the input capture register on each proper signal transition, regardless of whether the input capture flag (ICF1) is set or clear. The input capture register always contains the free-running counter value, which corresponds to the most recent input capture.

After a read of the input capture register 1 ($14) most significant byte, the counter transfer is inhibited until the least significant byte ($15) is also read. This characteristic causes the time used in the input capture software routine and its interaction with the main program to determine the minimum pulse period.

A read of the input capture register 1 least significant byte ($15) does not inhibit the free-running counter transfer since they occur on opposite edges of the internal bus clock.

Input Capture Register 2

The input capture register 2 (ICR2) is a 16-bit register that is composed of two 8-bit registers at locations $1C (most significant byte) and $1D (least significant byte). Input capture register 2 functions identically to input capture register 1, except that only negative edge sensitivity is available. By substituting the appropriate bits in the timer status register ($13) and substituting register locations, the ICR1 description applies to ICR2.

TIMER CONTROL REGISTER (TCR) $12

The TCR is an 8-bit read/write register, illustrated below with a definition of each bit.

7	6	5	4	3	2	1	0
ICIE	OCIE	TOIE	FOLV2	FOLV1	OLVL1	IEDG1	OLVL1
RESET:							
0	0	0	0	0	0	U	0

U = Uneffected by RESET

ICIE — Input Capture Interrupt Enable
- 1 = Interrupt enabled
- 0 = Interrupt disabled

OCIE — Output Compare Interrupt Enable
- 1 = Interrupt enabled
- 0 = Interrupt disabled

TOIE — Timer Overflow Interrupt Enable
- 1 = Interrupt enabled
- 0 = Interrupt disabled

FOLV2 — Force Output Compare 2
- 1 = Forces the OLVL2 bit to the corresponding output latch
- 0 = No effect

FOLV1 — Force Output Compare 1
- 1 = Forces the OLVL1 bit to the corresponding output latch
- 0 = No effect

OLVL2 — Output Level 2
- 1 = The value of the output level 2 bit, which is copied to the output level latch by the next successful output compare 2, appears at TCMP2
- 0 = No effect

IEDG1 — Input Edge

Value of input edge determines which level transition on TCAP1 pin will trigger free-running counter transfer to the input capture register.
- 1 = Positive edge
- 0 = Negative edge

OLVL1 — Output Level 1

Value of output level 1, which is copied into output level register by the next successful output compare 1, will appear on the TCMP1 pin.
- 1 = High output
- 0 = Low output

TIMER STATUS REGISTER (TSR) $13

The TSR is a read-only register containing three status flag bits. Bits 0–4 always read zero.

7	6	5	4	3	2	1	0
ICF1	OCF1	TOF	ICF2	OCF2	—	—	—
RESET:							
U	U	U	U	U	—	—	—

ICF1 — Input Capture Flag 1
- 1 = Flag set when selected polarity edge is sensed by input capture edge detector
- 0 = Flag cleared when TSR and input capture 1 low register ($15) are accessed

OCF1 — Output Capture Flag 1
- 1 = Flag set when output compare register contents match the free-running counter contents
- 0 = Flag cleared when TSR and output compare 1 low register ($17) are accessed

TOF — Timer Overflow Flag
- 1 = Flag set when free-running counter transition from $FFFF to $0000 occurs
- 0 = Flag cleared when TSR and counter low register ($19) are accessed

ICF2 — Input Capture Flag 2
- 1 = Flag set when selected polarity edge is sensed by input capture 2 edge detector
- 0 = Flag cleared when TSR and input capture 2 low register ($1D) are accessed

OCF2 — Output Capture Flag 2
- 1 = Flag set when output compare register contents match the free-running counter contents
- 0 = Flag cleared when TSR and output compare low register 2 ($1F) are accessed

Bits 0–2 — Not Used

Can read either zero or one.

TIMER DURING WAIT MODE

The CPU clock halts during the WAIT mode, but the timer remains active. An interrupt from the timer causes the processor to exit the WAIT mode.

TIMER DURING STOP MODE

In the STOP mode, the timer stops counting and holds the last count value if STOP is exited by an interrupt. If reset is used, the counter is forced to $FFFC. During STOP, if at least one valid input capture edge occurs at the TCAP pin, the input capture detect circuit is armed. This does not set any timer flags or wake up the MCU, but when the MCU does wake up, there is an active input capture flag and data from the first valid edge that occurred during the STOP mode. If reset is used to exit STOP mode, then no input capture flag or data remains, even if a valid input capture edge occurred.

Accessing the timer status register satisfies the first condition required to clear status bits. The remaining step is to access the register corresponding to the status bit. A problem can occur when using the timer overflow function and reading the free-running counter at random times to measure an elapsed time. Without incorporating the proper precautions into software, the timer overflow flag could unintentionally be cleared if 1) the timer status register is read or written when TOF is set, and 2) the least significant byte of the free-running counter is read but not for the purpose of servicing the flag.

The counter alternate register at address $1A and $1B contains the same value as the free-running counter (at address $18 and $19); therefore, this alternate register can be read at any time without affecting the timer overflow flag in the timer status register.

SERIAL COMMUNICATIONS INTERFACE

A full-duplex asynchronous SCI is provided with a standard NRZ format and a variety of baud rates. The SCI transmitter and receiver are functionally independent but use the same data format and baud rate prescaler. The terms baud and bit rate are used synonymously in the following description.

SCI TWO-WIRE SYSTEM FEATURES

- Standard NRZ (mark/space) format
- Advanced error detection method includes noise detection for noise duration of up to one-sixteenth bit time
- Full-duplex operation (simultaneous transmit and receive)
- Software programmable for one of 32 different baud rates
- Different baud rates possible for transmit and receive
- Software-selectable word length (eight- or nine-bit words)
- Separate transmitter and receiver enable bits
- SCI may be interrupt driven
- Four separate interrupt conditions

SCI RECEIVER FEATURES

- Receiver wake-up function (idle or address bit)
- Idle line detect
- Framing error detect
- Noise detect
- Overrun detect
- Receiver data register full flag

SCI TRANSMITTER FEATURES

- Transmit data register empty flag
- Transmit complete flag
- Break send

Any SCI two-wire system requires receive data in (RDI) and transmit data out (TDO).

DATA FORMAT

Receive data in (RDI) or transmit data out (TDO) is the serial data presented between the internal data bus and the output pin (TDO) and between the input pin (RDI) and the internal data bus. Data format is as shown for the NRZ in Figure 11.

WAKE-UP FEATURE

In a typical multiprocessor configuration, the software protocol will usually identify the addressee(s) at the beginning of the message. To permit uninterested MPUs to ignore the remainder of the message, a wake-up feature is included, whereby all further SCI receiver flag (and interrupt) processing can be inhibited until its data line returns to the idle state. An SCI receiver is re-enabled by an idle string of at least ten (or eleven) consecutive ones. Software for the transmitter must provide for the required idle string between consecutive messages and prevent it from occurring within messages.

A second wake-up method is available in which sleeping SCI receivers can be awakened by a logic one in the high-order bit of a received character.

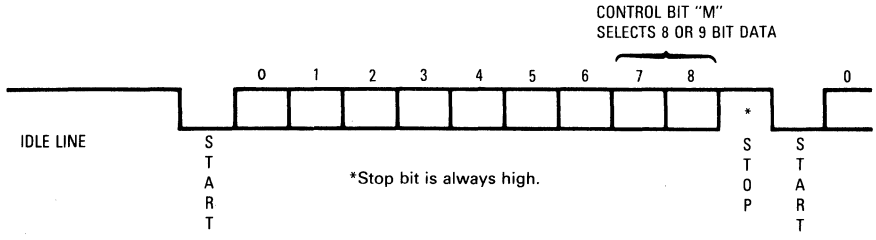


Figure 11. Data Format

RECEIVE DATA IN

Receive data in (RDI) is the serial data which is presented from the input pin via the SCI to the receive data register (RDR). While waiting for a start bit, the receiver samples the input at a rate 16 times higher than the set baud rate. This increased rate is referred to as the RT rate. When the input (idle) line is detected low, it is tested for three more sample times. If at least two of these three samples detect a logic low, a valid start bit is assumed to be detected. If in two or more samples, a logic high is detected, the line is assumed to be idle. The receive clock generator is controlled by the baud rate register (see Figure 13); however, the SCI is synchronized by the start bit independent of the transmitter. Once a valid start bit is detected, the start bit, each data bit, and the stop bit are each sampled three times. The value of the bit is determined by voting logic, which takes the value of a majority of samples. A noise flag is set when all three samples on a valid start bit, data bit, or stop bit do not agree. A noise flag is also set when the start verification samples do not agree.

3

START BIT DETECTION FOLLOWING A FRAMING ERROR

If there has been a framing error (FE) without detection of a break (10 zeros for 8-bit format or 11 zeros for a 9-bit format), the circuit continues to operate as if there actually were a stop bit, and the start edge will be placed artificially. The last bit received in the data shift register is inverted to a logic one, and the three logic-one start qualifiers are forced into the sample shift register during the interval when detection of a start bit is anticipated; therefore, the start bit will be accepted no sooner than it is anticipated.

If the receiver detects that a break (RDRF = 1, FE = 1, receiver data register = $00) produced the framing error, the start bit will not be artificially induced, and the receiver must actually receive a logic one before start.

SCI SYNCHRONOUS TRANSMISSION

The SCI transmitter allows a one-way synchronous transmission, with the SCLK pin as the clock output. No clock is sent to the SCLK pin during the stop and start bits. The LCL bit (SSCR1) controls whether clocks are active during the last valid data bit (address mark). The CPOL bit selects clock polarity, and the CPHA bit selects the phase of the external clock. During idle, preamble, and send break, the external SCLK clock is not active.

These options allow the SCI to control serial peripherals consisting of shift registers without losing any function of the SCI transmitter. These options do not affect the SCI receiver, which is independent of the transmitter.

The SCLK pin works in conjunction with the TDO pin. When the SCI transmitter is disabled, the SCLK and the TDO pins assume a high-impedance state.

NOTE

THE LBCL, CPOL and CPHA bits must be selected before the transmitter is enabled to ensure that the clocks function correctly. These bits should not be changed while the transmitter is enabled.

TRANSMIT DATA OUT

Transmit data out (TDO) is the serial data presented from the transmit data register (TDR) via the SCI to the output pin. The transmitter generates a bit time by using a derivative of the RT clock, producing a transmission rate equal to one-sixteenth that of the receiver sample clock (if the same baud rate is used for transmit and receive).

FUNCTIONAL DESCRIPTION

A block diagram of the SCI is shown in Figure 12. The user has option bits in the serial communications control register 1 (SCCR1) to determine the SCI wake-up method and data word length. Serial communications control register 2 (SCCR2) provides control bits that individually enable/disable the transmitter or receiver, enable system interrupts, and provide wake-up enable, and send break code bits. The baud rate register bits allow the user to select different baud rates for the transmitter and receiver.

Data transmission is initiated by a write to the serial communications data register (SCDAT). Provided the transmitter is enabled, data stored in the SCDAT is transferred to the transmit data shift register. This data transfer sets the SCI status register (SCSR) transmit data register empty (TDRE) bit and generates an interrupt if the transmit interrupt is enabled. Data transfer to the transmit data shift register is synchronized with the bit rate clock. All data is transmitted LSB first. Upon completion of data transmission, the transmission complete (TC) bit is set (provided no pending data, preamble, or break code is sent), and an interrupt is generated if the transmit complete interrupt is enabled. If the transmitter is disabled, and the data, preamble, or break code has been sent, the TC bit will also be set, which will also generate an interrupt if the TCIE bit is set. If the transmitter is disabled in the middle of a transmission, that character will be completed before the transmitter gives up control of the TDO pin.

When the SCDAT is read, it contains the last data byte received, provided that the receiver is enabled. The SCSR receive data register full (RDRF) bit is set to indicate that a data byte is transferred from the input serial shift register to the SCDAT, which can cause an interrupt if the receiver interrupt is enabled. Data transfer from the input serial shift register to the SCDAT is synchronized by the receiver bit rate clock. The SCSR overrun (OR), noise flag (NF), or FE bits are set if data reception errors occur.

An idle line interrupt is generated if the idle line interrupt is enabled and the SCSR IDLE bit (which detects idle line transmission) is set. This allows a receiver that is not in the wake-up mode to detect the end of a message, the preamble of a new message, or to resynchronize with the transmitter. A valid character must be received before the idle line condition for the IDLE bit to be set and for an idle line interrupt to be generated.

REGISTERS

There are five registers used in the SCI; the internal configuration of these registers is discussed in the following paragraphs.

Serial Communications Data Register (SCDAT) $11

The SCDAT is a read/write register used to receive and transmit SCI data.

7	6	5	4	3	2	1	0
SCD7	SCD6	SCD5	SCD4	SCD3	SCD2	SCD1	SCD0
RESET:							
U	U	U	U	U	U	U	U

As shown in Figure 12, SCDAT functions as two separate registers. The transmit data register (TDR) provides the parallel interface from the internal data bus to the transmit shift register. The receive data register (RDR) provides the interface from the receive shift register to the internal data bus.

Serial Communications Control Register 1 (SCCR1) $0E

The SCCR1 provides control bits that determine word length, select the wake-up method, and control the options to output the transmitter clocks for synchronous transmissions.

7	6	5	4	3	2	1	0
R8	T8	—	M	WAKE	CPOL	CPHA	LBCL
RESET:							
U	U	—	U	U	U	U	U

R8 — Receive Data Bit 8
R8 bit provides storage location for the ninth bit in the receive data byte (if M = 1).
T8 — Transmit Data Bit 8
T8 bit provides storage location for the ninth bit in the transmit data byte (if M = 1).
M — SCI Character Word Length
1 = one start bit, nine data bits, one stop bit
0 = one start bit, eight data bits, one stop bit
WAKE — Wake-Up Select
Wake bit selects the receiver wake-up method.
1 = Address bit (most significant bit)
0 = Idle line condition
CPOL - Clock Polarity
Selects the clock polarity sent to the SCLK pin.
1 = Steady state high outside the transmission window
0 = Steady state low outside the transmission window
The CPOL bit should not be changed with the transmitter active.
CPHA — Clock Phase
Selects the clock phase sent to the SCLK pin.
1 = SCLK line activated at the beginning of the data bit
0 = SCLK line activated in the middle of the data bit (see Figures 13 and 14)
The CPHA bit should not be changed with the transmitter active.
LBCL — Last Bit Clock
Selects whether the clock associated with the last data bit transmitted is output to the SCLK pin.
1 = Last data bit output
0 = Last data bit not output
The last data bit is the eighth or ninth bit, depending on whether an 8- or 9-bit format is used (see Table 5).
The LBCL bit should not be changed while the transmitter is enabled.
Bit 5 — Not used
Can read either one or zero

Table 5. SCI Clock on SCLK Pin

Data Format	M Bit	LBCL Bit	Number of Clocks on SCLK Pin
8 Bit	0	0	7
8 Bit	0	1	8
9 Bit	1	0	8
9 Bit	1	1	9

The address bit is dependent on both the wake-bit and the M-bit level. Additionally, the receiver does not use the wake-up feature unless the RWU control bit in SCCR2 is set.

Wake	M	Receiver Wake-Up
0	X	Detection of an idle line allows the next data byte received to cause the receive data register to fill and produce an RDRF flag.
1	0	Detection of a received one in the eighth data bit allows an RDRF flag and associated error flags.
1	1	Detection of a received one in the ninth data bit allows an RDRF flag and associated error flags.

Serial Communications Control Register 2 (SCCR2) $0F

The SCCR2 provides control of individual SCI functions such as interrupts, transmit/receive enabling, receiver wake-up, and break code.

7	6	5	4	3	2	1	0
TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
RESET:							
0	0	0	0	0	0	0	0

TIE — Transmit Interrupt Enable
1 = SCI interrupt enabled, provided TDRE is set
0 = TDRE interrupt disabled
TCIE — Transmit Complete Interrupt Enable
1 = SCI interrupt enabled, provided TC is set
0 = TC interrupt disabled
RIE — Receive Interrupt Enable
1 = SCI interrupt enabled, provided OR or RDRF is set
0 = RDRF and OR interrupts disabled
ILIE — Idle Line Interrupt Enable
1 = SCI interrupt enabled, provided IDLE is set
0 = Idle interrupt disabled

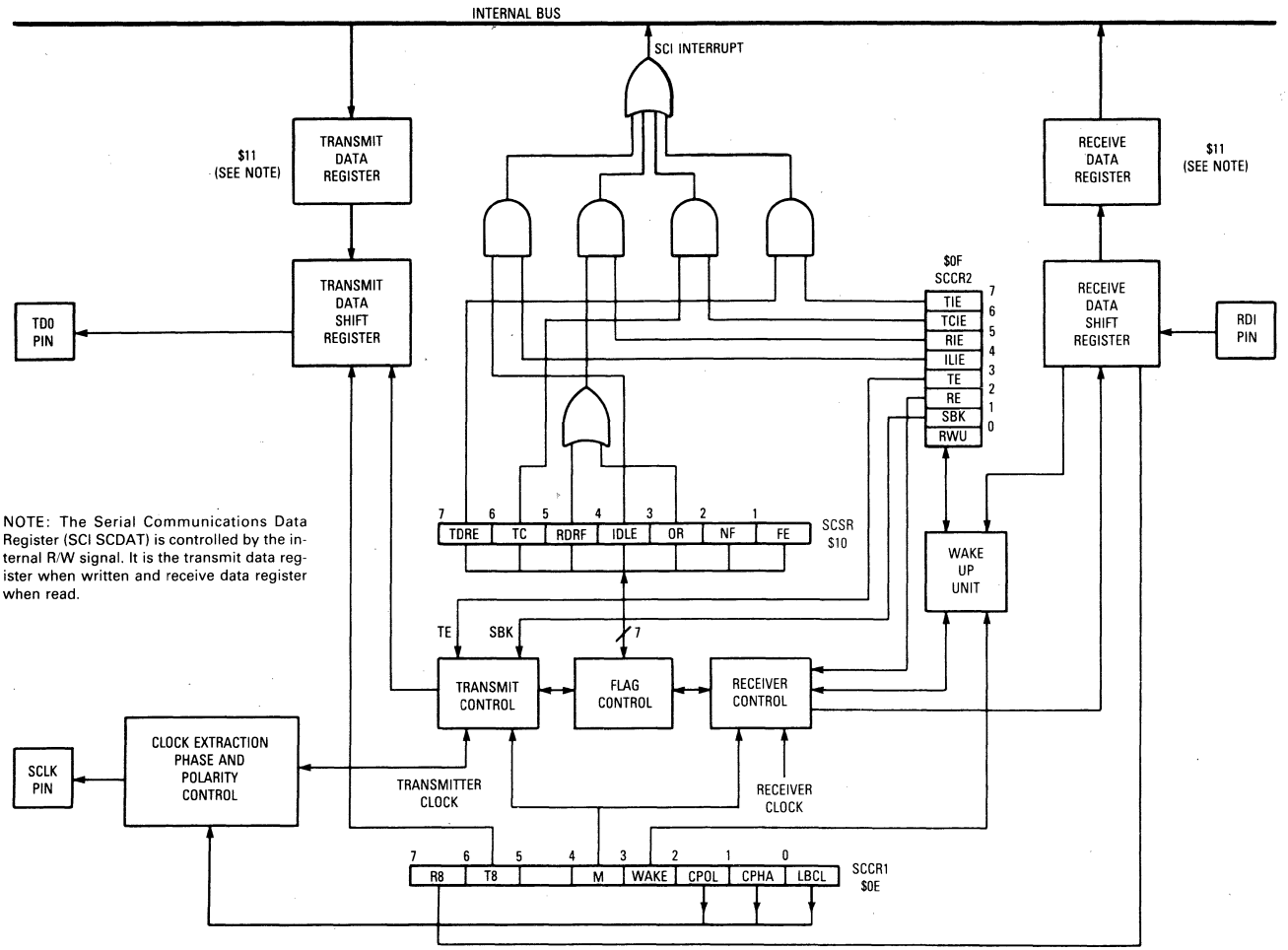


Figure 12. SCI Block Diagram

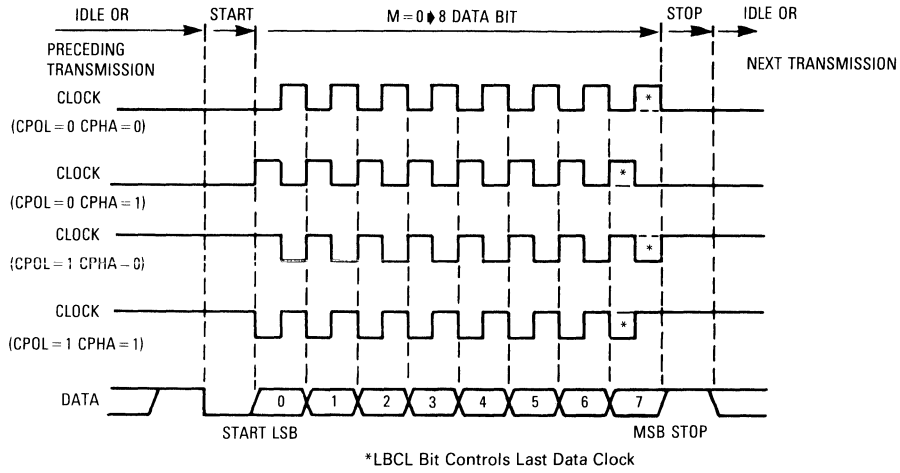


Figure 13. SCI Data Clock Timing Diagram (M = 0)

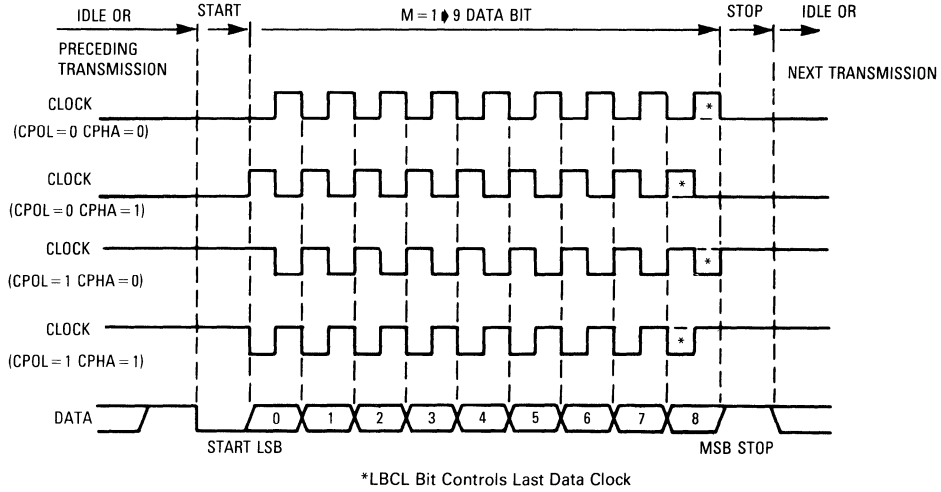


Figure 14. SCI Data CLock Timing Diagram (M = 1)

TE — Transmit Enable
- 1 = Transmit shift register output is applied to the TD0 line, and the corresponding clocks are applied to the SCLK pin. Depending upon the SCCR1 M bit, a preamble of 10 (M = 0) or 11 (M = 1) consecutive ones is transmitted.
- 0 = Transmitter disabled after last byte is loaded in the SCDAT and TDRE is set. After last byte is transmitted, TD0 line becomes a high-impedance line.

RE — Receive Enable
- 1 = Receiver shift register input is applied to the RDI line.
- 0 = Receiver disabled and RDRF, IDLE, OR, NF, and FE status bits are inhibited.

RWU — Receiver Wake-Up
- 1 = Places receiver in sleep mode and enables wake-up function
- 0 = Wake-up function disabled after receiving data word with MSB set (if WAKE = 1)

Wake-up function also disabled after receiving 10 (M = 0) or 11 (M = 1) consecutive ones (if WAKE = 0)

SBK — Send Break

1 = Transmitter continually sends blocks of zeros (sets of 10 or 11) until cleared. Upon completion of break code, transmitter sends one high bit for recognition of valid start bit.

0 = Transmitter sends 10 (M = 0) or 11 (M = 1) zeros then reverts to an idle state or continues sending data. If transmitter is empty and idle, setting and clearing the SBK bit may queue up to two character times of break because the first break transfers immediately to the shift register, and the second is queued into the parallel transmit buffer.

Serial Communications Status Register (SCSR) $10

The SCSR provides inputs to the SCI interrupt logic circuits. Noise flag and framing error bits are also contained in the SCSR.

7	6	5	4	3	2	1	0
TDRE	TC	RDRF	IDLE	OR	NF	FE	—
RESET:							
1	1	0	0	0	0	0	—

TDRE — Transmit Data Register (TDR) Empty

1 = TDR contents transferred to the transmit data shift register

0 = TDR still contains data. TDRE is cleared by reading the SCSR, followed by a write to the TDR.

TC — Transmit Complete

1 = Indicates end of data frame, preamble, or break condition has occurred if:

1. TE = 1, TDRE = 1, and no pending data, preamble or break is to be transmitted; or
2. TE = 0 and the data preamble or break (in the transmit shift register) has been transmitted.

0 = TC bit cleared by reading the SCSR, followed by a write to the TDR

The TC bit is a status register that indicates one of the above conditions has occurred. It does not inhibit the transmitter in any way.

RDRF — Receive Data Register (RDR) Full

1 = Receive data shift register contents transferred to the RDR

0 = Receive data shift register transfer did not occur. RDRF is cleared by reading the SCSR, followed by a read of the RDR

IDLE — Idle Line Detect

1 = Indicates receiver has detected an idle line

0 = IDLE is cleared by reading the SCSR, followed by a read of the RDR. Once IDLE is cleared, IDLE cannot be set until RDI line becomes active and idle again.

OR — Overrun Error

1 = Indicates receive data shift register data is ready to be sent to a full RDR (RDRF = 1). Data causing the overrun is lost, and RDR data is not disturbed.

0 = OR is cleared by reading the SCSR, followed by a read of the RDR.

NF — Noise Flag

1 = Indicates noise is present on the receive bits, including the start and stop bits. NF is not set until RDRF = 1.

0 = NF is cleared by reading the SCSR, followed by a read of the RDR.

FE — Framing Error

1 = Indicates stop bit not detected in received data character. FE is set the same time RDRF is set. If received byte causes both framing and overrun errors, processor will only recognize the overrun error. Further data transfer into the RDR is inhibited until FE is cleared.

0 = FE is cleared by reading the SCSR, followed by a read of the RDR.

Bit 0 — Not used

Can read either one or zero

Baud Rate Register $0D

The baud rate register selects the SCI transmitter and receiver baud rate. The SCP1 and SCP0 prescaler bits are used in conjunction with the SCR2–SCR0 bits to generate the receiver baud rate and in conjunction with the SCT2–SCT0 baud rate bits to generate the transmitter baud rate.

Tables 6 and 7 tabulate the divide chain used to obtain the baud rate clock (transmit or receive clock). The actual divider chain is controlled by the combined SCP1–SCP0 and SCR2–SCR0 or SCT2–SCT0 bits in the baud rate register. The divided frequencies shown in Table 6 represent the final baud rate that results from prescaler division only (SCR or SCT bits all zero). Table 7 lists the prescaler output frequency divided by the action of the SCR or SCT bits.

For example, assume that 9600-Hz baud rate is desired from a 2.4576-MHz system clock crystal. The prescaler bits could be set for either a divide-by-one or divide-by-four. If a divide-by-four prescaler is used, then the SCR and SCT bits must be set for divide-by-two. The same result, using the same crystal frequency, can be obtained with a prescaler divide-by-one and SCR and SCT bit divide-by-eight.

7	6	5	4	3	2	1	0
SCP1	SCP0	SCT2	SCT1	SCT0	SCR2	SCR1	SCR0
RESET:							
0	0	U	U	U	U	U	U

SCP1–SCP0 — SCI Prescaler Bits 1 and 0

These two prescaler bits are used to increase the range of standard baud rates controlled by the SCT2-SCT0 and SCR2–SCR0 bits. Prescaler internal processor clock division versus bit levels are shown in Table 6.

SCT2-SCT0 — SCI Transmit Baud Rate Selection Bits

These three bits, taken in conjunction with bits SCP1–SCP0, are used to select the SCI transmit baud rate. Baud rates versus bit levels are listed in Table 7.

Table 6. Prescaler Highest Baud Rate Frequency Output

SCP Bit		Clock* Divided By	Crystal Frequency MHz				
1	0		4.194304	4.0	2.4576	2.0	1.8432
0	0	1	131.072 kHz	125.000 kHz	76.80 kHz	62.50 kHz	57.60 kHz
0	1	3	43.691 kHz	41.666 kHz	25.60 kHz	20.833 kHz	19.20 kHz
1	0	4	32.768 kHz	31.250 kHz	19.20 kHz	15.625 kHz	14.40 kHz
1	1	13	10.082 kHz	9600 Hz	5.907 kHz	4800 Hz	4430 Hz

*Refers to the internal processor clock.

NOTE: The divided frequencies shown in Table 6 represent baud rates which are the highest transmit baud rate (Tx) that can be obtained by a specific crystal frequency and only using the prescaler division. Lower baud rates may be obtained by providing a further division using the SCI rate select bits as shown below for some representative prescaler outputs.

Table 7. Transmit Baud Rate Output for a Given Prescaler Output

SCR/T Bits			Divided By	Representative Highest Prescaler Baud Rate Output				
2	1	0		131.072 kHz	32.768 kHz	76.80 kHz	19.20 kHz	9600 Hz
0	0	0	1	131.072 kHz	32.768 kHz	76.80 kHz	19.20 kHz	9600 Hz
0	0	1	2	65.536 kHz	16.384 kHz	38.40 kHz	9600 Hz	4800 Hz
0	1	0	4	32.768 kHz	8.192 kHz	19.20 kHz	4800 Hz	2400 Hz
0	1	1	8	16.384 kHz	4.096 kHz	9600 Hz	2400 Hz	1200 Hz
1	0	0	16	8.192 kHz	2.048 kHz	4800 Hz	1200 Hz	600 Hz
1	0	1	32	4.096 kHz	1.024 kHz	2400 Hz	600 Hz	300 Hz
1	1	0	64	2.048 kHz	512 Hz	1200 Hz	300 Hz	150 Hz
1	1	1	128	1.024 kHz	256 Hz	600 Hz	150 Hz	75 Hz

NOTE: Table 7 illustrates how the SCI select bits can be used to provide lower transmitter or receiver baud rates by further dividing the prescaler output frequency. The five examples are only representative samples. In all cases, the baud rates shown are transmit baud rates (transmit clock), and the receive clock is 16 times higher in frequency than the actual baud rate.

SCR2-SCR0 — SCI Receive Baud Rate Selection Bits
These three bits, taken in conjunction with bits SCP1–SCP0, are used to select the SCI receive baud rate. Baud rates versus bit levels are listed in Table 7.

Load Program in RAM and Execute

This function is entered if the following conditions are met when reset is released:

IRQ is at V_{DD} + 4 V for at least two machine cycles after reset
TCAP1 is at V_{DD} for at least two machine cycles after reset
PD3 is at V_{DD} for at least 30 machine cycles after reset
PD4 is at V_{SS} for at least 30 machine cycles after reset

User programs are loaded into RAM using the SCI port and then executed. Data is loaded sequentially, starting at RAM location $50, until the last byte is loaded. Program control is then transferred to the RAM progam starting at location $51. The first byte loaded is the count of the number of bytes in the program plus the count byte. The program starts at the second byte in the RAM. During firmware initialization, the SCI is configured for the NRZ format (idle line, eight data bits, and stop bit). The baud rate is 9600 with a 4-MHz crystal. Figure 15 shows a schematic for the load program in RAM and execute function.

Immediate execution can be avoided by setting the byte count to a value greater than the length of data loaded, which causes the firmware to wait for additional data after loading is complete. Resetting the MCU then allows entering any routine without disturbing the RAM data that was loaded.

Jump to Any Address

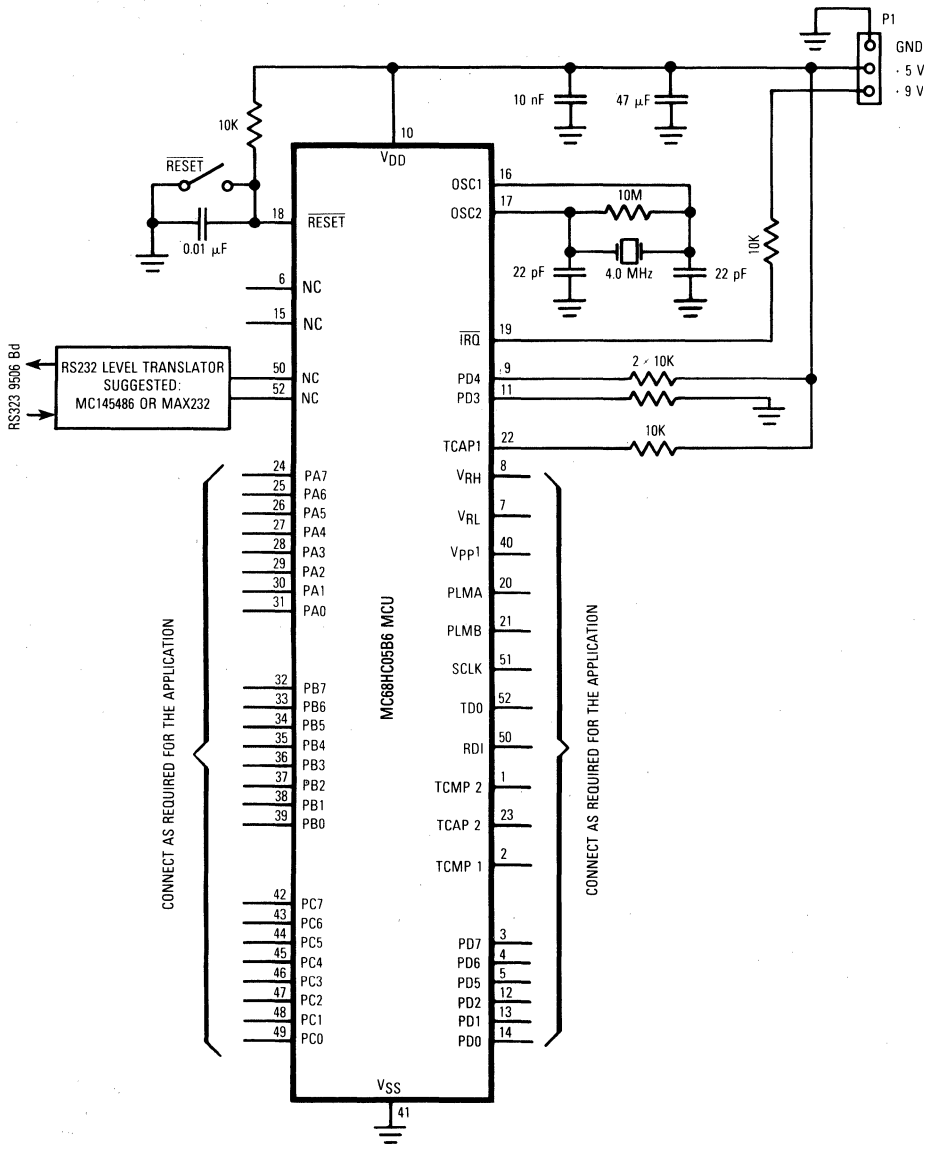
This function is entered if the following conditions are met when reset is released:

IRQ is at V_{DD} + 4 V for at least two machine cycles after reset
TCAP1 is at V_{DD} for at least two machine cycles after reset
PD3 is at V_{DD} for at least 30 machine cycles after reset
PD4 is at V_{DD} for at least 30 machine cycles after reset

To execute the jump to any address function, port A data input should be $CC, and port B and C should be the MSB and LSB, respectively, of the address desired for the jump. Figure 16 shows a schematic for the jump function.

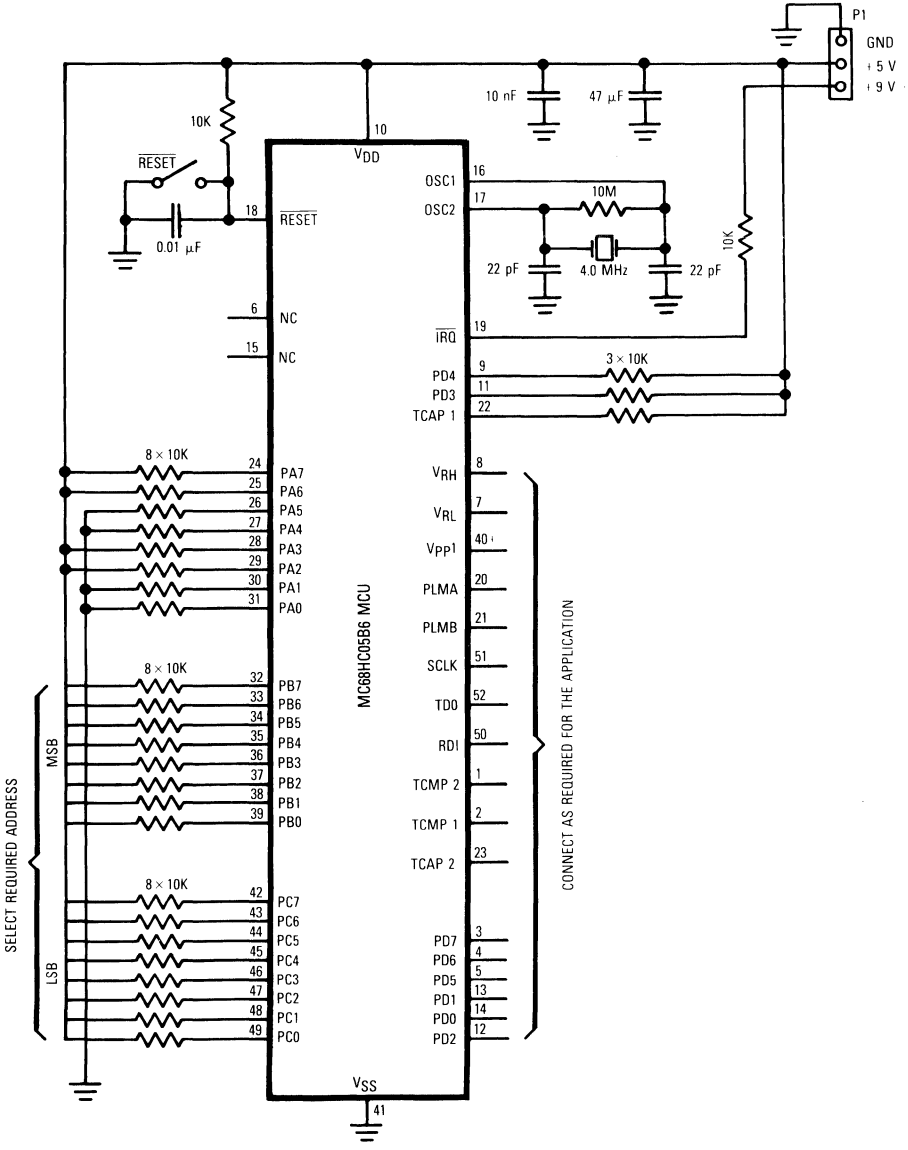
PULSE-LENGTH D/A CONVERTERS

The pulse-length D/A converter (PLM) works in conjunction with the timer to execute two 8-bit conversions



NOTE: Pin numbers are valid for the 52-pin PLCC packge only.

Figure 15. Load Program in RAM and Execute Diagram



NOTE: Pin numbers are valid for 52-pin PLCC package only.

Figure 16. Jump to Any Address Diagram

with a choice of two repetition rates. The outputs are pulse-length modulated signals whose duty-cycle ratio may be modified. These signals can be used directly as PLM, or the filtered average values can be used as general-purpose analog outputs.

Registers PLMA and PLMB contain the pulse-length values for the two PLMs. A value of $00 results in a continuously low output from the D/A. A value of $80 results in a 50-percent duty-cycle output, and a value of $FF gives an output that is a logic 1 for 255/256 of the cycle. When the MCU writes to the PLMA or PLMB register, the D/A picks up the new value at the end of a complete conversion cycle. A monotonic change in the dc component of the output results, without overshoots or vicious starts (a vicious start is an output that gives totally erroneous output during the first cycle following an update of the registers). WAIT mode does not affect the output waveform of the D/A converters.

NOTE

Since the PLM system uses the timer counter, PLM results will be affected while resetting the timer counter.

Figure 17 shows a block diagram of the PLM system.

PLMA (0A)

7	6	5	4	3	2	1	0
PLMA7	PLMA6	PLMA5	PLMA4	PLMA3	PLMA2	PLMA1	PLMA0
RESET:							
0	0	0	0	0	0	0	0

PLMB (0B)

7	6	5	4	3	2	1	0
PLMB7	PLMB6	PLMB5	PLMB4	PLMB3	PLMB2	PLMB1	PLMB0
RESET:							
0	0	0	0	0	0	0	0

Miscellaneous (0C)

7	6	5	4	3	2	1	0
—	—	—	—	SFA	SFB	—	—
RESET:							
—	—	—	—	0	0	—	—

SFA — Slow/Fast Control for PLMA Clock
1 = Slow speed of PLMA used (4096 times the timer clock period)
0 = Fast speed of PLMA used (256 times the timer clock period)

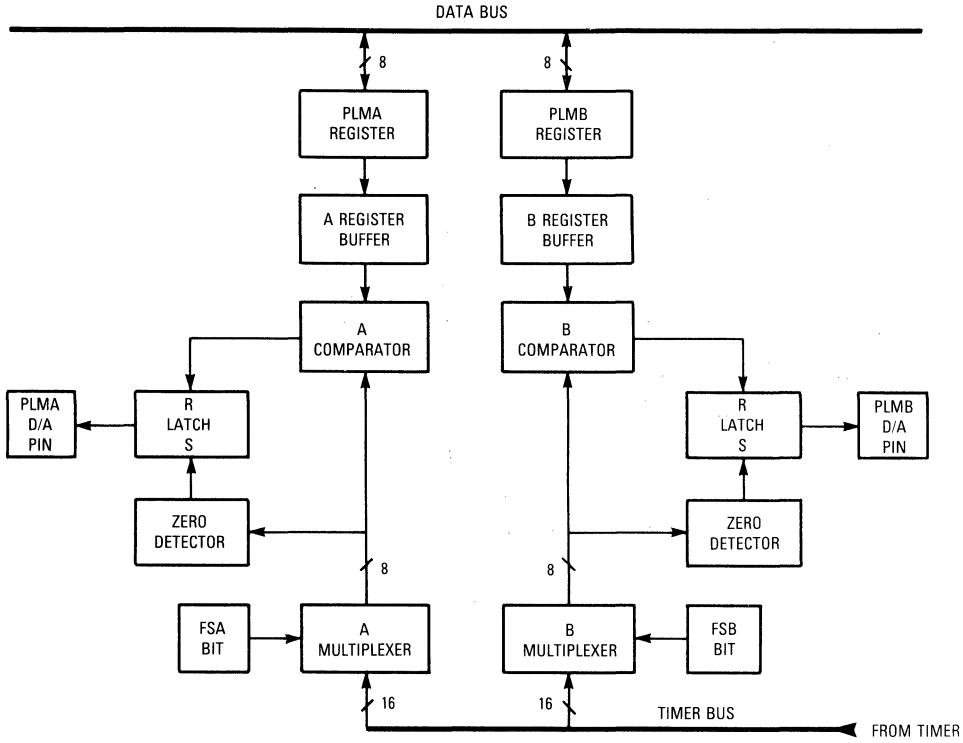


Figure 17. PLM Block Diagram

SFB — Slow/Fast Control for PLMB Clock
1 = Slow speed of PLMB used (4096 times the timer clock period)
0 = Fast speed of PLMB used (256 times the timer clock period)

NOTE

The highest speed of the PLM system corresponds to the frequency of the TOF bit being set, multiplied by 256. The slowest speed of the PLM system corresponds to the frequency of the TOF bit being set, multipled by 16.

The SFA and SFB bits are not double buffered; therefore, these bits must be selected before writing to either PLM register to avoid temporary wrong values from the PLM outputs. Figure 18 shows some examples of the PLM output waveforms.

A/D CONVERTER

The A/D converter system consists of an 8-bit successive approximation converter and a 16-channel multiplexer. Eight of the channels are available for output, and the other eight channels are dedicated to internal test functions. There is one 8-bit result data register (address $08) and one 8-bit status/control register (address $09).

NOTE

In the 48-pin dual-in-line package, the fixed input port (D) of the MC68HC05B6 is reduced to six pins (PD5–PD0, AN5–AN0). This change has no effect on either programming or operation of port D or the A/D converter.

The reference supply for the converter uses dedicated input pins instead of the power supply lines, because drops caused by loading in the power supply lines would degrade the accuracy of the A/D conversion. An internal RC oscillator is available if the bus speed is low enough to degrade the A/D accuracy. An ADON bit allows the A/D to be switched off to reduce power consumption, which is particularly useful in the WAIT mode.

For ratiometric conversions, the source of each analog input should use V_{RH} as the supply voltage and be referenced to V_{RL}. An input voltage greater than or equal to V_{RH} converts as $FF (full scale) with no overflow indication. An input voltage equal to V_{RL} converts as $00. The conversion is monotonic with no missing codes.

A/D STATUS/CONTROL REGISTER ($09)

7	6	5	4	3	2	1	0
COCO	ADRC	ADON	0	CH3	CH2	CH1	CH0
RESET:							
0	0	0	0	0	0	0	0

COCO — Conversion Complete
1 = Conversion is complete; a new result can be read from the result data register ($08).
0 = No conversion since last reset

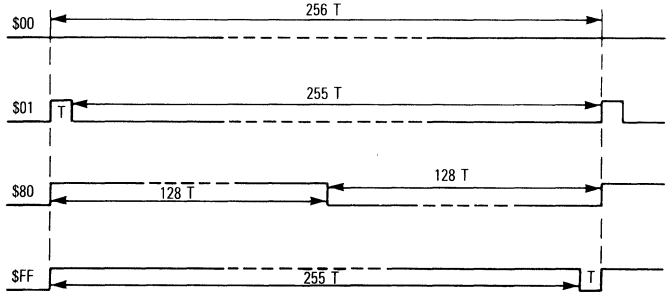
ADRC — A/D RC Oscillator Control
1 = A/D uses RC clock
0 = A/D uses CPU clock
When the RC oscillator is turned on, it requires a time t_{adrc} to stabilize, and results can be inaccurate during this time.

ADON — A/D On
1 = A/D enabled
0 = A/D disabled
When the A/D is turned on, it requires a time t_{ADON} for the current sources to stabilize, and results can be inaccurate during this time.

CH3–CH0 — Channel 3 through Channel 0
These bits select the A/D channel assignment (see Table 8).

NOTE

Using one or more pins of PD7/AN7–PD0/AN0 as analog inputs does not affect the ability to use port D inputs as digital inputs. However, using port D



T = 4 CPU Clocks Fast Mode
T = 64 CPU Clocks Slow Mode

Figure 18. PLM Output Waveform Examples

for digital inputs during an analog conversion sequence may inject noise on the analog inputs and reduce the accuracy of the A/D result.

Performing a digital read of port D with levels other than V_{DD} or V_{SS} on the inputs causes greater than normal power dissipation during the read and may give erroneous results.

Table 8. A/D Channel Assignments

CH3	CH2	CH1	CH0	Channel Selected
0	0	0	0	AN0, Port D Bit 0
0	0	0	1	AN1, Port D Bit 1
0	0	1	0	AN2, Port D Bit 2
0	0	1	1	AN3, Port D Bit 3
0	1	0	0	AN4, Port D Bit 4
0	1	0	1	AN5, Port D Bit 5
0	1	1	0	AN6, Port D Bit 6
0	1	1	1	AN7, Port D Bit 7
1	0	0	0	V_{RH} Pin (High)
1	0	0	1	$((V_{RH})+(V_{RL}))/2$
1	0	1	0	V_{RL} Pin (Low)
1	0	1	1	V_{RL} Pin (Low)
1	1	0	0	V_{RL} Pin (Low)
1	1	0	1	V_{RL} Pin (Low)
1	1	1	0	V_{RL} Pin (Low)
1	1	1	1	V_{RL} Pin (Low)

3

INSTRUCTION SET

The MCU instructions can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

This MCU uses all the instructions available in the M146805 CMOS Family plus one more: the unsigned multiply (MUL) instruction. This instruction allows unsigned multiplication of the contents of the accumulator (A) and the index register (X). The high-order product is then stored in the index register, and the low-order product is stored in the accumulator. A detailed definition of the MUL instruction is shown below.

Operation	X:A ♦ X × A		
Description	Multiplies the eight bits in the index register by the eight bits in the accumulator to obtain a 16-bit unsigned number in the concatenated accumulator and index register		
Condition Codes	H: Cleared I: Not affected N: Not affected Z: Not affected C: Cleared		
Source Form(s)	MUL		
Addressing Mode	Cycles	Bytes	Opcode
Inherent	11	1	$42

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following instruction list.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two-byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Higher	BHI
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
Branch if Higher or Same	BHS
Branch if Carry Set	BCS
Branch if Lower	BLO
Branch if Not Equal	BNE
Branch if Equal	BEQ
Branch if Half Carry Clear	BHCC
Branch if Half Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Bit is Clear	BMC
Branch if Interrupt Mask Bit is Set	BMS
Branch if Interrupt Line is Low	BIL
Branch if Interrupt Line is High	BIH
Branch to Subroutine	BSR

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (Twos Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST
Multiply	MUL

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No-Operation	NOP
Stop	STOP
Wait	WAIT

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any writable bit which resides in the first 256 bytes of the memory space where all port registers, port DDRs, timer, timer control, ROM, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear and bit test, and branch functions are all implemented with a single instruction. For test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch if Bit n is Set	BRSET n (n = 0 . . . 7)
Branch if Bit n is Clear	BRCLR n (n = 0 . . . 7)
Set Bit n	BSET n (n = 0 . . . 7)
Clear Bit n	BCLR n (n = 0 . . . 7)

OPCODE MAP SUMMARY

Table 9 is an opcode map for the instructions used on the MCU.

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single byte instructions; the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. One- or two-byte direct addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction. When using the Motorola assembler, the user need not specify whether an instruction uses direct or extended addressing. The assembler automatically selects the shortest form of the instruction.

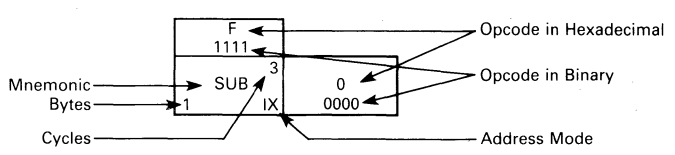
Table 9. Opcode Map

	Bit Manipulation		Branch	Read-Modify-Write					Control		Register/Memory						
	BTB	BSC	REL	DIR	INH	INH	IX1	IX	INH	INH	IMM	DIR	EXT	IX2	IX1	IX	
HI / LOW	0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111	8 1000	9 1001	A 1010	B 1011	C 1100	D 1101	E 1110	F 1111	HI / LOW
0 0000	BRSET0 5 3 BTB	BSET0 5 2 BSC	BRA 3 2 REL	NEG 5 2 DIR	NEGA 3 1 INH	NEGX 3 1 INH	NEG 6 2 IX1	NEG 5 1 IX	RTI 9 1 INH		SUB 2 2 IMM	SUB 3 2 DIR	SUB 4 3 EXT	SUB 5 3 IX2	SUB 4 2 IX1	SUB 3 1 IX	0 0000
1 0001	BRCLR0 5 3 BTB	BCLR0 5 2 BSC	BRN 3 2 REL						RTS 6 1 INH		CMP 2 2 IMM	CMP 3 2 DIR	CMP 4 3 EXT	CMP 5 3 IX2	CMP 4 2 IX1	CMP 3 1 IX	1 0001
2 0010	BRSET1 5 3 BTB	BSET1 5 2 BSC	BHI 3 2 REL		MUL 11 1 INH						SBC 2 2 IMM	SBC 3 2 DIR	SBC 4 3 EXT	SBC 5 3 IX2	SBC 4 2 IX1	SBC 3 1 IX	2 0010
3 0011	BRCLR1 5 3 BTB	BCLR1 5 2 BSC	BLS 3 2 REL	COM 5 2 DIR	COMA 3 1 INH	COMX 3 1 INH	COM 6 2 IX1	COM 5 1 IX	SWI 10 1 INH		CPX 2 2 IMM	CPX 3 2 DIR	CPX 4 3 EXT	CPX 5 3 IX2	CPX 4 2 IX1	CPX 3 1 IX	3 0011
4 0100	BRSET2 5 3 BTB	BSET2 5 2 BSC	BCC 3 2 REL	LSR 5 2 DIR	LSRA 3 1 INH	LSRX 3 1 INH	LSR 6 2 IX1	LSR 5 1 IX			AND 2 2 IMM	AND 3 2 DIR	AND 4 3 EXT	AND 5 3 IX2	AND 4 2 IX1	AND 3 1 IX	4 0100
5 0101	BRCLR2 5 3 BTB	BCLR2 5 2 BSC	BCS 3 2 REL								BIT 2 2 IMM	BIT 3 2 DIR	BIT 4 3 EXT	BIT 5 3 IX2	BIT 4 2 IX1	BIT 3 1 IX	5 0101
6 0110	BRSET3 5 3 BTB	BSET3 5 2 BSC	BNE 3 2 REL	ROR 5 2 DIR	RORA 3 1 INH	RORX 3 1 INH	ROR 6 2 IX1	ROR 5 1 IX			LDA 2 2 IMM	LDA 3 2 DIR	LDA 4 3 EXT	LDA 5 3 IX2	LDA 4 2 IX1	LDA 3 1 IX	6 0110
7 0111	BRCLR3 5 3 BTB	BCLR3 5 2 BSC	BEQ 3 2 REL	ASR 5 2 DIR	ASRA 3 1 INH	ASRX 3 1 INH	ASR 6 2 IX1	ASR 5 1 IX		TAX 2 1 INH		STA 4 2 DIR	STA 5 3 EXT	STA 6 3 IX2	STA 5 2 IX1	STA 4 1 IX	7 0111
8 1000	BRSET4 5 3 BTB	BSET4 5 2 BSC	BHCC 3 2 REL	LSL 5 2 DIR	LSLA 3 1 INH	LSLX 3 1 INH	LSL 6 2 IX1	LSL 5 1 IX		CLC 2 1 INH	EOR 2 2 IMM	EOR 3 2 DIR	EOR 4 3 EXT	EOR 5 3 IX2	EOR 4 2 IX1	EOR 3 1 IX	8 1000
9 1001	BRCLR4 5 3 BTB	BCLR4 5 2 BSC	BHCS 3 2 REL	ROL 5 2 DIR	ROLA 3 1 INH	ROLX 3 1 INH	ROL 6 2 IX1	ROL 5 1 IX		SEC 2 1 INH	ADC 2 2 IMM	ADC 3 2 DIR	ADC 4 3 EXT	ADC 5 3 IX2	ADC 4 2 IX1	ADC 3 1 IX	9 1001
A 1010	BRSET5 5 3 BTB	BSET5 5 2 BSC	BPL 3 2 REL	DEC 5 2 DIR	DECA 3 1 INH	DECX 3 1 INH	DEC 6 2 IX1	DEC 5 1 IX		CLI 2 1 INH	ORA 2 2 IMM	ORA 3 2 DIR	ORA 4 3 EXT	ORA 5 3 IX2	ORA 4 2 IX1	ORA 3 1 IX	A 1010
B 1011	BRCLR5 5 3 BTB	BCLR5 5 2 BSC	BMI 3 2 REL							SEI 2 1 INH	ADD 2 2 IMM	ADD 3 2 DIR	ADD 4 3 EXT	ADD 5 3 IX2	ADD 4 2 IX1	ADD 3 1 IX	B 1011
C 1100	BRSET6 5 3 BTB	BSET6 5 2 BSC	BMC 3 2 REL	INC 5 2 DIR	INCA 3 1 INH	INCX 3 1 INH	INC 6 2 IX1	INC 5 1 IX		RSP 2 1 INH		JMP 2 2 DIR	JMP 3 3 EXT	JMP 4 3 IX2	JMP 3 2 IX1	JMP 2 1 IX	C 1100
D 1101	BRCLR6 5 3 BTB	BCLR6 5 2 BSC	BMS 3 2 REL	TST 4 2 DIR	TSTA 3 1 INH	TSTX 3 1 INH	TST 5 2 IX1	TST 4 1 IX		NOP 2 1 INH	BSR 6 2 REL	JSR 5 2 DIR	JSR 6 3 EXT	JSR 7 3 IX2	JSR 6 2 IX1	JSR 5 1 IX	D 1101
E 1110	BRSET7 5 3 BTB	BSET7 5 2 BSC	BIL 3 2 REL						STOP 2 1 INH		LDX 2 2 IMM	LDX 3 2 DIR	LDX 4 3 EXT	LDX 5 3 IX2	LDX 4 2 IX1	LDX 3 1 IX	E 1110
F 1111	BRCLR7 5 3 BTB	BCLR7 5 2 BSC	BIH 3 2 REL	CLR 5 2 DIR	CLRA 3 1 INH	CLRX 3 1 INH	CLR 6 2 IX1	CLR 5 1 IX	WAIT 2 1 INH	TXA 2 1 INH		STX 4 2 DIR	STX 5 3 EXT	STX 6 3 IX2	STX 5 2 IX1	STX 4 1 IX	F 1111

Abbreviations for Address Modes

INH	Inherent	REL	Relative
A	Accumulator	BSC	Bit Set/Clear
X	Index Register	BTB	Bit Test and Branch
IMM	Immediate	IX	Indexed (No Offset)
DIR	Direct	IX1	Indexed, 1 Byte (8-Bit) Offset
EXT	Extended	IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −126 to +129 from the opcode address. The programmer need not calculate the offset when using the Motorola assembler, since it calculates the proper offset and checks to see that it is within the span of the branch.

INDEXED, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. This addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this two-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This address mode can be used in a manner similar to indexed, 8-bit offset except that this three-byte instruction allows tables to be anywhere in memory. As with direct and extended addressing, the Motorola assembler determines the shortest form of indexed addressing.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode, and the byte following the opcode specifies the direct addressing of the byte in which the specified bit is to be set or cleared. Any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single two-byte instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit that is to be tested and its condition (set or clear), is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS (Voltages referenced to V_{SS})

Rating	Symbol	Value	Unit
Supply Voltage	V_{DD}	−0.5 to +7.0	V
Input Voltage	V_{in}	V_{SS} −0.5 to V_{DD} +0.5	V
Self-Check Mode ($\overline{IRQ}$ Pin Only)	V_{in}	V_{SS} −0.5 to 2×V_{DD} +0.5	V
Current Drain Per Pin Excluding V_{DD} and V_{SS}	I	25	mA
Operating Temperature Range MC68HC05B6P, FN (Standard) MC68HC05B6CP, CFN (Extended) MC68HC05B6MP, MFN (Automotive)	T_A	T_L to T_H 0 to +70 −40 to +85 −40 to +125	°C
Storage Temperature Range	T_{stg}	−65 to +150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation, it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{DD}$. Reliability of operation is enhanced if unused inputs are connected to an appropriate logic voltage level (e.g., either V_{SS} or V_{DD}).

3

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic Plastic Leaded Chip Carrier (PLCC)	θ_{JA}	 40 50	°C/W

V_{DD} = 4.5 V

Pins	R1	R2	C
PA7–PA0, PB7–PB0, PC7–PC0, TCMP1 TCMP2	3.26 kΩ	2.38 kΩ	50 pF
TDO, SCLK, PLMA, PLMB	1.9 kΩ	2.26 kΩ	200 pF

V_{DD} = 3.0 V

Pins	R1	R2	C
PA7–PA0, PB7–PB0, PC7–PC0, TCMP1, TCMP2	10.91 kΩ	6.32 kΩ	50 pF
TDO, SCLK, PLMA, PLMB	6 kΩ	6 kΩ	200 pF

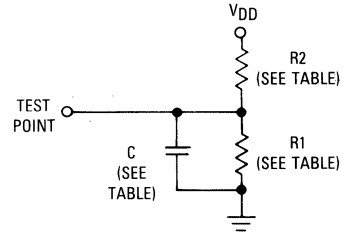


Figure 19. Equivalent Test Load

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{I/O}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
$P_{I/O}$ = Power Dissipation on Input and Output Pins — User Determined

For most applications $P_{I/O} < P_{INT}$ and can be neglected.

The following is an approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected):

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

DC ELECTRICAL CHARACTERISTICS (V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage, $I_{Load} \leq 10.0$ µA	V_{OL} V_{OH}	— $V_{DD} - 0.1$	— —	0.1 —	V
Output High Voltage (I_{Load} = 0.8 mA) PA7–PA0, PB7–PB0, PC7–PC0, TCMP1, TCMP2 (I_{Load} = 1.6 mA) TDO, SCLK, PLMA, PLMB	V_{OH}	 $V_{DD} - 0.8$ $V_{DD} - 0.8$	 $V_{DD} - 0.4$ $V_{DD} - 0.8$	 — —	V
Output Low Voltage (I_{Load} = 1.6 mA) PA7–PA0, PB7–PB0, PC7–PC0, TCMP1, TCMP2, PLMA, PLMB, TDO, SCLK $\overline{RESET}$	V_{OL}	 — —	 0.1 0.4	 0.4 1.0	V
Input High Voltage PA7–PA0, PB7–PB0, PC7–PC0, PD7–PD0, TCAP1, TCAP2, $\overline{IRQ}$, $\overline{RESET}$, OSC1, RDI	V_{IH}	$0.7 \times V_{DD}$	—	V_{DD}	V
Input Low Voltage PA7–PA0, PB7–PB0, PC7–PC0, PD7–PD0, TCAP1, TCAP2, $\overline{IRQ}$, $\overline{RESET}$, OSC1, RDI	V_{IL}	V_{SS}	—	$0.2 \times V_{DD}$	V
Supply Current (see Notes) RUN (SM = 0) RUN (SM = 1, t_{cyc} = 8 µs) WAIT (SM = 0) WAIT (SM = 1, t_{cyc} = 8 µs) STOP 0 to 70 (Standard) −40 to 85 (Extended) −40 to 125 (Automotive)	I_{DD}	 — — — — — — —	 3.5 0.5 1 0.35 2 — —	 9 2 4 1 10 20 50	 mA mA mA mA µA µA µA
I/O Ports Hi-Z Leakage Current PA7–PA0, PB7–PB0, PC7–PC0, TDO, $\overline{RESET}$, SCLK	I_{IL}	—	0.2	±1	µA
Input Current $\overline{IRQ}$, TCAP1, TCAP2, OSC1, RDI PD7/AN7–PD0/AN0 (A/D off) PD7/AN7–PD0/AN0 (A/D on)	I_{in}	 — — —	 ±0.2 ±0.2 ±10	 ±1 ±1 TBD	µA
Capacitance Ports (as Input or Output), $\overline{RESET}$ TDO, SCLK $\overline{IRQ}$, TCAP1, TCAP2, OSC1, RDI PD7/AN7–PD0/AN0 (A/D off) PD7/AN7–PD0/AN0 (A/D on)	 C_{out} C_{out} C_{in} C_{in} C_{in}	 — — — — —	 — — — 12 22	 12 12 8 TBD TBD	pF

NOTES:
1. All values shown reflect average measurements.
2. Typical values at midpoint of voltage range, 25°C only.
3. Wait I_{DD}: Only timer system active (TE = RE = 0). If SCI active (TE = RE = 1) add 10% current draw.
4. Run (Operating) I_{DD}, Wait I_{DD}: Measured using external square wave clock source (f_{osc} = 4.0 MHz), all inputs 0.2 V from rail; no dc loads, less than 50 pF on all outputs, C_L = 20 pF on OSC2.
5. Wait, Stop I_{DD}: All ports configured as inputs, V_{IL} = 0.2 V, $V_{IH} = V_{DD} - 0.2$ V.
6. Wait I_{DD} is affected linearly by the OSC2 capacitance.

TBD — To be determined.

DC ELECTRICAL CHARACTERISTICS (V_{DD} = 3.3 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage, $I_{Load} \leq 10.0\ \mu A$	V_{OL} V_{OH}	— $V_{DD}-0.1$	— —	0.1 —	V
Output High Voltage (I_{Load} = 0.2 mA) PA7–PA0, PB7–PB0, PC7–PC0, TCMP1, TCMP2 (I_{Load} = 0.4 mA) TDO, SCLK, PLMA, PLMB	V_{OH}	 $V_{DD}-0.3$ $V_{DD}-0.3$	 $V_{DD}-0.1$ $V_{DD}-0.1$	 — —	V
Output Low Voltage (I_{Load} = 0.4 mA) PA7–PA0, PB7–PB0, PC7–PC0, TCMP1, TCMP2, PLMA, PLMB, TDO, SCLK $\overline{RESET}$	V_{OL}	 — —	 0.1 0.2	 0.3 0.6	V
Input High Voltage PA7–PA0, PB7–PB0, PC7–PC0, PD7–PD0, TCAP1, TCAP2, $\overline{IRQ}$, $\overline{RESET}$, OSC1, RDI	V_{IH}	$0.7 \times V_{DD}$	—	V_{DD}	V
Input Low Voltage PA7–PA0, PB7–PB0, PC7–PC0, PD7–PD0, TCAP1, TCAP2, $\overline{IRQ}$, $\overline{RESET}$, OSC1, RDI	V_{IL}	V_{SS}	—	$0.2 \times V_{DD}$	V
Supply Current (see Notes) RUN (SM = 0) RUN (SM = 1, t_{cyc} = 8 µs) WAIT (SM = 0) WAIT (SM = 1, t_{cyc} = 8 µs) STOP 0 to 70 (Standard) –40 to 85 (Extended) –40 to 125 (Automotive)	I_{DD}	 — — — — — — —	 1.2 0.2 0.4 0.15 1 — —	 5 1 2 0.5 10 10 30	 mA mA mA mA µA µA µA
I/O Ports Hi-Z Leakage Current PA7–PA0, PB7–PB0, PC7–PC0, TDO, $\overline{RESET}$, SCLK	I_{IL}	—	±0.2	±10	µA
Input Current $\overline{IRQ}$, TCAP1, TCAP2, OSC1, RDI PD7/AN7–PD0/AN0 (A/D off) PD7/AN7–PD0/AN0 (A/D on)	I_{in}	 — — —	 ±0.2 ±0.2 ±10	 ±1 ±1 TBD	µA
Capacitance Ports (as Input or Output), $\overline{RESET}$, TDO TDO, SCLK $\overline{IRQ}$, TCAP1, TCAP2, OSC1, RDI PD7/AN7–PD0/AN0 (A/D off) PD7/AN7–PD0/AN0 (A/D on)	 C_{out} C_{out} C_{in} C_{in} C_{in}	 — — — — —	 — — — 12 22	 12 12 8 TBD TBD	pF

NOTES:
1. All values shown reflect average measurements.
2. Typical values at midpoint of voltage range, 25°C only.
3. Wait I_{DD}: Only timer system active (TE = RE = 0). If SCI active (TE = RE = 1) add 10% current draw.
4. Run (Operating) I_{DD}, Wait I_{DD}: Measured using external square wave clock source (f_{osc} = 4.0 MHz), all inputs 0.2 V from rail; no dc loads, less than 50 pF on all outputs, C_L = 20 pF on OSC2.
5. Wait, Stop I_{DD}: All ports configured as inputs, V_{IL} = 0.2 V, $V_{IH} = V_{DD} - 0.2$ V.
6. Wait I_{DD} is affected linearly by the OSC2 capacitance.

TBD — To be determined.

A/D CONVERTER CHARACTERISTICS (V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc)

Characteristic	Parameter	Min	Max	Unit
Resolution	Number of bits resolved by the A/D	8	—	Bit
Non-Linearity	Maximum deviation from the best straight line through the A/D transfer characteristics ($V_{RH} = V_{DD}$ and V_{RL} = 0 V)	—	± ½	LSB
Quantization Error	Uncertainty due to converter resolution	—	± ½	LSB
Absolute Accuracy	Difference between the actual input voltage and the full-scale equivalent of the binary code output code for all errors	—	± 1	LSB
Conversion Range	Analog input voltage range	V_{RL}	V_{RH}	V
V_{RH}	Maximum analog reference voltage	V_{RL}	V_{DD} + 0.1	V
V_{RL}	Minimum analog reference voltage	V_{SS} − 0.1	V_{RH}	V
Conversion Time	Total time to perform a single analog to digital conversion a. External Clock (XTAL, EXTAL) b. Internal RC oscillator	 — —	 32 32	 t_{cyc} µs
Monotonicity	Conversion result never decreases with an increase in input voltage and has no missing codes	Guaranteed		
Zero-Input Reading	Conversion result when $V_{in} = V_{RL}$	00	—	Hex
Full-Scale Reading	Conversion result when $V_{in} = V_{RH}$	—	FF	Hex
Sample Acquisition Time (see Note 1)	Analog input acquisition sampling a. External Clock (XTAL, EXTAL) b. Internal RC oscillator	 — —	 12 12	 t_{cyc} µs
Sample/Hold Capacitance	Input capacitance on PD7/AN7–PD0/AN0	—	12	pF
Input Leakage (see Note 2)	Input leakage on A/D pins PD7/AN7–PD0/AN0, V_{RL}, V_{RH}	— —	1 1	µA

NOTES:
1. Source impedances greater than 10K ohm will adversely affect internal RC charging time during input sampling.
2. The external system error caused by input leakage current is approximately equal to the product of R source and input current.

3

CONTROL TIMING (V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation Crystal Option External Clock Option	f_{osc}	 — dc	 4.2 4.2	MHz
Internal Operating Frequency Crystal (f_{osc}/2) External Clock (f_{osc}/2)	f_{op}	 — dc	 2.1 2.1	MHz
Cycle Time (see Figure 21)	t_{cyc}	480	—	ns
Crystal Oscillator Startup Time (see Figure 21)	t_{OXOV}	—	100	ms
Stop Recovery Startup Time (Crystal Oscillator)	t_{ILCH}	—	100	ms
External RESET Input Pulse Width (see Figure 21)	t_{RL}	1.5	—	t_{cyc}
Power-On RESET Output Pulse Width 4064 Cycle Option 16 Cycle Option	t_{PORL}	 4064 16	 — —	t_{cyc}
Watchdog RESET Output Pulse Width	t_{DOGL}	1.5	—	t_{cyc}
Watchdog Time-Out	t_{DOG}	6144	7168	t_{cyc}
EEPROM Byte Erase Time 0 to 70 (Standard) −40 to 85 (Extended) −40 to 125 (Automotive)	t_{ERA}	 10 10 10	 — — —	ms
EEPROM Byte Programming Time 0 to 70 (Standard) −40 to 85 (Extended) −40 to 125 (Automotive)	t_{PROG}	 10 10 20	 — — —	ms
Timer Resolution** Input Capture Pulse Width (see Figure 20) Input Capture Pulse Period (see Figure 20)	 t_{RESL} t_{TH}, t_{TL} t_{TL}, t_{TL}	 4.0 125 ***	 — — —	 t_{cyc} ns t_{cyc}
Interrupt Pulse Width (Edge-Triggered)	t_{ILIH}	125	—	ns
Interrupt Pulse Period	t_{ILIL}	*	—	t_{cyc}
OSC1 Pulse Width	t_{OH}, t_{OL}	90	—	ns

NOTES:

*The minimum period t_{ILIL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus 21 t_{cyc}.

**Since a 2-bit prescaler in the timer must count four internal cycles (t_{cyc}), this is the limiting minimum factor in determining the timer resolution.

***The minimum period t_{TLTL} should not be less than the number of cycle times it takes to execute the capture interrupt service routine plus 24 t_{cyc}.

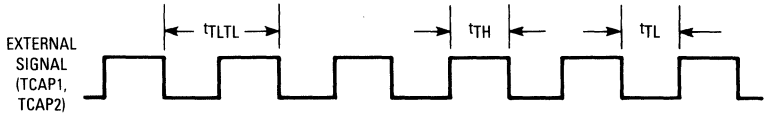


Figure 20. Timer Relationship

CONTROL TIMING (V_{DD} = 3.3 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation Crystal Option External Clock Option	f_{osc}	 — dc	 2.0 2.0	MHz
Internal Operating Frequency Crystal (f_{osc}/2) External Clock (f_{osc}/2)	f_{op}	 — dc	 1.0 1.0	MHz
Cycle Time (see Figure 21)	t_{cyc}	1000	—	ns
Crystal Oscillator Startup Time (see Figure 21)	t_{OXOV}	—	100	ms
Stop Recovery Startup Time (Crystal Oscillator)	t_{ILCH}	—	100	ms
External RESET Input Pulse Width (see Figure 21)	t_{RL}	1.5	—	t_{cyc}
Power-On RESET Output Pulse Width 4064 Cycle Option 16 Cycle Option	t_{PORL}	 4064 16	 — —	t_{cyc}
Watchdog RESET Output Pulse Width	t_{DOGL}	1.5	—	t_{cyc}
Watchdog Time-Out	t_{DOG}	6144	7168	t_{cyc}
EEPROM Byte Erase Time 0 to 70 (Standard) −40 to 85 (Extended) −40 to 125 (Automotive)	t_{ERA}	 30 TBD TBD	 — — —	ms
EEPROM Byte Programming Time 0 to 70 (Standard) −40 to 85 (Extended) −40 to 125 (Automotive)	t_{PROG}	 30 TBD TBD	 — — —	ms
Timer Resolution** Input Capture Pulse Width (see Figure 20) Input Capture Pulse Period (see Figure 20)	 t_{RESL} t_{TH}, t_{TL} t_{TL}, t_{TL}	 4.0 250 ***	 — — —	 t_{cyc} ns t_{cyc}
Interrupt Pulse Width (Edge-Triggered)	t_{ILIH}	250	—	ns
Interrupt Pulse Period	t_{ILIL}	*	—	t_{cyc}
OSC1 Pulse Width	t_{OH}, t_{OL}	200	—	ns

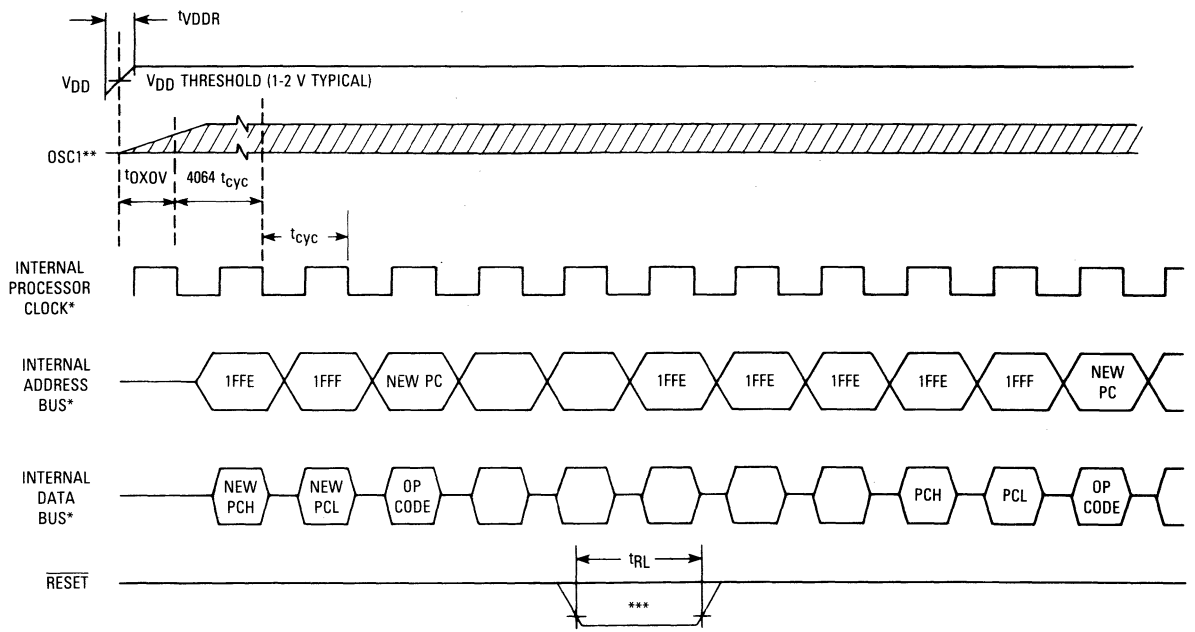
NOTES:

*The minimum period t_{ILIL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus 21 t_{cyc}.

**Since a 2-bit prescaler in the timer must count four internal cycles (t_{cyc}), this is the limiting minimum factor in determining the timer resolution.

***The minimum period t_{TLTL} should not be less than the number of cycle times it takes to execute the capture interrupt service routine plus 24 t_{cyc}.

3



*Internal timing signal and bus information not available externally.
**OSC1 line is not meant to represent frequency. It is only used to represent time.
***The next rising edge of the internal processor clock following the rising edge of $\overline{\text{RESET}}$ initiates the reset sequence.

Figure 21. Power-On Reset and $\overline{\text{RESET}}$

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola in the following media:

MS™-DOS/PC-DOS disk file (360K)
EPROM(s) 2764, MCM68764, MCM68766, or EEPROM MC68HC805C4

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, a sales person, or a Motorola representative.

FLEXIBLE DISKS

A flexible disk (MS-DOS/PC-DOS disk file), programmed with the customer's program may be submitted for pattern generation. The diskette should be clearly labeled with the customer's name, data, project or product name, and the name of the file containing the pattern.

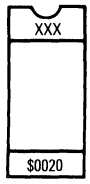
MS-DOS is Microsoft's Disk Operating System. PC-DOS is the IBM® Personal Computer (PC) Disk Operating System. Disk media submitted must be a standard density (360K) double-sided 5 1/4 inch compatible floppy diskette. The diskette must contain object file code in Motorola's S-record format. The S-record format is a character-based object file format generated by M6805 cross assemblers and linkers on IBM PC style machines.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to expedite the process in case of any difficulty with the pattern file.

EPROMs

A 2764, 68764, or 68766 type EPROM, programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. Since all program and data space information will fit on one 2764, 68764, or 68766 EPROM device, the EPROM must be programmed as described in the following paragraphs.

For an MC68HC805B6 MCU start the page zero, user ROM at EEPROM address $0020 through $004F. Start the user ROM at EEPROM address $0800 through $1EFF with vectors from $1FF0 to $1FFF. All unused bytes, including the user's space, must be set to zero. For shipment to Motorola, EPROMs should be placed in a conductive IC carrier and packed securely. Styrofoam is not acceptable for shipment.



xxx = Customer ID

Verification Media

All original pattern media (EPROMs or floppy disks) are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked, and the verification form should be completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for the creation of the customer mask. To aid in the verification process, Motorola will program *customer supplied* blank EPROM(s) or DOS disks from the data file used to create the custom mask.

ROM VERIFICATION UNITS (RVUs)

Ten MCUs containing the customer's ROM pattern will be sent for program verification. These units will have been made using the custom mask, but are for the purpose of ROM verification only. For expediency, the MCUs are unmarked, packaged in ceramic, and tested with five volts at room temperature. These RVUs are free with the minimum order quantity, but are not production parts. RVUs are not backed or guaranteed by Motorola Quality Assurance.

ORDERING INFORMATION

The following table provides ordering information pertaining to the package type, temperature, and MC order numbers for the MC68HC05B6 device.

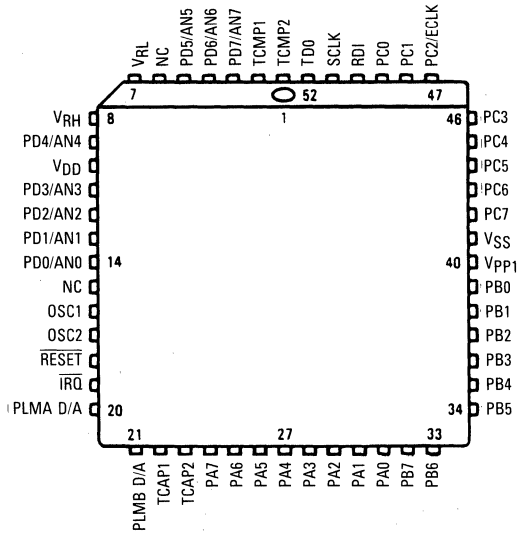
Package Type	Temperature	MC Order Number
Plastic (P Suffix)	0°C to +70°C	MC68HC05B6P
	−40°C to +85°C	MC68HC05B6CP
	−40°C to +125°C	MC68HC05B6MP
PLCC (FN Suffix)	0°C to +70°C	MC68HC05B6FN
	−40°C to +85°C	MC68HC05B6CFN
	−40°C to +125°C	MC68HC05B6MFN

3

MS is a trademark of Microsoft, Inc.
IBM is a registered trademark of International Business Machines Corporation.

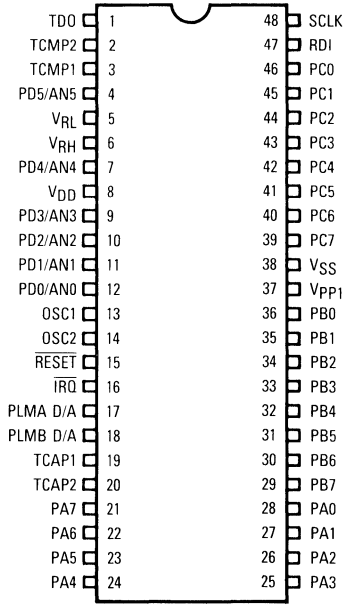
PIN ASSIGNMENTS

52-Pin PLCC



3

48-Pin Dual-in-Line Package



MC68HC05C2

Technical Summary
8-Bit Microcontroller Unit

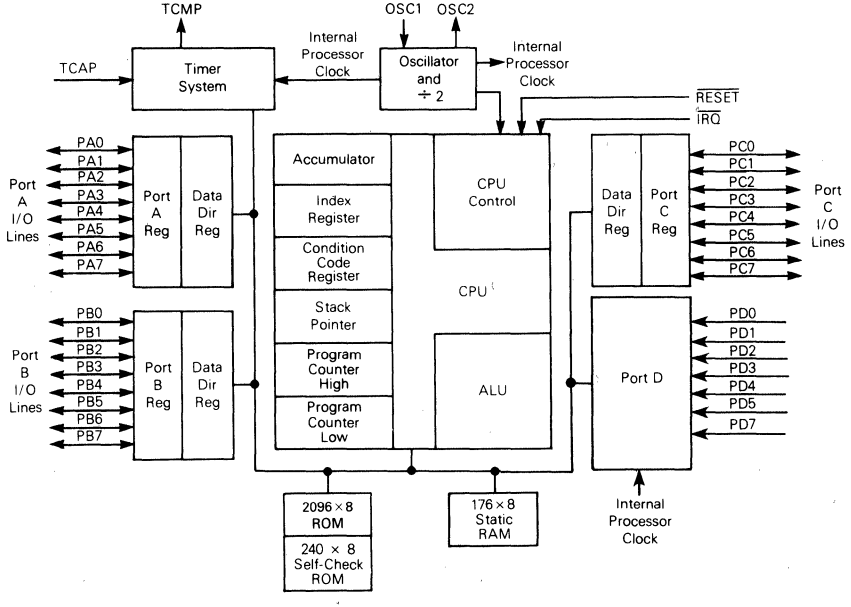
The MC68HC05C2 (HCMOS) microcontroller unit (MCU) is a member of the M68HC05 Family of microcontrollers. This high-performance, low-power MCU has parallel I/O capability with pins programmable as input or output. This publication contains condensed information on the MCU; for more detailed information, contact your local Motorola sales office.

The following block diagram depicts the hardware features; additional features available on the MCU are as follows:

- On-Chip Oscillator with RC or Crystal/Ceramic Resonator Mask Options
- Memory-Mapped I/O
- 176 Bytes of On-Chip RAM
- 2096 Bytes of User ROM
- 24 Bidirectional I/O Lines and 7 Input-Only Lines
- Self-Check Mode
- Power-Saving STOP, WAIT, and Data Retention Modes
- Single 3.0- to 5.5-Volt Supply (2-Volt Data Retention Mode)
- Fully Static Operation
- 8×8 Unsigned Multiply Instruction

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

The signal descriptions of the MCU are discussed in the following paragraphs.

V_{DD} AND V_{SS}

Power is supplied to the microcontroller using these two pins. V_{DD} is the positive supply, and V_{SS} is ground.

$\overline{IRQ}$

This pin is a programmable option that provides two different choices of interrupt triggering sensitivity. Refer to **INTERRUPTS** for more detail.

OSC1, OSC2

These pins provide control input for an on-chip clock oscillator circuit. A crystal, a ceramic resonator, a resistor/capacitor combination, or an external signal connects to these pins providing a system clock. A mask option selects either a crystal/ceramic resonator or a resistor/capacitor as the frequency determining element. The oscillator frequency is two times the internal bus rate.

RC Oscillator

With this option, a resistor is connected to the oscillator pins as shown in Figure 1(d). The relationship between R and f_{osc} is shown in Figure 2.

Crystal

The circuit shown in Figure 1(b) is recommended when using a crystal. Using an external CMOS oscillator is recommended when crystals outside the specified ranges are to be used. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and start-up stabilization time. Refer to **ELECTRICAL SPECIFICATIONS** for V_{DD} specifications.

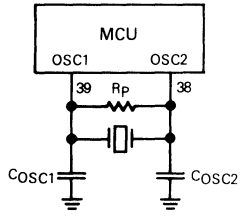
Crystal

	2 MHz	4 MHz	Units
R_{SMAX}	400	75	Ω
C_0	5	7	pF
C_1	0.008	0.012	μF
C_{OSC1}	15-40	15-30	pF
C_{OSC2}	15-30	15-25	pF
R_P	10	10	MΩ
Q	30	40	K

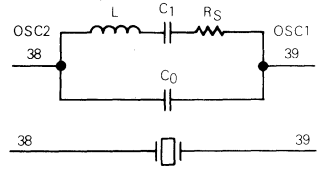
Ceramic Resonator

	2-4 MHz	Units
R_S (typical)	10	Ω
C_O	40	pF
C_1	4.3	pF
C_{OSC1}	30	pF
C_{OSC2}	30	pF
R_P	1-10	MΩ
Q	1250	—

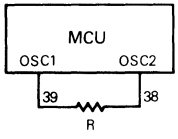
(a) Crystal/Ceramic Resonator Parameters



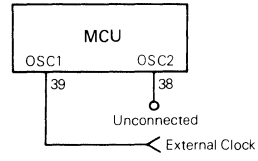
(b) Crystal/Ceramic Resonator Oscillator Connections



(c) Equivalent Crystal Circuit



(d) RC Oscillator Connections



(e) External Clock Source Connections (For Crystal Mask Option Only)

Figure 1. Oscillator Connections

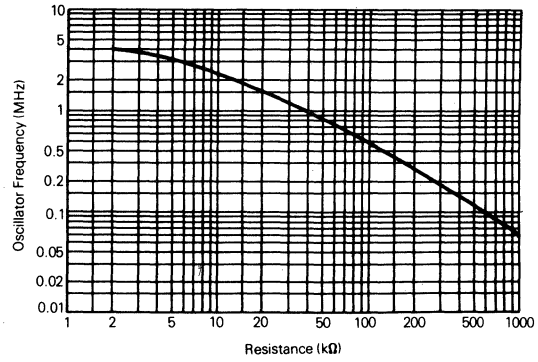


Figure 2. Typical Frequency vs Resistance for RC Oscillator Option Only

3

Ceramic Resonator

A ceramic resonator may be used in place of the crystal in cost-sensitive applications. The circuit in Figure 1(b) is recommended when using a ceramic resonator. Figure 1(a) lists the recommended capacitance and resistance values. The manufacturer of the resonator considered should be consulted for specific information on resonator operation.

External Clock

An external clock should be applied to the OSC1 input with the OSC2 input not connected, as shown in Figure 1(e). This option may only be used with the crystal oscillator mask option.

INPUT CAPTURE (TCAP)

This pin controls the input capture feature for the on-chip programmable timer.

OUTPUT COMPARE (TCMP)

This pin provides an output for the output compare feature of the on-chip timer.

$\overline{\text{RESET}}$

This pin is used to reset the MCU and provide an orderly start-up procedure by pulling $\overline{\text{RESET}}$ low.

INPUT/OUTPUT PORTS (PA0-PA7, PB0-PB7, PC0-PC7)

These 24 lines are arranged into three 8-bit ports (A, B, and C). These ports are programmable as either inputs or outputs under software control of the data direction registers. Refer to **PROGRAMMING** for additional information.

FIXED INPUT PORT (PD0-PD5, PD7)

These seven lines comprise port D, a fixed input port. Refer to **PROGRAMMING** for additional information.

PROGRAMMING

Input/output port programming and fixed input port programming are discussed in the following paragraphs.

INPUT/OUTPUT PORT PROGRAMMING

Any port pin is programmable as either an input or an output under software control of the corresponding data direction register (DDR). Each port bit can be selected as output or input by writing the corresponding bit in the port DDR to a logic one for output and logic zero for input. On reset, all DDRs are initialized to logic zero to put the ports in the input mode. The port output registers are not initialized on reset but may be written to before setting the DDR bits to avoid undefined levels.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This port write may be used to initialize the data registers and avoid undefined outputs. Refer to Figure 3 for typical port circuitry and to Table 1 for a list of the I/O pin functions.

Table 1. I/O Pin Functions

R/$\overline{\text{W}}$*	DDR	I/O Pin Functions
0	0	The I/O pin is in input mode. Data is written into the output data latch.
0	1	Data is written into the output data latch and output to the I/O pin.
1	0	The state of the I/O pin is read.
1	1	The I/O pin is in an output mode. The output data latch is read.

*R/$\overline{\text{W}}$ is an internal signal.

FIXED INPUT PORT PROGRAMMING

Port D is a fixed input port (PD0-PD5, PD7) that monitors the external pins. To avoid spurious interrupts and erratic operation of port D, memory accesses to unused locations $000A through $0011 must not be performed.

NOTE

Any unused inputs and I/O ports should be tied to an appropriate logic level (e.g., either V_{DD} or V_{SS}).

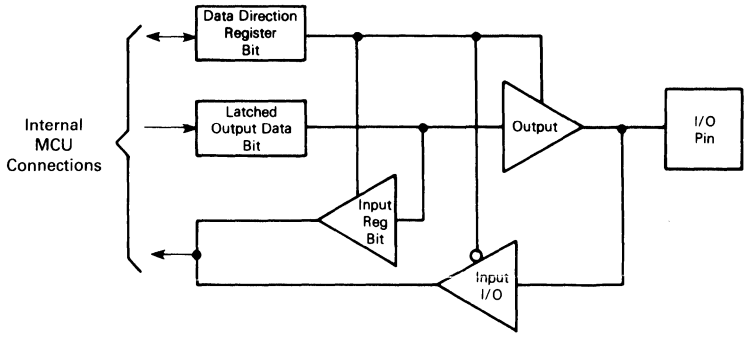


Figure 3. Typical Port I/O Circuit

MEMORY

The MCU is capable of addressing 8192 bytes of memory and I/O registers, as shown in Figure 4. The locations consist of user ROM, user RAM, self-check ROM, control registers, and I/O. The user-defined reset and interrupt vectors are located from $1FF4 to $1FFF.

The shared stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer decrements during pushes and increments during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

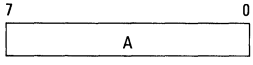
Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

REGISTERS

The MCU contains the registers described in the following paragraphs.

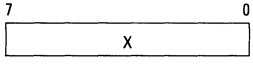
ACCUMULATOR (A)

The accumulator is a general-purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



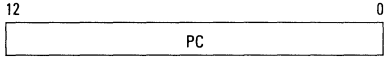
INDEX REGISTER (X)

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16-bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



PROGRAM COUNTER (PC)

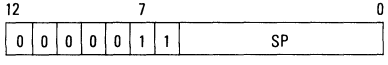
The program counter is a 13-bit register that contains the address of the next byte to be fetched.



STACK POINTER (SP)

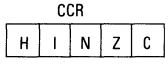
The stack pointer is a 13-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set to location $00FF. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

When accessing memory, the seven most significant bits are permanently set to 0000011. These seven bits are appended to the six least significant register bits to produce an address within the range of $00FF to $00C0. Subroutines and interrupts may use up to 64 (decimal) locations. If 64 locations are exceeded, the stack pointer wraps around and loses the previously stored information. A subroutine call occupies two locations on the stack; an interrupt uses five locations.



CONDITION CODE REGISTER (CCR)

The CCR is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program, and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.



Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

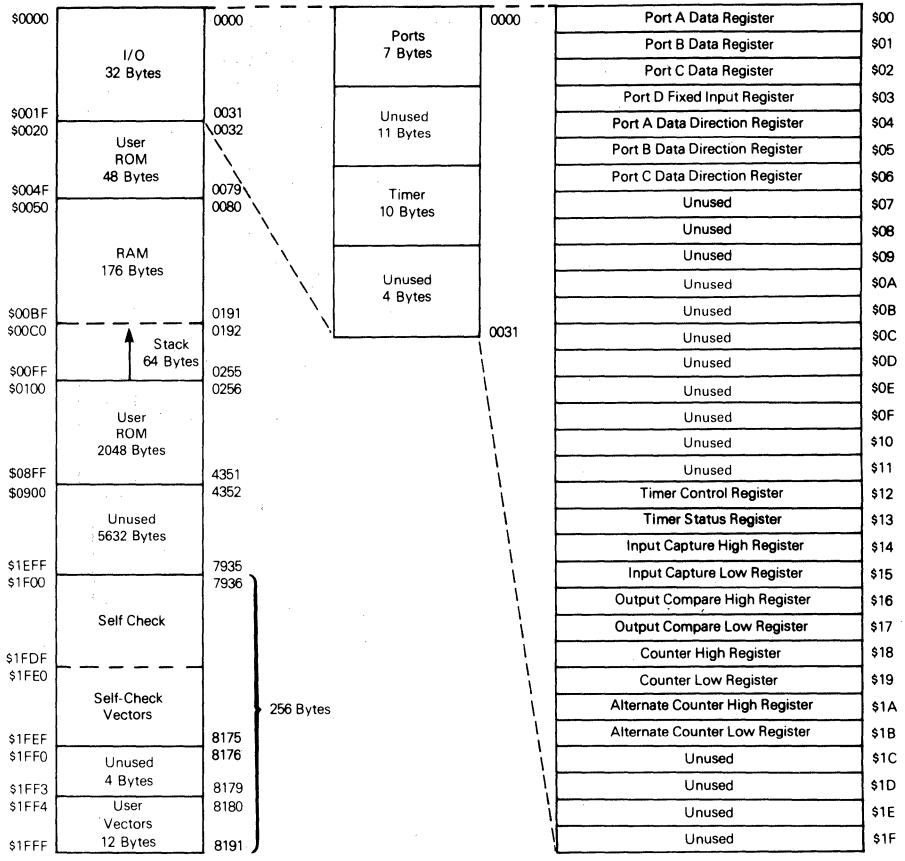


Figure 4. Memory Map

Interrupt (I)

When this bit is set, the timer and external interrupt is masked (disabled). If an interrupt occurs while this bit is set, the interrupt is latched and processed as soon as the interrupt bit is cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic one).

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/Borrow (C)

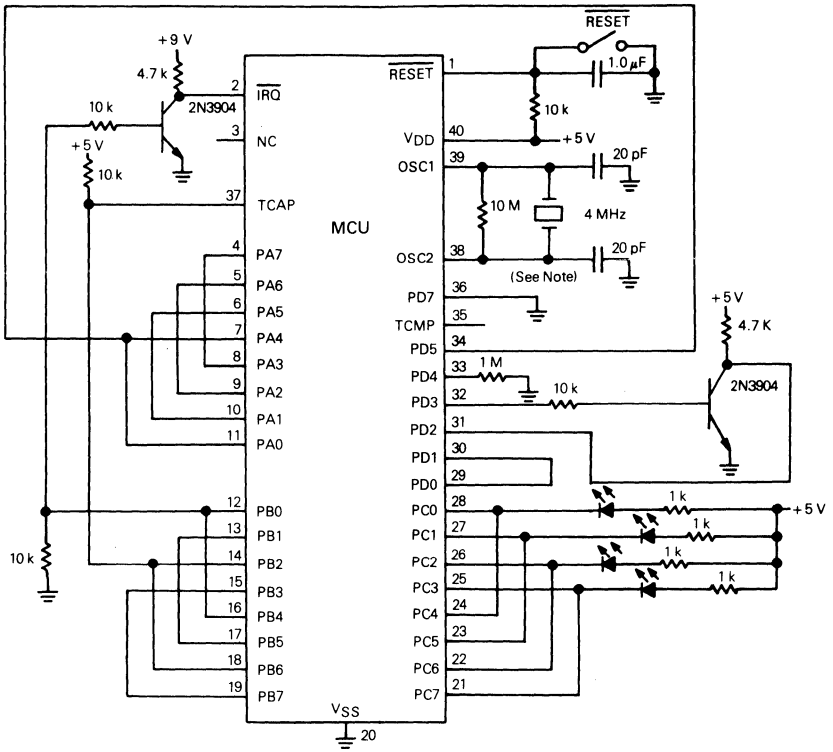
When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions and during shifts and rotates.

SELF-CHECK

The self-check capability provides the ability to determine if the device is functional. Self-check is performed using the circuit shown in Figure 5. Port C pins PC0-PC3 are monitored for the self-check results. After reset, the following seven tests are performed automatically:

I/O — Exercise of ports A, B, and C
RAM — Counter test for each RAM byte
ROM — Exclusive OR with odd ones parity result
Timer — Tracks counter register and checks OCF flag
Interrupts — Tests external, and timer interrupts

Self-check results (using the LEDs as monitors) are shown in Table 2. The following subroutines are available to the user and do not require any external hardware.



NOTE: The RC Oscillator Option may also be used in this circuit.

Figure 5. Self-Check Circuit Schematic Diagram

Table 2. Self-Check Results

PC3	PC2	PC1	PC0	Remarks
1	0	0	1	Bad I/O
1	0	1	0	Bad RAM
1	0	1	1	Bad Timer
1	1	0	1	Bad ROM
1	1	1	1	Bad Interrupts or $\overline{IRQ}$ Request
Flashing				Good Device
All Others				Bad Device, Bad Port C, etc.

0 indicates LED is on; 1 indicates LED is off.

TIMER TEST SUBROUTINE

This subroutine returns with the Z bit cleared if any error is detected; otherwise, the Z bit is set. The timer test subroutine is called at location $1FOE. The output compare register is first set to the current timer state. Because the timer is free running and has only a divide-by-four prescaler, each timer count cannot be tested. The test reads the timer once every 10 counts (40 cycles) and checks for correct counting. The test tracks the counter until the timer wraps around, triggering the output compare flag in the timer status register. RAM locations $0050 and $0051 are overwritten. Upon return to the user's program, X = 40. If the test passed, A = 0.

ROM CHECKSUM SUBROUTINE

This subroutine returns with the Z bit cleared if any error is detected; otherwise, the Z bit is set. The ROM checksum subroutine is called at location $1F93 with RAM location $0053 equal to $01 and A = 0. A short routine is set up and executed in RAM to compute a checksum of the entire ROM pattern. RAM locations $0050 through $0053 are overwritten. Upon return to the user's program, X = 0. If the test passed, A = 0.

RESETS

The MCU can be reset two ways: by initial power-up and by the external reset input ($\overline{RESET}$). The $\overline{RESET}$ input consists mainly of a Schmitt trigger that senses the $\overline{RESET}$ line logic level.

POWER-ON RESET (POR)

An internal reset is generated on power-up to allow the internal clock generator to stabilize. The power-on reset is strictly for power turn-on conditions and should not be used to detect a drop in the power supply voltage. There is a 4064 internal processor clock cycle (t_{CYC}) delay after the oscillator becomes active. If the $\overline{RESET}$ pin is low at the end of 4046 t_{CYC}, the MCU will remain in the reset condition until $\overline{RESET}$ goes high.

EXTERNAL RESET INPUT

The MCU is reset when a logic zero is applied to the $\overline{RESET}$ input for a period of one and one-half machine cycles (t_{CYC}).

INTERRUPTS

The MCU can be interrupted three different ways: the two maskable hardware interrupts ($\overline{IRQ}$ and timer) and the nonmaskable software interrupt instruction (SWI).

Interrupts cause the processor to save register contents on the stack and to set the interrupt mask (I bit) to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack and normal processing to resume. The stacking order is shown in Figure 6.

Unlike $\overline{RESET}$, hardware interrupts do not cause the current instruction execution to be halted but are considered pending until the current instruction is complete.

NOTE

The current instruction is the one already fetched and being operated on.

When the current instruction is complete, the processor checks all pending hardware interrupts. If unmasked (I bit clear) and if the corresponding interrupt enable bit is set, the processor proceeds with interrupt processing; otherwise, the next instruction is fetched and executed.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction, regardless of the I-bit state. Refer to Figure 7 for the reset and interrupt instruction processing sequence.

TIMER INTERRUPT

There are three different timer interrupt flags that cause a timer interrupt whenever they are set and enabled. The interrupt flags are in the timer status register (TSR), and the enable bits are in the timer control register (TCR). Refer to **TIMER** for more information.

EXTERNAL INTERRUPT

If the interrupt mask bit (I bit) of the CCR is set, all interrupts are disabled. Clearing the I bit enables the external interrupt. The external interrupt is internally synchronized and then latched on the falling edge of $\overline{IRQ}$. The action of the external interrupt is identical to the timer interrupt with the exception that the interrupt request input at $\overline{IRQ}$ is latched internally and the service routine address is specified by the contents of $1FFA and $1FFB.

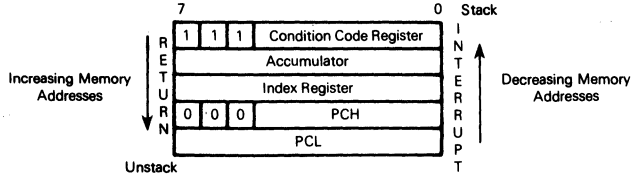
Either a level-sensitive and edge-sensitive trigger, or an edge-sensitive-only trigger are available as a mask option. Figure 8 shows both a functional internal diagram and a mode timing diagram for the interrupt line. The timing diagram shows two treatments of the interrupt line to the processor. The first method shows a single pulse on the interrupt line spaced far enough apart to be serviced. The minimum time between pulses is a function of the length of the interrupt service. Once a pulse occurs, the next pulse should not occur until an RTI occurs. This time (t_{ILIL}) is obtained by adding 21 instruction cycles to the total number of cycles it takes to complete the service routine (not including the RTI instruction). The second method shows many interrupt lines "wire-ORed" to form the interrupts at the processor. If the interrupt line remains low after servicing an interrupt, then the next interrupt is recognized.

NOTE

The internal interrupt latch is cleared in the first part of the interrupt service routine; therefore, one external interrupt pulse could be latched and serviced as soon as the I bit is cleared.

SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. If the I bit is zero, SWI executes after the other interrupts. The SWI operation is similar to the hardware interrupts. The interrupt service routine address is specified by the contents of memory locations $1FFC and $1FFD.



NOTE: Since the Stack Pointer decrements during pushes, the PCL is stacked first, followed by PCH, etc. Pulling from the stack is in the reverse order.

Figure 6. Interrupt Stacking Order

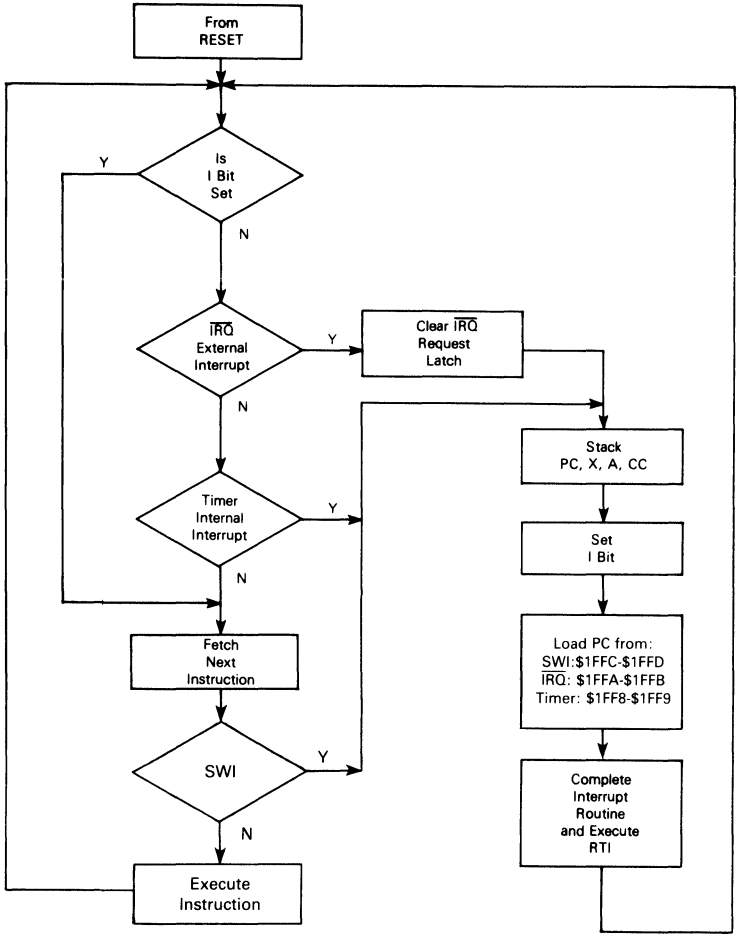
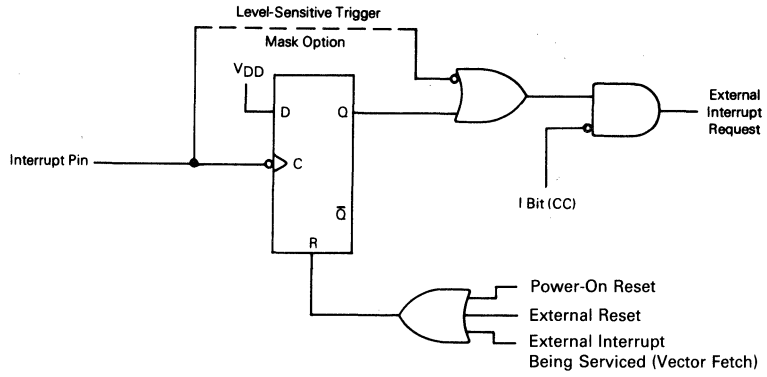
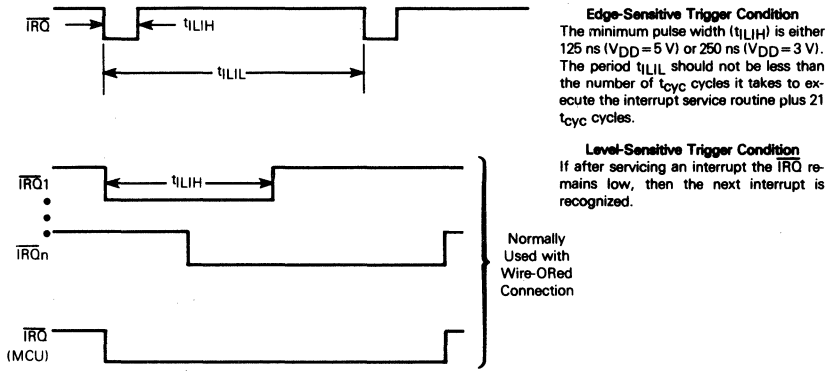


Figure 7. Reset and Interrupt Processing Flowchart

3



(a) Interrupt Internal Function Diagram



(b) Interrupt Mode Diagram

Figure 8. External Interrupt

LOW-POWER MODES

STOP

The STOP instruction places the MCU in its lowest power consumption mode. In the STOP mode, the internal oscillator is turned off, halting all internal processing including timer operation (refer to Figure 9).

During the STOP mode, the TCR bits are altered to remove any pending timer interrupt request and to disable any further timer interrupts. The timer prescaler is cleared. The I bit in the CCR is cleared to enable external interrupts. All other registers and memory remain unaltered. All input/output lines remain unchanged. The processor can be brought out of the STOP mode only by an external interrupt or reset.

WAIT

The WAIT instruction places the MCU in a low-power consumption mode, but the WAIT mode consumes more power than the STOP mode. All CPU action is suspended, but the timer remains active (refer to Figure 10). An interrupt from the timer can cause the MCU to exit the WAIT mode.

During the WAIT mode, the I bit in the CCR is cleared to enable interrupts. All other registers, memory, and input/output lines remain in their previous state. The timer may be enabled to allow a periodic exit from the WAIT mode.

DATA RETENTION MODE

The contents of RAM and CPU registers are retained at supply voltages as low as 2.0 Vdc. This is called the

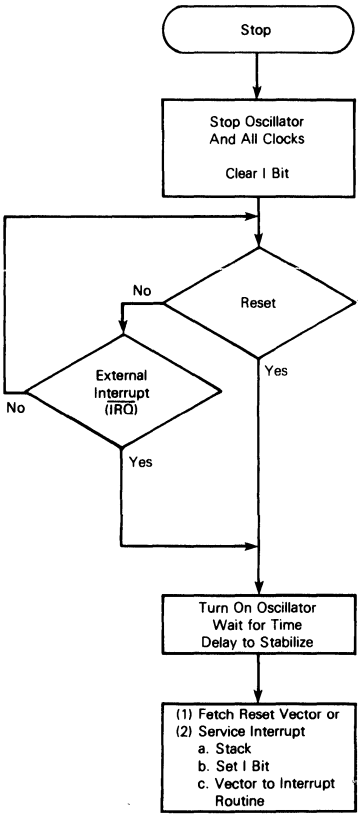


Figure 9. STOP Function Flowchart

data retention mode where the data is held, but the device is not guaranteed to operate. The MCU should be in $\overline{\text{RESET}}$ during data retention mode.

TIMER

The timer consists of a 16-bit, software-programmable counter driven by a fixed divide-by-four prescaler. This timer can be used for many purposes, including input waveform measurements while simultaneously generating an output waveform. Pulse widths can vary from several microseconds to many seconds. Refer to Figure 11 for a timer block diagram.

Because the timer has a 16-bit architecture, each specific functional segment (capability) is represented by two registers. These registers contain the high and low byte of that functional segment. Generally, accessing the low byte of a specific timer function allows full control of that function; however, an access of the high byte inhibits that specific timer function until the low byte is also accessed.

NOTE

The I bit in the CCR should be set while manipulating both the high and low byte register of a specific timer function to ensure that an interrupt does not occur.

COUNTER

The key element in the programmable timer is a 16-bit, free-running counter or counter register, preceded by a prescaler that divides the internal processor clock by four. The prescaler gives the timer a resolution of 2.0 microseconds if the internal bus clock is 2.0 MHz. The counter is incremented during the low portion of the internal bus clock. Software can read the counter at any time without affecting its value.

The double-byte, free-running counter can be read from either of two locations, $18–$19 (counter register) or $1A–$1B (counter alternate register). A read from only the least significant byte (LSB) of the free-running counter ($19, $1B) receives the count value at the time of the read. If a read of the free-running counter or counter alternate register first addresses the most significant byte (MSB) ($18, $1A), the LSB ($19, $1B) is transferred to a buffer. This buffer value remains fixed after the first MSB read, even if the user reads the MSB several times. This buffer is accessed when reading the free-running counter or counter alternate register LSB ($19 or $1B) and, thus, completes a read sequence of the total counter value. In reading either the free-running counter or counter alternate register, if the MSB is read, the LSB must also be read to complete the sequence.

The counter alternate register differs from the counter register in one respect: a read of the counter register MSB can clear the timer overflow flag (TOF). Therefore, the counter alternate register can be read at any time without the possibility of missing timer overflow interrupts due to clearing of the TOF.

The free-running counter is configured to $FFFC during reset and is always a read-only register. During a power-on reset, the counter is also preset to $FFFC and begins running after the oscillator start-up delay. Because the free-running counter is 16 bits preceded by a fixed divide-by-four prescaler, the value in the free-running counter repeats every 262,144 internal bus clock cycles. When the counter rolls over from $FFFF to $0000, the TOF bit is set. An interrupt can also be enabled when counter rollover occurs by setting its interrupt enable bit (TOIE).

OUTPUT COMPARE REGISTER

The 16-bit output compare register is made up of two 8-bit registers at locations $16 (MSB) and $17 (LSB). The output compare register is used for several purposes, such as indicating when a period of time has elapsed. All bits are readable and writable and are not altered by the timer hardware or reset. If the compare function is not needed, the two bytes of the output compare register can be used as storage locations.

The output compare register contents are compared with the contents of the free-running counter continually, and if a match is found, the corresponding output compare flag (OCF) bit is set and the corresponding output

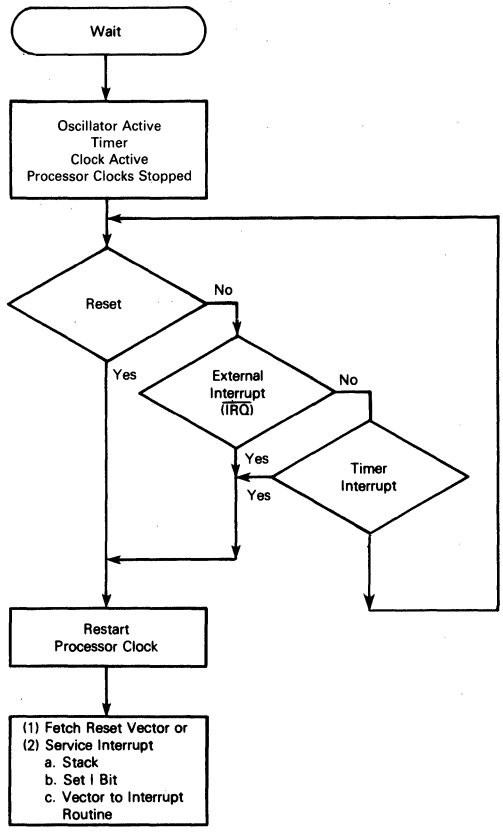


Figure 10. WAIT Function Flowchart

level (OLCL) bit is clocked to an output level register. The output compare register values and the output level bit should be changed after each successful comparison to establish a new elapsed timeout. An interrupt can also accompany a successful output compare provided the corresponding interrupt enable bit (OCIE) is set.

After a processor write cycle to the output compare register containing the MSB ($16), the output compare function is inhibited until the LSB ($17) is also written. The user must write both bytes (locations) if the MSB is written first. A write made only to the LSB ($17) will not inhibit the compare function. The free-running counter is updated every four internal bus clock cycles. The minimum time required to update the output compare register is a function of the program rather than the internal hardware.

The processor can write to either byte of the output compare register without affecting the other byte. The output level (OLVL) bit is clocked to the output level register regardless of whether the output compare flag (OCF) is set or clear.

INPUT CAPTURE REGISTER

Two 8-bit registers, which make up the 16-bit input capture register, are read-only and are used to latch the value of the free-running counter after the corresponding input capture edge detector senses a defined transition. The level transition which triggers the counter transfer is defined by the corresponding input edge bit (IEDG). Reset does not affect the contents of the input capture register.

The result obtained by an input capture will be one more than the value of the free-running counter on the rising edge of the internal bus clock preceding the external transition. This delay is required for internal synchronization. Resolution is one count of the free-running counter, which is four internal bus clock cycles.

The free-running counter contents are transferred to the input capture register on each proper signal transition regardless of whether the input capture flag (ICF) is set or clear. The input capture register always contains the free-running counter value that corresponds to the most recent input capture.

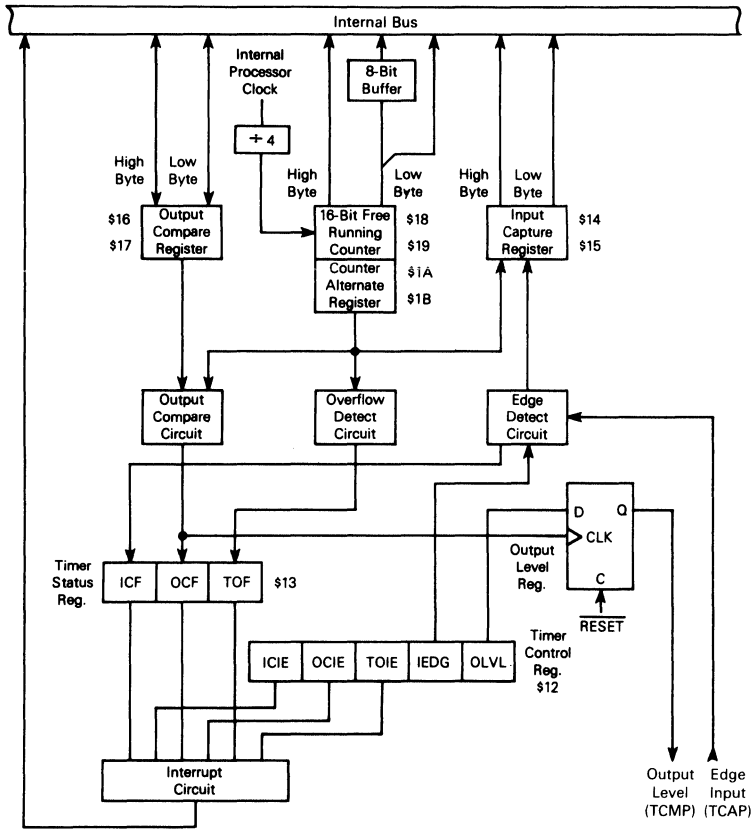


Figure 11. Timer Block Diagram

After a read of the input capture register ($14) MSB, the counter transfer is inhibited until the LSB ($15) is also read. This characteristic causes the time used in the input capture software routine and its interaction with the main program to determine the minimum pulse period.

A read of the input capture register LSB ($15) does not inhibit the free-running counter transfer since they occur on opposite edges of the internal bus clock.

TIMER CONTROL REGISTER (TCR) $12

The TCR is a read/write register containing five control bits. Three bits control interrupts associated with the timer status register flags ICF, OCF, and TOF.

7	6	5	4	3	2	1	0
ICIE	OCIE	TOIE	0	0	0	IEDG	OLVL
RESET:							
0	0	0	0	0	0	U	0

ICIE — Input Capture Interrupt Enable
- 1 = Interrupt enabled
- 0 = Interrupt disabled

OCIE — Output Compare Interrupt Enable
- 1 = Interrupt enabled
- 0 = Interrupt disabled

TOIE — Timer Overflow Interrupt Enable
- 1 = Interrupt enabled
- 0 = Interrupt disabled

IEDG — Input Edge

Value of input edge determines which level transition on TCAP pin will trigger free-running counter transfer to the input capture register
- 1 = Positive edge
- 0 = Negative edge

Reset does not affect the IEDG bit (U = unaffected).

OLVL — Output Level
Value of output level is clocked into output level register by the next successful output compare and will appear on the TCMP pin
1 = High output
0 = Low output
Bits 2, 3, and 4 — Not used
Always read zero

TIMER STATUS REGISTER (TSR) $13

The TSR is a read-only register containing three status flag bits.

7	6	5	4	3	2	1	0
ICF	OCF	TOF	0	0	0	0	0
RESET:							
U	U	U	0	0	0	0	0

ICF — Input Capture Flag
1 = Flag set when selected polarity edge is sensed by input capture edge detector
0 = Flag cleared when TSR and input capture low register ($15) are accessed
OCF — Output Compare Flag
1 = Flag set when output compare register contents match the free-running counter contents
0 = Flag cleared when TSR and output compare low register ($17) are accessed
TOF — Timer Overflow Flag
1 = Flag set when free-running counter transition from $FFFF to $0000 occurs
0 = Flag cleared when TSR and counter low register ($19) are accessed
Bits 0–4 — Not used
Always read zero

Accessing the timer status register satisfies the first condition required to clear status bits. The remaining step is to access the register corresponding to the status bit.

A problem can occur when using the timer overflow function and reading the free-running counter at random times to measure an elapsed time. Without incorporating the proper precautions into software, the timer overflow flag could unintentionally be cleared if:

1) The timer status register is read or written when TOF is set, and
2) The LSB of the free-running counter is read but not for the purpose of servicing the flag.

The counter alternate register at address $1A and $1B contains the same value as the free-running counter (at address $18 and $19); therefore, this alternate register can be read at any time without affecting the timer overflow flag in the timer status register.

TIMER DURING WAIT MODE

The CPU clock halts during the WAIT mode, but the timer remains active. An interrupt from the timer causes the processor to exit the WAIT mode.

TIMER DURING STOP MODE

In the STOP mode, the timer stops counting and holds the last count value if STOP is exited by an interrupt. If RESET is used, the counter is forced to $FFFC. During STOP, if at least one valid input capture edge occurs at the TCAP pin, the input capture detect circuit is armed. This does not set any timer flags nor wake up the MCU, but when the MCU does wake up, there is an active input capture flag and data from the first valid edge that occurred during the STOP mode. If RESET is used to exit STOP mode, then no input capture flag or data remains, even if a valid input capture edge occurred.

INSTRUCTION SET

The MCU has a set of 62 basic instructions. They can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

This MCU uses all the instructions available in the M146805 CMOS Family plus one more: the unsigned multiply (MUL) instruction. This instruction allows unsigned multiplication of the contents of the accumulator (A) and the index register (X). The high-order product is then stored in the index register, and the low-order product is stored in the accumulator. A detailed definition of the MUL instruction is shown below.

Operation	X:A X*A			
Description	Multiplies the eight bits in the index register by the eight bits in the accumulator to obtain a 16-bit unsigned number in the concatenated accumulator and index register			
Condition Codes	H: Cleared I: Not affected N: Not affected Z: Not affected C: Cleared			
Source	MUL			
Form(s)	Addressing Mode	Cycles	Bytes	Opcode
	Inherent	11	1	$42

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following instruction list.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC

— Continued

Function	Mnemonic
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (Twos Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST
Multiply	MUL

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any writable bit which resides in the first 256 bytes of the memory space where all port registers, port DDRs, timer, timer control, ROM, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear and bit test, and branch functions are all implemented with a single instruction. For test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch if Bit n is Set	BRSET n (n=0 . . . 7)
Branch if Bit n is Clear	BRCLR n (n=0 . . . 7)
Set Bit n	BSET n (n=0 . . . 7)
Clear Bit n	BCLR n (n=0 . . . 7)

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two-byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Higher	BHI
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
Branch if Higher or Same	BHS
Branch if Carry Set	BCS
Branch if Lower	BLO
Branch if Not Equal	BNE
Branch if Equal	BEQ
Branch if Half Carry Clear	BHCC
Branch if Half Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Bit is Clear	BMC
Branch if Interrupt Mask Bit is Set	BMS
Branch if Interrupt Line is Low	BIL
Branch if Interrupt Line is High	BIH
Branch to Subroutine	BSR

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No-Operation	NOP
Stop	STOP
Wait	WAIT

OPCODE MAP SUMMARY

Table 3 is an opcode map for the instructions used on the MCU.

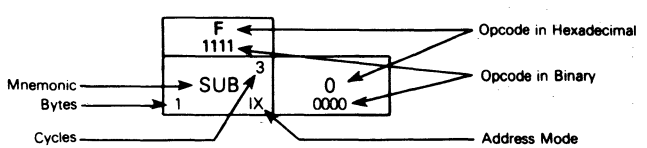
Table 3. Opcode Map

Hi / Low	Bit Manipulation BTB 0 0000	Bit Manipulation BSC 1 0001	Branch REL 2 0010	Read/Modify/Write DIR 3 0011	Read/Modify/Write INH 4 0100	Read/Modify/Write INH 5 0101	Read/Modify/Write IX1 6 0110	Read/Modify/Write IX 7 0111	Control INH 8 1000	Control INH 9 1001	Register/Memory IMM A 1010	Register/Memory DIR B 1011	Register/Memory EXT C 1100	Register/Memory IX2 D 1101	Register/Memory IX1 E 1110	Register/Memory IX F 1111	Hi / Low
0 0000	BRSET0 5 3 BTB	BSET0 5 2 BSC	BRA 3 2 REL	NEG 5 2 DIR	NEGA 3 1 INH	NEGX 3 1 INH	NEG 6 2 IX1	NEG 5 1 IX	RTI 9 1 INH		SUB 2 2 IMM	SUB 3 2 DIR	SUB 4 3 EXT	SUB 5 3 IX2	SUB 4 2 IX1	SUB 3 1 IX	0 0000
1 0001	BRCLR0 5 3 BTB	BCLR0 5 2 BSC	BRN 3 2 REL						RTS 6 1 INH		CMP 2 2 IMM	CMP 3 2 DIR	CMP 4 3 EXT	CMP 5 3 IX2	CMP 4 2 IX1	CMP 3 1 IX	1 0001
2 0010	BRSET1 5 3 BTB	BSET1 5 2 BSC	BHI 3 2 REL		MUL 11 1 INH						SBC 2 2 IMM	SBC 3 2 DIR	SBC 4 3 EXT	SBC 5 3 IX2	SBC 4 2 IX1	SBC 3 1 IX	2 0010
3 0011	BRCLR1 5 3 BTB	BCLR1 5 2 BSC	BLS 3 2 REL	COM 5 2 DIR	COMA 3 1 INH	COMX 3 1 INH	COM 6 2 IX1	COM 5 1 IX	SWI 10 1 INH		CPX 2 2 IMM	CPX 3 2 DIR	CPX 4 3 EXT	CPX 5 3 IX2	CPX 4 2 IX1	CPX 3 1 IX	3 0011
4 0100	BRSET2 5 3 BTB	BSET2 5 2 BSC	BCC 3 2 REL	LSR 5 2 DTR	LSRA 3 1 INH	LSRX 3 1 INH	LSR 6 2 IX1	LSR 5 1 IX			AND 2 2 IMM	AND 3 2 DIR	AND 4 3 EXT	AND 5 3 IX2	AND 4 2 IX1	AND 3 1 IX	4 0100
5 0101	BRCLR2 5 3 BTB	BCLR2 5 2 BSC	BCS 3 2 REL								BIT 2 2 IMM	BIT 3 2 DIR	BIT 4 3 EXT	BIT 5 3 IX2	BIT 4 2 IX1	BIT 3 1 IX	5 0101
6 0110	BRSET3 5 3 BTB	BSET3 5 2 BSC	BNE 3 2 REL	ROR 5 2 DIR	RORA 3 1 INH	RORX 3 1 INH	ROR 6 2 IX1	ROR 5 1 IX			LDA 2 2 IMM	LDA 3 2 DIR	LDA 4 3 EXT	LDA 5 3 IX2	LDA 4 2 IX1	LDA 3 1 IX	6 0110
7 0111	BRCLR3 5 3 BTB	BCLR3 5 2 BSC	BEQ 3 2 REL	ASR 5 2 DIR	ASRA 3 1 INH	ASRX 3 1 INH	ASR 6 2 IX1	ASR 5 1 IX		TAX 2 1 INH		STA 4 2 DIR	STA 5 3 EXT	STA 6 3 IX2	STA 5 2 IX1	STA 4 1 IX	7 0111
8 1000	BRSET4 5 3 BTB	BSET4 5 2 BSC	BHCC 3 2 REL	LSL 5 2 DIR	LSLA 3 1 INH	LSLX 3 1 INH	LSL 6 2 IX1	LSL 5 1 IX		CLC 2 1 INH	EOR 2 2 IMM	EOR 3 2 DIR	EOR 4 3 EXT	EOR 5 3 IX2	EOR 4 2 IX1	EOR 3 1 IX	8 1000
9 1001	BRCLR4 5 3 BTB	BCLR4 5 2 BSC	BHCS 3 2 REL	ROL 5 2 DIR	ROLA 3 1 INH	ROLX 3 1 INH	ROL 6 2 IX1	ROL 5 1 IX		SEC 2 1 INH	ADC 2 2 IMM	ADC 3 2 DIR	ADC 4 3 EXT	ADC 5 3 IX2	ADC 4 2 IX1	ADC 3 1 IX	9 1001
A 1010	BRSET5 5 3 BTB	BSET5 5 2 BSC	BPL 3 2 REL	DEC 5 2 DIR	DECA 3 1 INH	DECX 3 1 INH	DEC 6 2 IX1	DEC 5 1 IX		CLI 2 1 INH	ORA 2 2 IMM	ORA 3 2 DIR	ORA 4 3 EXT	ORA 5 3 IX2	ORA 4 2 IX1	ORA 3 1 IX	A 1010
B 1011	BRCLR5 5 3 BTB	BCLR5 5 2 BSC	BMI 3 2 REL							SEI 2 1 INH	ADD 2 2 IMM	ADD 3 2 DIR	ADD 4 3 EXT	ADD 5 3 IX2	ADD 4 2 IX1	ADD 3 1 IX	B 1011
C 1100	BRSET6 5 3 BTB	BSET6 5 2 BSC	BMC 3 2 REL	INC 5 2 DIR	INCA 3 1 INH	INCX 3 1 INH	INC 6 2 IX1	INC 5 1 IX		RSP 2 1 INH		JMP 2 2 DIR	JMP 3 3 EXT	JMP 4 3 IX2	JMP 3 2 IX1	JMP 2 1 IX	C 1100
D 1101	BRCLR6 5 3 BTB	BCLR6 5 2 BSC	BMS 3 2 REL	TST 4 2 DIR	TSTA 3 1 INH	TSTX 3 1 INH	TST 5 2 IX1	TST 4 1 IX		NOP 2 1 INH	BSR 6 2 REL	JSR 5 2 DIR	JSR 6 3 EXT	JSR 7 3 IX2	JSR 6 2 IX1	JSR 5 1 IX	D 1101
E 1110	BRSET7 5 3 BTB	BSET7 5 2 BSC	BIL 3 2 REL						STOP 2 1 INH		LDX 2 2 IMM	LDX 3 2 DIR	LDX 4 3 EXT	LDX 5 3 IX2	LDX 4 2 IX1	LDX 3 1 IX	E 1110
F 1111	BRCLR7 5 3 BTB	BCLR7 5 2 BSC	BIH 3 2 REL	CLR 5 2 DIR	CLRA 3 1 INH	CLRX 3 1 INH	CLR 6 2 IX1	CLR 5 1 IX	WAIT 2 1 INH	TXA 2 1 INH		STX 4 2 DIR	STX 5 3 EXT	STX 6 3 IX2	STX 5 2 IX1	STX 4 1 IX	F 1111

Abbreviations for Address Modes

INH	Inherent
A	Accumulator
X	Index Register
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single byte instructions; the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. One- or two-byte direct addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction. When using the Motorola assembler, the user need not specify whether an instruction uses direct or extended addressing. The assembler automatically selects the shortest form of the instruction.

RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −126 to +129 from the opcode address. The programmer need not calculate the offset when using the Motorola assembler, since it calculates the proper offset and checks to see that it is within the span of the branch.

INDEXED, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. This addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this two-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This address mode can be used in a manner similar to indexed, 8-bit offset except that this three-byte instruction allows tables to be anywhere in memory. As with direct and extended addressing, the Motorola assembler determines the shortest form of indexed addressing.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode, and the byte following the opcode specifies the direct addressing of the byte in which the specified bit is to be set or cleared. Any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single two-byte instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit that is to be tested and its condition (set or clear), is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS (Voltages referenced to V_{SS})

Rating	Symbol	Value	Unit
Supply Voltage	V_{DD}	−0.3 to +7.0	V
Input Voltage	V_{in}	V_{SS} −0.3 to V_{DD} +0.3	V
Self-Check Mode ($\overline{IRQ}$ Pin Only)	V_{in}	V_{SS} −0.3 to 2×V_{DD} +0.3	V
Current Drain Per Pin Excluding V_{DD} and V_{SS}	I	25	mA
Operating Temperature Range MC68HC05C2P, FN MC68HC05C2CP, CFN MC68HC05C2VP, VFN MC68HC05C2MP, MFN	T_A	T_L to T_H 0 to +70 −40 to +85 −40 to +105 −40 to +125	°C
Storage Temperature Range	T_{stg}	−65 to +150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation, it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{DD}$. Reliability of operation is enhanced if unused inputs are connected to an appropriate logic voltage level (e.g., either V_{SS} or V_{DD}).

3

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic Plastic Leaded Chip Carrier (PLCC)	θ_{JA}	 60 70	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{I/O}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
$P_{I/O}$ = Power Dissipation on Input and Output Pins — User Determined

For most applications $P_{I/O} < P_{INT}$ and can be neglected.

The following is an approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected):

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

V_{DD} = 4.5 V

Pins	R1	R2	C
PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4	3.26 kΩ	2.38 kΩ	50 pF
PD0, PD5, PD7	1.9 kΩ	2.26 kΩ	200 pF

V_{DD} = 3.0 V

Pins	R1	R2	C
PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4	10.91 kΩ	6.32 kΩ	50 pF
PD0, PD5, PD7	6 kΩ	6 kΩ	200 pF

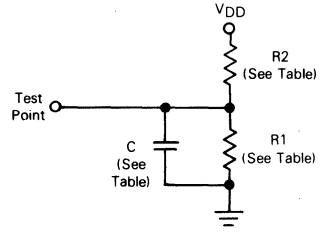


Figure 12. Equivalent Test Load

DC ELECTRICAL CHARACTERISTICS

(V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage, $I_{Load} \leq 10.0$ µA	V_{OL} V_{OH}	— $V_{DD}-0.1$	— —	0.1 —	V
Output High Voltage (I_{Load} = 0.8 mA) PA0-PA7, PB0-PB7, PC0-PC7, TCMP (see Figure 13) (I_{Load} = 1.6 mA) PD1-PD4 (see Figure 14)	V_{OH}	 $V_{DD}-0.8$ $V_{DD}-0.8$	 — —	 — —	V
Output Low Voltage (see Figure 15) (I_{Load} = 1.6 mA) PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4, TCMP	V_{OL}	—	—	0.4	V
Input High Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IH}	$0.7 \times V_{DD}$	—	V_{DD}	V
Input Low Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IL}	V_{SS}	—	$0.2 \times V_{DD}$	V
Data Retention Mode (0° to 70°C)	V_{RM}	2.0	—	—	V
Supply Current (see Notes) Run (see Figures 16 and 17) Wait (see Figures 16 and 17) Stop (see Figure 17) 25°C 0° to 70°C (Standard) −40° to +85°C −40° to +125°C	I_{DD}	 — — — — — —	 3.5 1.6 2.0 — — —	 7.0 4.0 50 140 180 250	 mA mA µA µA µA µA
I/O Ports Hi-Z Leakage Current PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4	I_{IL}	—	—	±10	µA
Input Current $\overline{RESET}$, $\overline{IRQ}$, TCAP, OSC1, PD0, PD5, PD7	I_{in}	—	—	±1	µA
Capacitance Ports (as Input or Output) $\overline{RESET}$, $\overline{IRQ}$, TCAP, PD0-PD5, PD7	 C_{out} C_{in}	 — —	 — —	 12 8	pF

NOTES:
1. All values shown reflect average measurements.
2. Typical values at midpoint of voltage range, 25°C only.
3. Wait I_{DD}: Only timer system active (SPE = TE = RE = 0). If SPI, SCI active (SPE = TE = RE = 1) add 10% current draw.
4. Run (Operating) I_{DD}, Wait I_{DD}: Measured using external square wave clock source (f_{osc} = 4.2 MHz), all inputs 0.2 V from rail; no dc loads, less than 50 pF on all outputs, C_L = 20 pF on OSC2.
5. Wait, Stop I_{DD}: All ports configured as inputs, V_{IL} = 0.2 V, $V_{IH} = V_{DD} - 0.2$ V.
6. Stop I_{DD} measured with OSC1 = V_{SS}.
7. Standard temperature range is 0° to 70°C. Extended temperature versions and a 25°C only version are available.
8. Wait I_{DD} is affected linearly by the OSC2 capacitance.

DC ELECTRICAL CHARACTERISTICS

(V_{DD} = 3.3 Vdc ± 0.3 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage, $I_{Load} \leq 10.0$ µA	V_{OL} V_{OH}	— $V_{DD}-0.1$	— —	0.1 —	V
Output High Voltage (I_{Load} = 0.2 mA) PA0-PA7, PB0-PB7, PC0-PC7, TCMP (see Figure 13) (I_{Load} = 1.6 mA) PD1-PD4 (see Figure 14)	V_{OH}	 $V_{DD}-0.3$ $V_{DD}-0.3$	 — —	 — —	V
Output Low Voltage (see Figure 15) (I_{Load} = 0.4 mA) PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4, TCMP	V_{OL}	—	—	0.3	V
Input High Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IH}	$0.7 \times V_{DD}$	—	V_{DD}	V
Input Low Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IL}	V_{SS}	—	$0.2 \times V_{DD}$	V
Data Retention Mode (0° to 70°C)	V_{RM}	2.0	—	—	V
Supply Current (see Notes) Run (see Figures 16 and 18) Wait (see Figures 16 and 18) Stop (see Figure 18) 25°C 0° to 70°C (Standard) −40° to +85°C −40° to +125°C	I_{DD}	 — — — — — —	 1.0 0.5 1.0 — — —	 2.5 1.4 30 80 120 175	 mA mA µA µA µA µA
I/O Ports Hi-Z Leakage Current PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4	I_{IL}	—	—	±10	µA
Input Current RESET, IRQ, TCAP, OSC1, PD0, PD5, PD7	I_{in}	—	—	±1	µA
Capacitance Ports (as Input or Output) RESET, IRQ, TCAP, PD0-PD5, PD7	 C_{out} C_{in}	 — —	 — —	 12 8	pF

NOTES:
1. All values shown reflect average measurements.
2. Typical values at midpoint of voltage range, 25°C only.
3. Wait I_{DD}: Only timer system active (SPE = TE = RE = 0). If SPI, SCI active (SPE = TE = RE = 1) add 10% current draw.
4. Run (Operating) I_{DD}, Wait I_{DD}: Measured using external square wave clock source (f_{osc} = 4.2 MHz), all inputs 0.2 V from rail; no dc loads, less than 50 pF on all outputs, C_L = 20 pF on OSC2.
5. Wait, Stop I_{DD}: All ports configured as inputs, V_{IL} = 0.2 V, $V_{IH} = V_{DD} - 0.2$ V.
6. Stop I_{DD} measured with OSC1 = V_{SS}.
7. Standard temperature range is 0° to 70°C. Extended temperature versions and a 25°C only version are available.
8. Wait I_{DD} is affected linearly by the OSC2 capacitance.

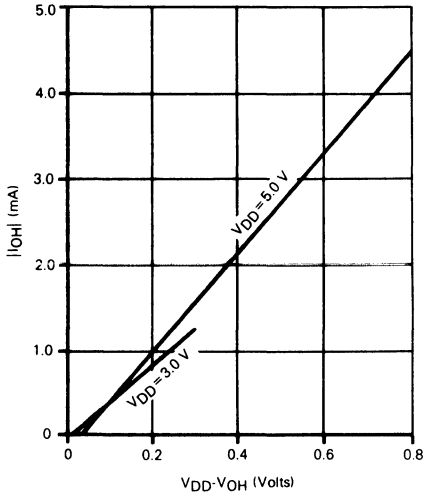


Figure 13. Typical V_{OH} vs I_{OH} for Ports A, B, C, and TCMP

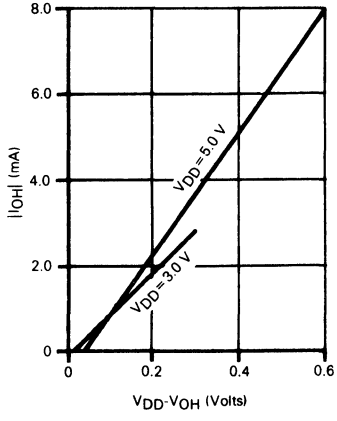


Figure 14. Typical V_{OH} vs I_{OH} for PD1-PD4

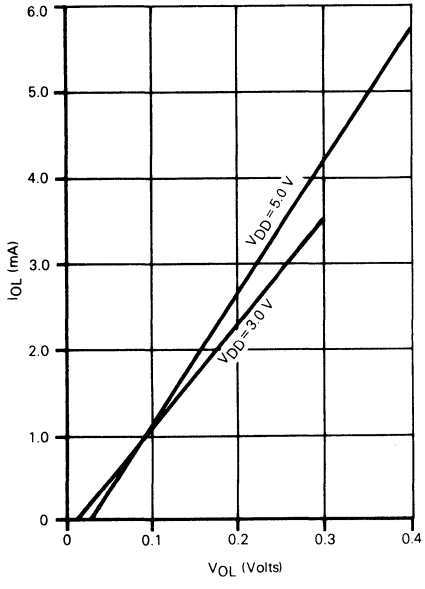


Figure 15. Typical V_{OL} vs I_{OL} for All Ports

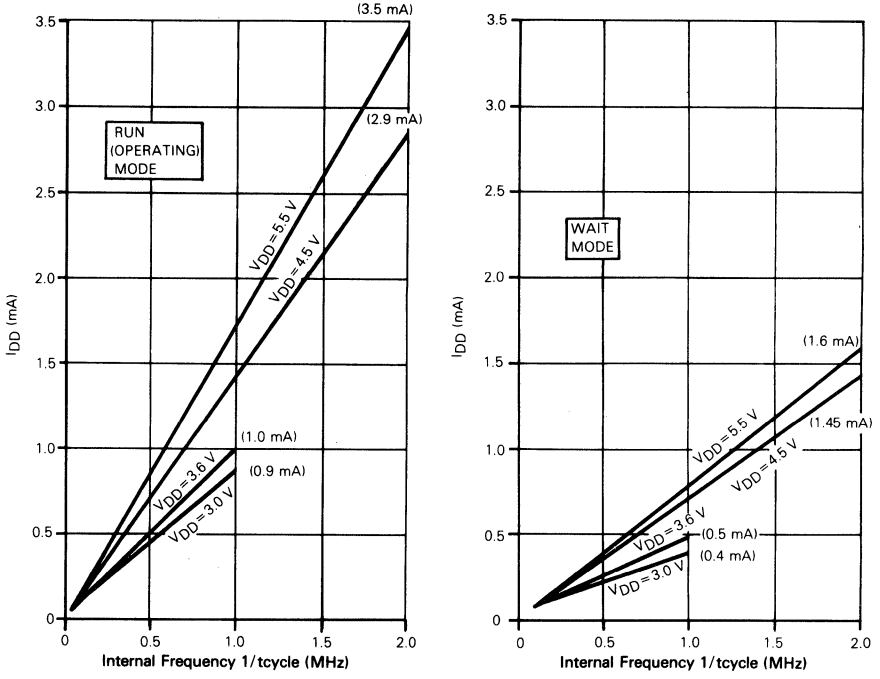


Figure 16. Typical Current vs Internal Frequency for Run and Wait Modes

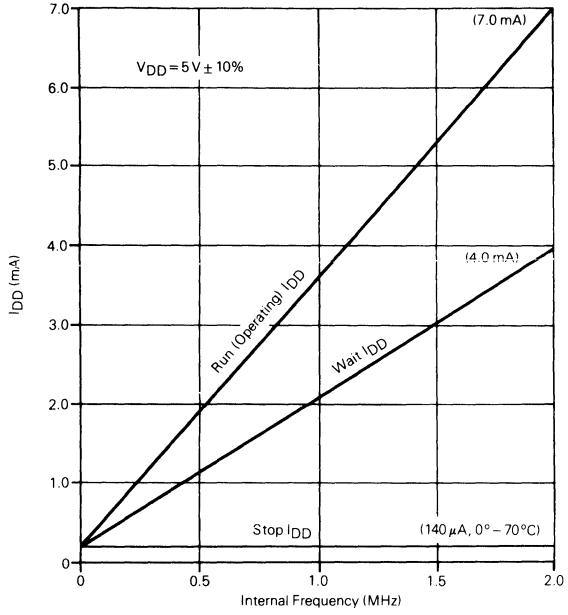


Figure 17. Maximum I_{DD} vs Frequency for V_{DD} = 5.0 Vdc

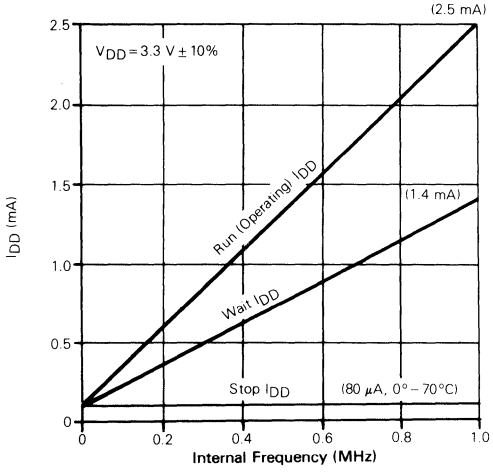


Figure 18. Maximum I_{DD} vs Frequency for V_{DD} = 3.3 Vdc

CONTROL TIMING

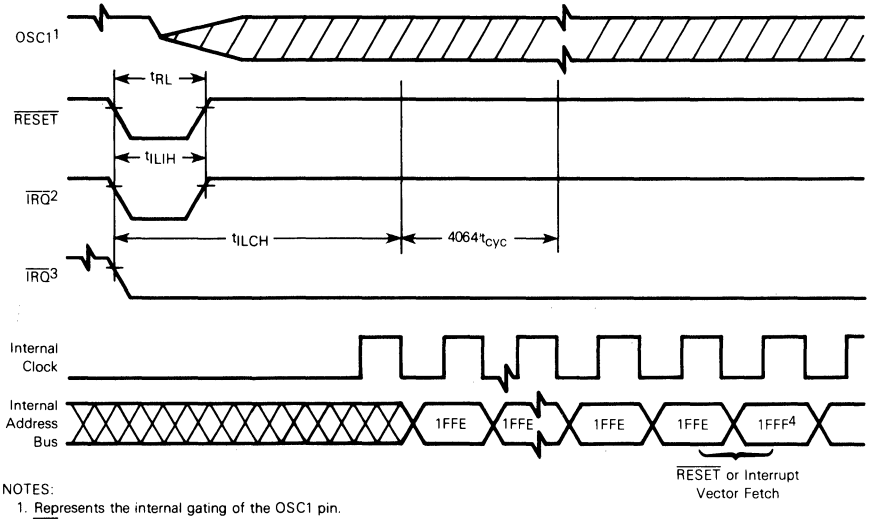
(V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation Crystal Option External Clock Option	f_{osc}	 — dc	 4.2 4.2	MHz
Internal Operating Frequency Crystal ($f_{osc} \div 2$) External Clock ($f_{osc} \div 2$)	f_{op}	 — dc	 2.1 2.1	MHz
Cycle Time (see Figure 21)	t_{cyc}	480	—	ns
Crystal Oscillator Startup Time (see Figure 21)	t_{OXOV}	—	100	ms
Stop Recovery Startup Time (Crystal Oscillator) (see Figure 19)	t_{ILCH}	—	100	ms
$\overline{\text{RESET}}$ Pulse Width (see Figure 21)	t_{RL}	1.5	—	t_{cyc}
Timer Resolution** Input Capture Pulse Width (see Figure 20) Input Capture Pulse Period (see Figure 20)	 t_{RESL} t_{TH}, t_{TL} t_{TLTL}	 4.0 125 ***	 — — —	 t_{cyc} ns t_{cyc}
Interrupt Pulse Width Low (Edge-Triggered) (see Figure 8)	t_{ILIH}	125	—	ns
Interrupt Pulse Period (see Figure 8)	t_{ILIL}	*	—	t_{cyc}
OSC1 Pulse Width	t_{OH}, t_{OL}	90	—	ns

*The minimum period t_{ILIL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus 21 t_{cyc}.

**Since a 2-bit prescaler in the timer must count four internal cycles (t_{cyc}), this is the limiting minimum factor in determining the timer resolution.

***The minimum period t_{TLTL} should not be less than the number of cycle times it takes to execute the capture interrupt service routine plus 24 t_{cyc}.



NOTES:
1. Represents the internal gating of the OSC1 pin.
2. $\overline{\text{IRQ}}$ pin edge-sensitive mask option.
3. $\overline{\text{IRQ}}$ pin level and edge-sensitive mask option.
4. $\overline{\text{RESET}}$ vector address shown for timing example.

Figure 19. Stop Recovery Timing Diagram

CONTROL TIMING

(V_{DD} = 3.3 Vdc ± 0.3 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation Crystal Option External Clock Option	f_{osc}	 — dc	 2.0 2.0	MHz
Internal Operating Frequency Crystal ($f_{osc} \div 2$) External Clock ($f_{osc} \div 2$)	f_{op}	 — dc	 1.0 1.0	MHz
Cycle Time (see Figure 21)	t_{cyc}	1000	—	ns
Crystal Oscillator Startup Time (see Figure 21)	t_{OXOV}	—	100	ms
Stop Recovery Startup Time (Crystal Oscillator) (see Figure 19)	t_{ILCH}	—	100	ms
$\overline{\text{RESET}}$ Pulse Width — Excluding Power-Up (see Figure 21)	t_{RL}	1.5	—	t_{cyc}
Timer Resolution** Input Capture Pulse Width (see Figure 20) Input Capture Pulse Period (see Figure 20)	 t_{RESL} t_{TH}, t_{TL} t_{TLTL}	 4.0 250 ***	 — — —	 t_{cyc} ns t_{cyc}
Interrupt Pulse Width Low (Edge-Triggered) (see Figure 8)	t_{ILIH}	250	—	ns
Interrupt Pulse Period (see Figure 8)	t_{ILIL}	*	—	t_{cyc}
OSC1 Pulse Width	t_{OH}, t_{OL}	200	—	ns

*The minimum period t_{ILIL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus 21 t_{cyc}.

**Since a 2-bit prescaler in the timer must count four internal cycles (t_{cyc}), this is the limiting minimum factor in determining the timer resolution.

***The minimum period t_{TLTL} should not be less than the number of cycle times it takes to execute the capture interrupt service routine plus 24 t_{cyc}.

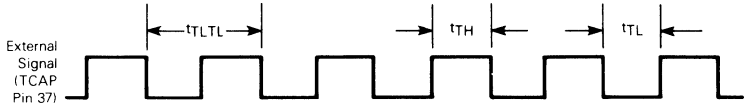
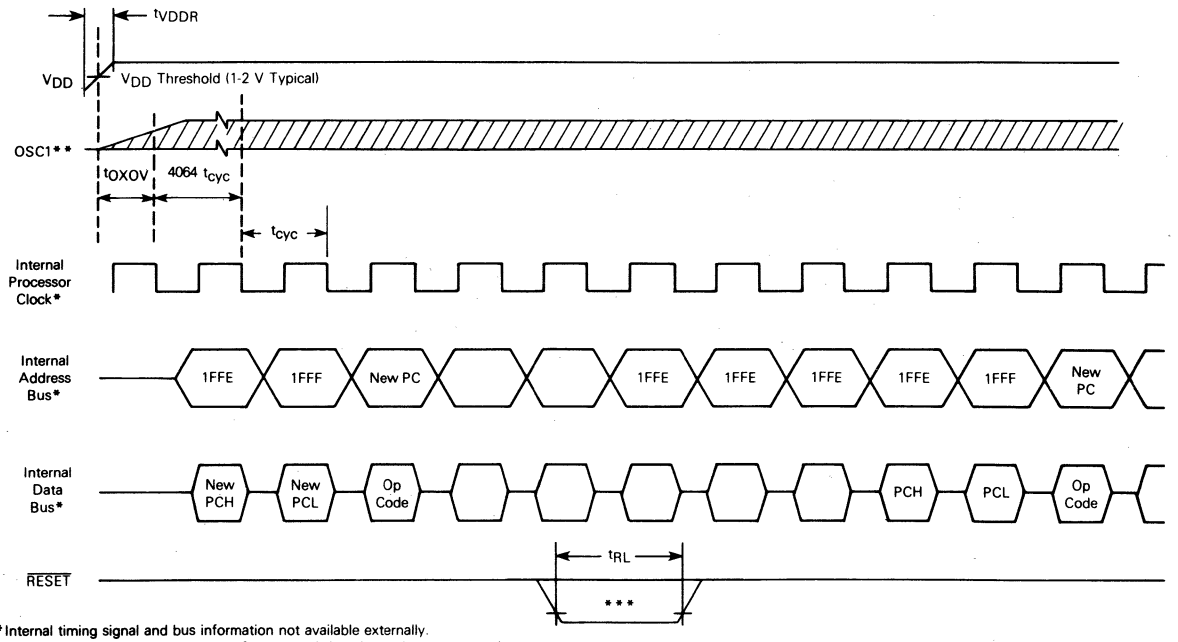


Figure 20. Timer Relationships



*Internal timing signal and bus information not available externally.
**OSC1 line is not meant to represent frequency. It is only used to represent time.
***The next rising edge of the internal processor clock following the rising edge of $\overline{\text{RESET}}$ initiates the reset sequence.

Figure 21. Power-On Reset and $\overline{\text{RESET}}$

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola in the following media:

MDOS™, disk file
MS™-DOS/PC-DOS disk file (360K)
EPROM(s) 2764, MCM68764, MCM68766, or EEPROM MC68HC805C4

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, a sales person, or a Motorola representative.

FLEXIBLE DISKS

A flexible disk (MS-DOS/PC-DOS disk file), programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. The diskette should be clearly labeled with the customer's name, data, project or product name, and the name of the file containing the pattern.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to expedite the process in case of any difficulty with the pattern file.

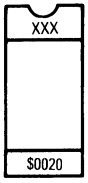
MS-DOS/PC-DOS Disk File

MS-DOS is Microsoft's Disk Operating System. PC-DOS is the IBM® Personal Computer (PC) Disk Operating System. Disk media submitted must be a standard density (360K) double-sided 5 1/4 inch compatible floppy diskette. The diskette must contain object file code in Motorola's S-record format. The S-record format is a character-based object file format generated by M6805 cross assemblers and linkers on IBM PC style machines.

EPROMs

A 2764, 68764, or 68766 type EPROM, programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. Since all program and data space information will fit on one 2764, 68764, or 68766 EPROM device, the EPROM must be programmed as described in the following paragraphs.

For an MC68HC805C4 MCU start the page zero, user ROM at EEPROM address $0020 through $004F. Start the user ROM at EEPROM address $0100 through $08FF with vectors from $1FF4 to $1FFF. All unused bytes, including the user's space, must be set to zero. For shipment to Motorola, EPROMs should be placed in a conductive IC carrier and packed securely. Styrofoam is not acceptable for shipment.



xxx = Customer ID

Verification Media

All original pattern media (EPROMs or floppy disks) are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked, and the verification form should be completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for the creation of the customer mask. To aid in the verification process, Motorola will program *customer supplied* blank EPROM(s) or DOS disks from the data file used to create the custom mask.

ROM VERIFICATION UNITS (RVUs)

Ten MCUs containing the customer's ROM pattern will be sent for program verification. These units will have been made using the custom mask, but are for the purpose of ROM verification only. For expediency, the MCUs are unmarked, packaged in ceramic, and tested with five volts at room temperature. These RVUs are free with the minimum order quantity, but are not production parts. RVUs are not backed or guaranteed by Motorola Quality Assurance.

ORDERING INFORMATION

The following table provides ordering information pertaining to the package type, temperature, and MC order numbers for the MC68HC05C2 device.

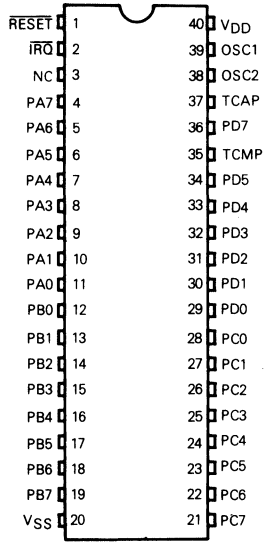
Package Type	Temperature	MC Order Number
Plastic (P Suffix)	0°C to +70°C	MC68HC05C2P
	−40°C to +85°C	MC68HC05C2CP
	−40° to +105°C	MC68HC05C2VP
	−40°C to +125°C	MC68HC05C2MP
PLCC (FN Suffix)	0°C to +70°C	MC68HC05C2FN
	−40°C to +85°C	MC68HC05C2CFN
	−40°C to +105°C	MC68HC05C2VFN
	−40°C to +125°C	MC68HC05C2MFN

3

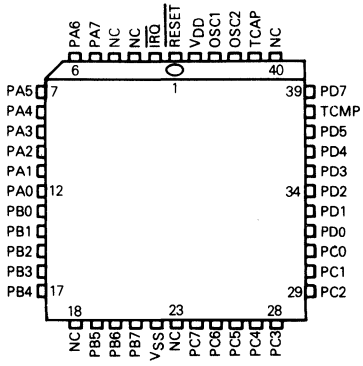
MDOS is a trademark of Motorola Inc.
MS is a trademark of Microsoft, Inc.
IBM is a registered trademark of International Business Machines Corporation.

PIN ASSIGNMENTS

40-PIN DUAL-IN-LINE PACKAGE



44-LEAD PLCC PACKAGE



NOTE: Bulk substrate tied to V_{SS}.

MC68HC05C3

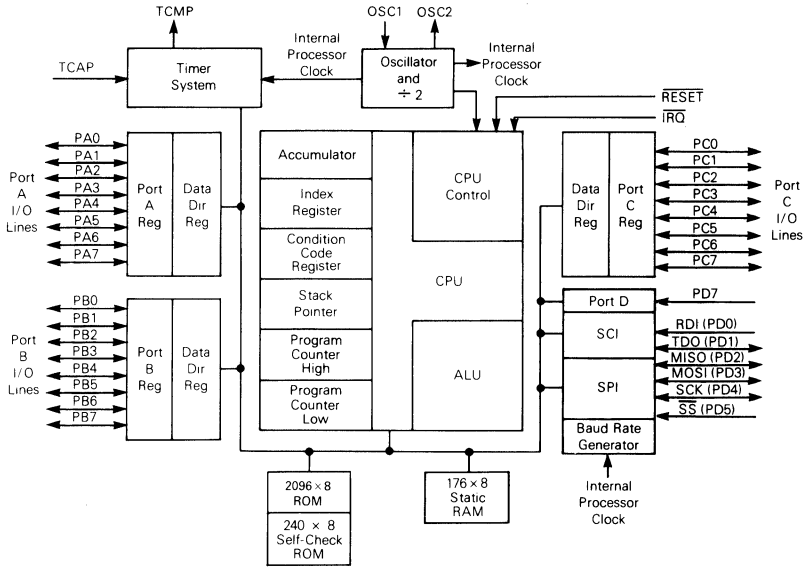
Technical Summary
8-Bit Microcontroller Unit

The MC68HC05C3 (HCMOS) microcontroller unit (MCU) is a member of the M68HC05 Family of microcontrollers. This high-performance, low-power MCU has parallel I/O capability with pins programmable as input or output. This publication contains condensed information on the MCU; for more detailed information, contact your local Motorola sales office.

The following block diagram depicts the hardware features; additional features available on the MCU are as follows:

- On-Chip Oscillator with RC or Crystal/Ceramic Resonator Mask Options
- Memory-Mapped I/O
- 176 Bytes of On-Chip RAM
- 2096 Bytes of User ROM
- 24 Bidirectional I/O Lines and 7 Input-Only Lines
- Serial Communications Interface (SCI) System
- Serial Peripheral Interface (SPI) System
- Self-Check Mode
- Power-Saving STOP, WAIT, and Data Retention Modes
- Single 3.0- to 5.5-Volt Supply (2-Volt Data Retention Mode)
- Fully Static Operation
- 8×8 Unsigned Multiply Instruction

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

The signal descriptions of the MCU are discussed in the following paragraphs.

V_{DD} AND V_{SS}

Power is supplied to the microcontroller using these two pins. V_{DD} is the positive supply, and V_{SS} is ground.

$\overline{IRQ}$

This pin is a programmable option that provides two different choices of interrupt triggering sensitivity. Refer to **INTERRUPTS** for more detail.

OSC1, OSC2

These pins provide control input for an on-chip clock oscillator circuit. A crystal, a ceramic resonator, a resistor/capacitor combination, or an external signal connects to these pins providing a system clock. A mask option selects either a crystal/ceramic resonator or a resistor/capacitor as the frequency determining element. The oscillator frequency is two times the internal bus rate.

RC Oscillator

With this option, a resistor is connected to the oscillator pins as shown in Figure 1(d). The relationship between R and f_{osc} is shown in Figure 2.

Crystal

The circuit shown in Figure 1(b) is recommended when using a crystal. Using an external CMOS oscillator is recommended when crystals outside the specified ranges are to be used. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and start-up stabilization time. Refer to **ELECTRICAL SPECIFICATIONS** for V_{DD} specifications.

Ceramic Resonator

A ceramic resonator may be used in place of the crystal in cost-sensitive applications. The circuit in Figure 1(b) is recommended when using a ceramic resonator. Figure 1(a) lists the recommended capacitance and resistance values. The manufacturer of the resonator considered

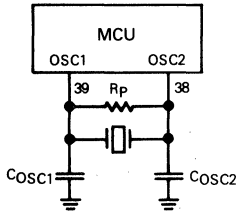
Crystal

	2 MHz	4 MHz	Units
R_{SMAX}	400	75	Ω
C_0	5	7	pF
C_1	0.008	0.012	μF
C_{OSC1}	15-40	15-30	pF
C_{OSC2}	15-30	15-25	pF
R_P	10	10	MΩ
Q	30	40	K

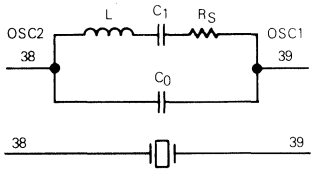
Ceramic Resonator

	2-4 MHz	Units
R_S (typical)	10	Ω
C_0	40	pF
C_1	4.3	pF
C_{OSC1}	30	pF
C_{OSC2}	30	pF
R_P	1-10	MΩ
Q	1250	–

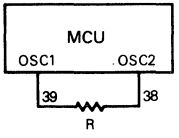
(a) Crystal/Ceramic Resonator Parameters



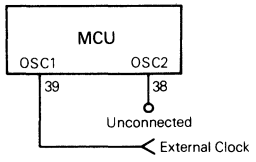
(b) Crystal/Ceramic Resonator Oscillator Connections



(c) Equivalent Crystal Circuit



(d) RC Oscillator Connections



(e) External Clock Source Connections (For Crystal Mask Option Only)

Figure 1. Oscillator Connections

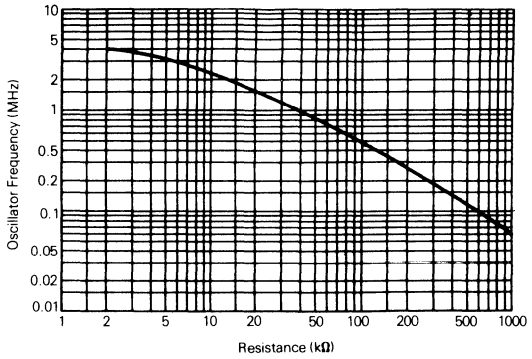


Figure 2. Typical Frequency vs Resistance for RC Oscillator Option Only

should be consulted for specific information on resonator operation.

External Clock

An external clock should be applied to the OSC1 input with the OSC2 input not connected, as shown in Figure 1(e). This option may only be used with the crystal oscillator mask option.

INPUT CAPTURE (TCAP)

This pin controls the input capture feature for the on-chip programmable timer.

OUTPUT COMPARE (TCMP)

This pin provides an output for the output compare feature of the on-chip timer.

$\overline{\text{RESET}}$

This pin is used to reset the MCU and provide an orderly start-up procedure by pulling $\overline{\text{RESET}}$ low.

INPUT/OUTPUT PORTS (PA0-PA7, PB0-PB7, PC0-PC7)

These 24 lines are arranged into three 8-bit ports (A, B, and C). These ports are programmable as either inputs or outputs under software control of the data direction registers. Refer to **PROGRAMMING** for additional information.

FIXED INPUT PORT (PD0-PD5, PD7)

These seven lines comprise port D, a fixed input port. All special functions that are enabled (SPI, SCI) affect this port. Refer to **PROGRAMMING** for additional information.

PROGRAMMING

Input/output port programming, fixed input port programming, and serial port programming are discussed in the following paragraphs.

INPUT/OUTPUT PORT PROGRAMMING

Any port pin is programmable as either an input or an output under software control of the corresponding data direction register (DDR). Each port bit can be selected as output or input by writing the corresponding bit in the port DDR to a logic one for output and logic zero for input. On reset, all DDRs are initialized to logic zero to put the ports in the input mode. The port output registers are not initialized on reset but may be written to before setting the DDR bits to avoid undefined levels.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This port write may be used to initialize the data registers and avoid undefined outputs. Refer to Figure 3 for typical port circuitry and to Table 1 for a list of the I/O pin functions.

Table 1. I/O Pin Functions

R/$\overline{\text{W}}$*	DDR	I/O Pin Functions
0	0	The I/O pin is in input mode. Data is written into the output data latch.
0	1	Data is written into the output data latch and output to the I/O pin.
1	0	The state of the I/O pin is read.
1	1	The I/O pin is in an output mode. The output data latch is read.

*R/$\overline{\text{W}}$ is an internal signal.

FIXED INPUT PORT PROGRAMMING

Port D is a fixed input port (PD0-PD5, PD7) that monitors the external pins whenever the SCI or SPI is disabled. After reset, all seven bits become valid inputs because all special function drivers are disabled. For example, with the SCI enabled, PD0 and PD1 inputs will read zero.

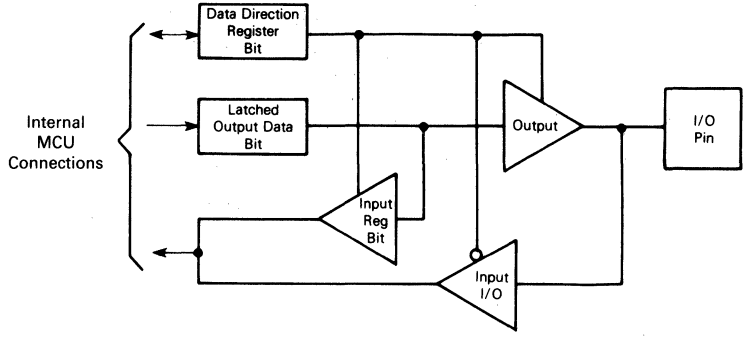


Figure 3. Typical Port I/O Circuit

With the SPI disabled, PD2 through PD5 will read the state of the pin at the time of the read operation.

NOTE

Any unused inputs and I/O ports should be tied to an appropriate logic level (e.g., either V_{DD} or V_{SS}).

SERIAL PORT (SCI AND SPI) PROGRAMMING

The SCI and SPI use the port D pins for their functions. The SCI requires two pins (PD0-PD1) for its receive data input (RDI) and transmit data output (TD0), respectively. The SPI function requires four of the pins (PD2-PD5) for its serial data input/output (MISO), serial data output/input (MOSI), serial clock (SCK), and slave select ($\overline{SS}$), respectively.

MEMORY

The MCU is capable of addressing 8192 bytes of memory and I/O registers, as shown in Figure 4. The locations consist of user ROM, user RAM, self-check ROM, control registers, and I/O. The user-defined reset and interrupt vectors are located from $1FF4 to $1FFF.

The shared stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer decrements during pushes and increments during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

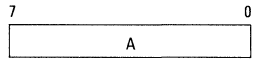
Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

REGISTERS

The MCU contains the registers described in the following paragraphs.

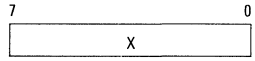
ACCUMULATOR (A)

The accumulator is a general-purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



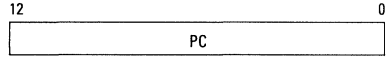
INDEX REGISTER (X)

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16-bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



PROGRAM COUNTER (PC)

The program counter is a 13-bit register that contains the address of the next byte to be fetched.



STACK POINTER (SP)

The stack pointer is a 13-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set to location $00FF. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

When accessing memory, the seven most significant bits are permanently set to 0000011. These seven bits are appended to the six least significant register bits to produce an address within the range of $00FF to $00C0. Subroutines and interrupts may use up to 64 (decimal) locations. If 64 locations are exceeded, the stack pointer

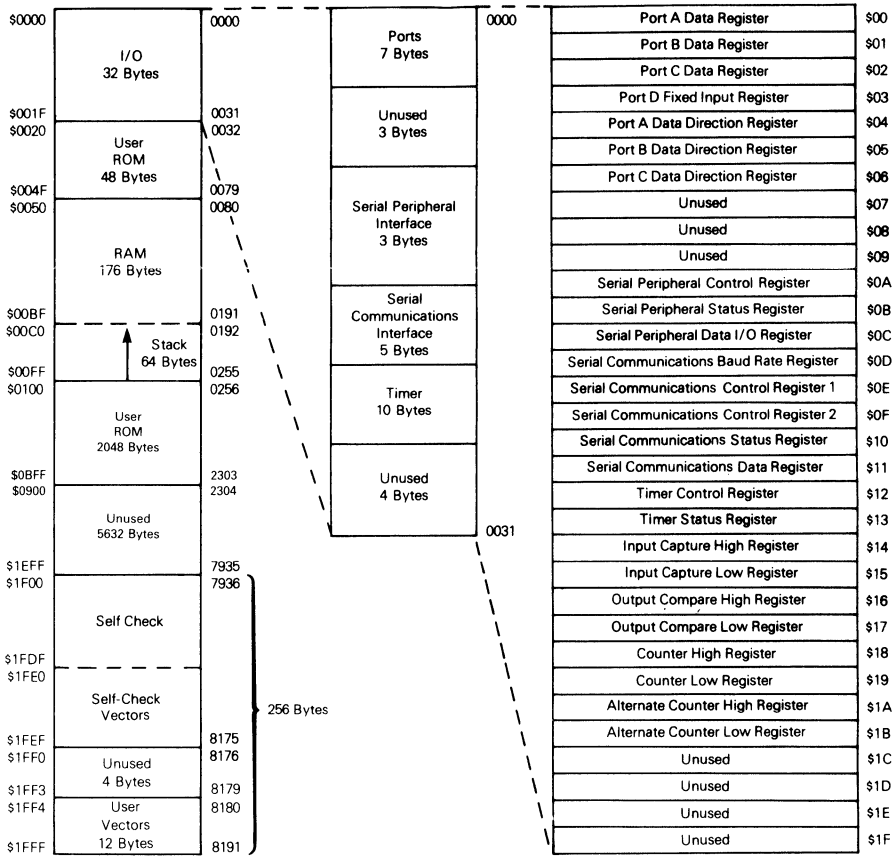
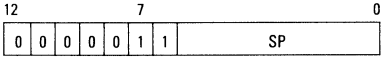


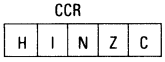
Figure 4. Memory Map

wraps around and loses the previously stored information. A subroutine call occupies two locations on the stack; an interrupt uses five locations.



CONDITION CODE REGISTER (CCR)

The CCR is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program, and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.



Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, the timer and external interrupt is masked (disabled). If an interrupt occurs while this bit is set, the interrupt is latched and processed as soon as the interrupt bit is cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic one).

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/Borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions and during shifts and rotates.

SELF-CHECK

The self-check capability provides the ability to determine if the device is functional. Self-check is performed using the circuit shown in Figure 5. Port C pins PC0-PC3 are monitored for the self-check results. After reset, the following seven tests are performed automatically:

I/O — Exercise of ports A, B, and C
RAM — Counter test for each RAM byte
ROM — Exclusive OR with odd ones parity result
Timer — Tracks counter register and checks OCF flag
Interrupts — Tests external, timer, SCI and SPI interrupts
SCI — Transmission test; checks RDRF, TDRE, TC, and FE flags
SPI — Transmission test; checks SPIF, WCOL, and MODF flags

Self-check results (using the LEDs as monitors) are shown in Table 2. The following subroutines are available to the user and do not require any external hardware.

TIMER TEST SUBROUTINE

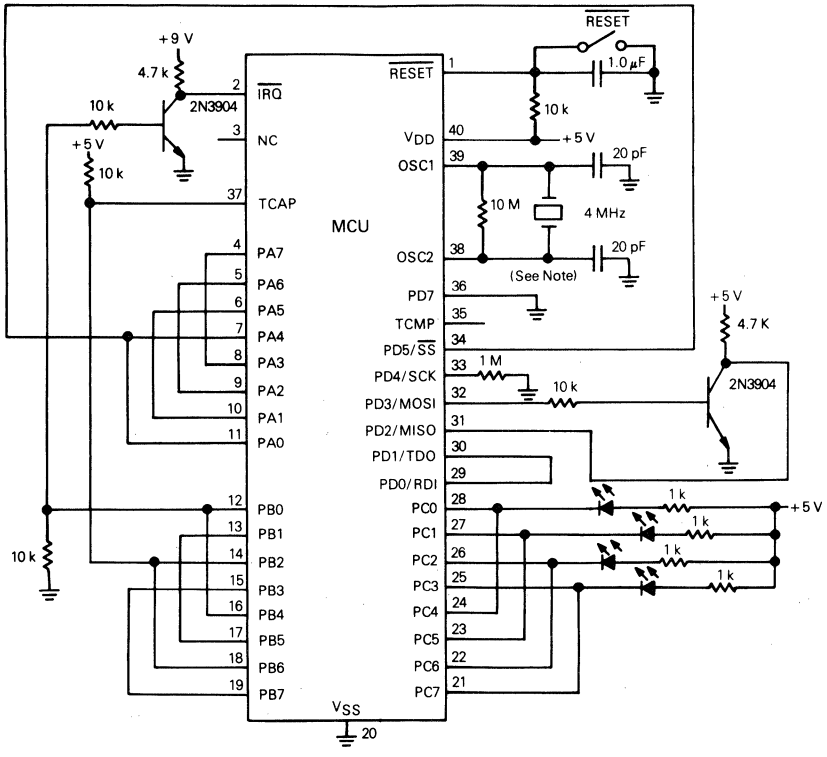
This subroutine returns with the Z bit cleared if any error is detected; otherwise, the Z bit is set. The timer test subroutine is called at location $1FOE. The output compare register is first set to the current timer state. Because the timer is free running and has only a divide-by-four prescaler, each timer count cannot be tested. The test reads the timer once every 10 counts (40 cycles) and checks for correct counting. The test tracks the counter until the timer wraps around, triggering the output compare flag in the timer status register. RAM locations $0050 and $0051 are overwritten. Upon return to the user's program, X=40. If the test passed, A=0.

ROM CHECKSUM SUBROUTINE

This subroutine returns with the Z bit cleared if any error is detected; otherwise, the Z bit is set. The ROM checksum subroutine is called at location $1F93 with RAM location $0053 equal to $01 and A=0. A short routine is set up and executed in RAM to compute a checksum of the entire ROM pattern. RAM locations $0050 through $0053 are overwritten. Upon return to the user's program, X=0. If the test passed, A=0.

RESETS

The MCU can be reset two ways: by initial power-up and by the external reset input ($\overline{\text{RESET}}$). The $\overline{\text{RESET}}$ input



NOTE: The RC Oscillator Option may also be used in this circuit.

Figure 5. Self-Check Circuit Schematic Diagram

Table 2. Self-Check Results

PC3	PC2	PC1	PC0	Remarks
1	0	0	1	Bad I/O
1	0	1	0	Bad RAM
1	0	1	1	Bad Timer
1	1	0	0	Bad SCI
1	1	0	1	Bad ROM
1	1	1	0	Bad SPI
1	1	1	1	Bad Interrupts or $\overline{IRQ}$ Request
Flashing				Good Device
All Others				Bad Device, Bad Port C, etc.

0 indicates LED is on; 1 indicates LED is off.

consists mainly of a Schmitt trigger that senses the $\overline{RESET}$ line logic level.

POWER-ON RESET (POR)

An internal reset is generated on power-up to allow the internal clock generator to stabilize. The power-on reset is strictly for power turn-on conditions and should not be used to detect a drop in the power supply voltage. There is a 4064 internal processor clock cycle (t_{cyc}) delay after the oscillator becomes active. If the $\overline{RESET}$ pin is low at the end of 4046 t_{cyc}, the MCU will remain in the reset condition until $\overline{RESET}$ goes high.

EXTERNAL RESET INPUT

The MCU is reset when a logic zero is applied to the $\overline{RESET}$ input for a period of one and one-half machine cycles (t_{cyc}).

INTERRUPTS

The MCU can be interrupted five different ways: the four maskable hardware interrupts ($\overline{IRQ}$, SPI, SCI, and timer) and the nonmaskable software interrupt instruction (SWI).

Interrupts cause the processor to save register contents on the stack and to set the interrupt mask (I bit) to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack and normal processing to resume. The stacking order is shown in Figure 6.

Unlike $\overline{RESET}$, hardware interrupts do not cause the current instruction execution to be halted but are considered pending until the current instruction is complete.

NOTE

The current instruction is the one already fetched and being operated on.

When the current instruction is complete, the processor checks all pending hardware interrupts. If unmasked (I bit clear) and if the corresponding interrupt enable bit is set, the processor proceeds with interrupt processing; otherwise, the next instruction is fetched and executed.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction, regardless of the I-bit state. Refer to Figure 7 for the reset and interrupt instruction processing sequence.

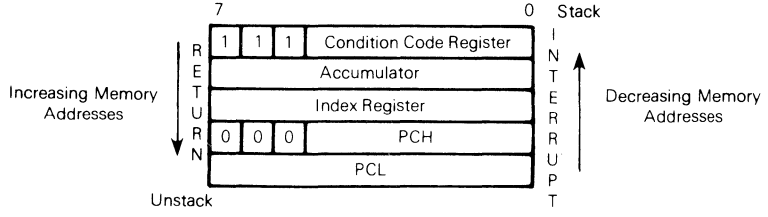
TIMER INTERRUPT

There are three different timer interrupt flags that cause a timer interrupt whenever they are set and enabled. The interrupt flags are in the timer status register (TSR), and the enable bits are in the timer control register (TCR). Refer to **TIMER** for more information.

EXTERNAL INTERRUPT

If the interrupt mask bit (I bit) of the CCR is set, all interrupts are disabled. Clearing the I bit enables the external interrupt. The external interrupt is internally synchronized and then latched on the falling edge of $\overline{IRQ}$. The action of the external interrupt is identical to the timer interrupt with the exception that the interrupt request input at $\overline{IRQ}$ is latched internally and the service routine address is specified by the contents of $1FFA and $1FFB.

Either a level-sensitive and edge-sensitive trigger, or an edge-sensitive-only trigger are available as a mask option. Figure 8 shows both a functional internal diagram and a mode timing diagram for the interrupt line. The timing diagram shows two treatments of the interrupt line to the processor. The first method shows a single pulse on the interrupt line spaced far enough apart to be



NOTE: Since the Stack Pointer decrements during pushes, the PCL is stacked first, followed by PCH, etc. Pulling from the stack is in the reverse order.

Figure 6. Interrupt Stacking Order

serviced. The minimum time between pulses is a function of the length of the interrupt service. Once a pulse occurs, the next pulse should not occur until an RTI occurs. This time (t_{ILIL}) is obtained by adding 21 instruction cycles to the total number of cycles it takes to complete the service routine (not including the RTI instruction). The second method shows many interrupt lines "wire-ORed" to form the interrupts at the processor. If the interrupt line remains low after servicing an interrupt, then the next interrupt is recognized.

NOTE

The internal interrupt latch is cleared in the first part of the interrupt service routine; therefore, one external interrupt pulse could be latched and serviced as soon as the I bit is cleared.

SOFTWARE INTERRUPT (SWI)

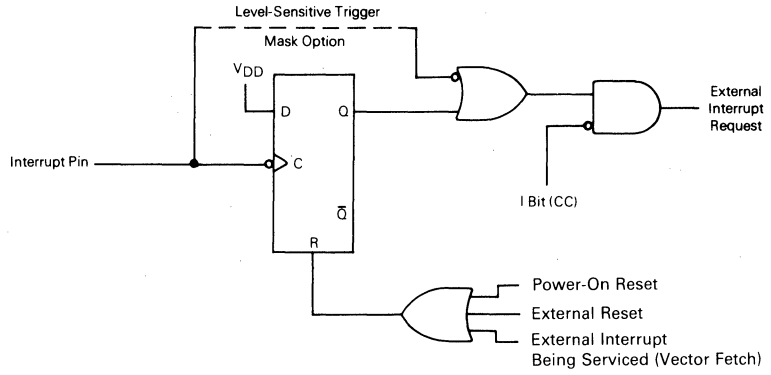
The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. If the I bit is zero, SWI executes after the other interrupts. The SWI operation is similar to the hardware interrupts. The interrupt service routine address is specified by the contents of memory locations $1FFC and $1FFD.

SCI INTERRUPTS

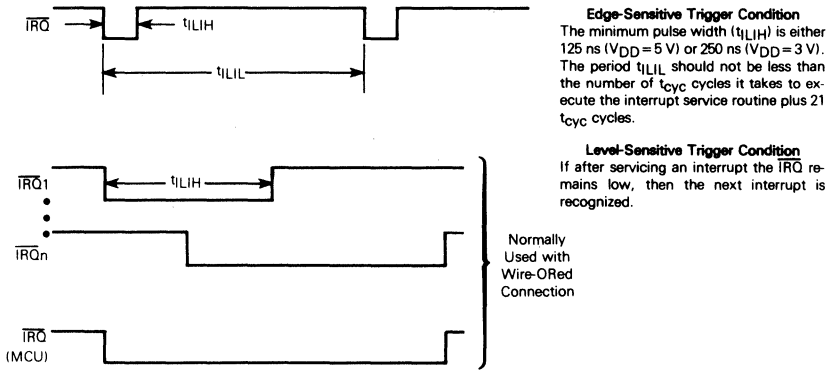
An interrupt in the SCI occurs when one of the interrupt flag bits in the serial communications status register is set, provided the I bit in the CCR is clear and the enable bit in the serial communications control register 2 is set. Software in the serial interrupt service routine must determine the cause and priority of the SCI interrupt by examining the interrupt flags and status bits in the SCI status register.

SPI INTERRUPTS

An interrupt in the SPI occurs when one of the interrupt flag bits in the serial peripheral status register is set, provided the I bit in the CCR is clear and the enable bit



(a) Interrupt Internal Function Diagram



(b) Interrupt Mode Diagram

Figure 8. External Interrupt

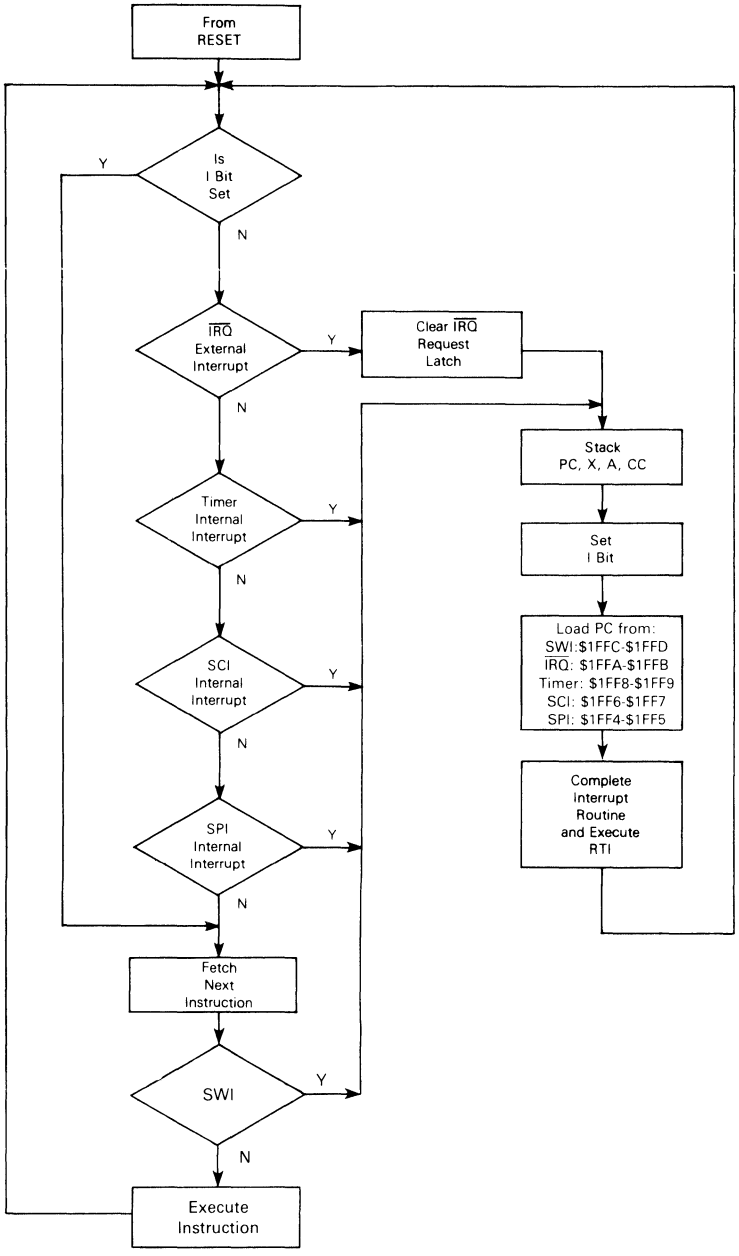


Figure 7. Reset and Interrupt Processing Flowchart

in the serial peripheral control register is set. Software in the serial peripheral interrupt service routine must determine the cause and priority of the SPI interrupt by examining the interrupt flag bits in the SPI status register.

LOW-POWER MODES

STOP

The STOP instruction places the MCU in its lowest power consumption mode. In the STOP mode, the internal oscillator is turned off, halting all internal processing including timer, SCI, and SPI operation (refer to Figure 9).

During the STOP mode, the TCR bits are altered to remove any pending timer interrupt request and to disable any further timer interrupts. The timer prescaler is cleared. The I bit in the CCR is cleared to enable external interrupts. All other registers and memory remain unaltered. All input/output lines remain unchanged. The processor can be brought out of the STOP mode only by an external interrupt or reset.

SCI during STOP Mode

When the MCU enters the STOP mode, the baud rate generator stops, halting all SCI activity. If the STOP instruction is executed during a transmitter transfer, that transfer is halted. If a low input to the $\overline{IRQ}$ pin is used to exit STOP mode, the transfer resumes. If the SCI receiver is receiving data and the STOP mode is entered, received data sampling stops because the baud rate generator stops, and all subsequent data is lost. For these reasons, all SCI transfers should be in the idle state when the STOP instruction is executed.

SPI during Stop Mode

When the MCU enters the STOP mode, the baud rate generator stops, terminating all master mode SPI operations. If the STOP instruction is executed during an SPI transfer, that transfer halts until the MCU exits the STOP mode by a low signal on the $\overline{IRQ}$ pin. If reset is used to exit the STOP mode, then the SPI control and status bits are cleared, and the SPI is disabled. If the MCU is in the slave mode when the STOP instruction is executed, the slave SPI continues to operate and can still accept data and clock information in addition to transmitting its own data back to a master device.

At the end of a possible transmission with a slave SPI in the STOP mode, no flags are set until a low on the $\overline{IRQ}$ pin wakes up the MCU. Caution should be observed when operating the SPI as a slave during the STOP mode because the protective circuitry (WCOL, MODF, etc.) is inactive.

WAIT

The WAIT instruction places the MCU in a low-power consumption mode, but the WAIT mode consumes more power than the STOP mode. All CPU action is suspended, but the timer, SCI, and SPI remain active (refer to Figure 10). An interrupt from the timer, SCI, or SPI can cause the MCU to exit the WAIT mode.

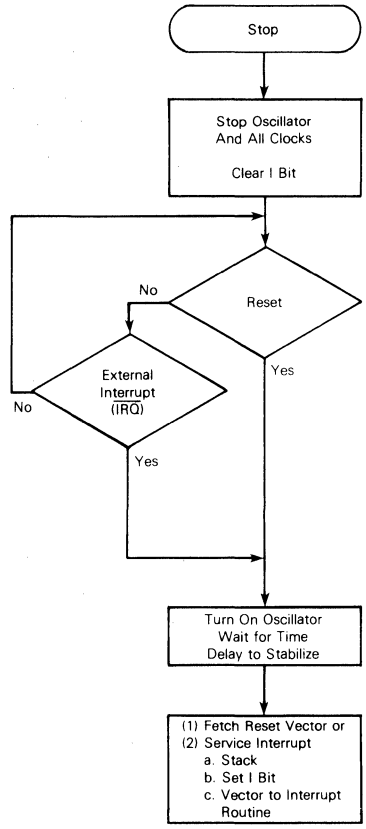


Figure 9. STOP Function Flowchart

During the WAIT mode, the I bit in the CCR is cleared to enable interrupts. All other registers, memory, and input/output lines remain in their previous state. The timer may be enabled to allow a periodic exit from the WAIT mode.

DATA RETENTION MODE

The contents of RAM and CPU registers are retained at supply voltages as low as 2.0 Vdc. This is called the data retention mode where the data is held, but the device is not guaranteed to operate. The MCU should be in $\overline{RESET}$ during data retention mode.

TIMER

The timer consists of a 16-bit, software-programmable counter driven by a fixed divide-by-four prescaler. This timer can be used for many purposes, including input waveform measurements while simultaneously generating an output waveform. Pulse widths can vary from

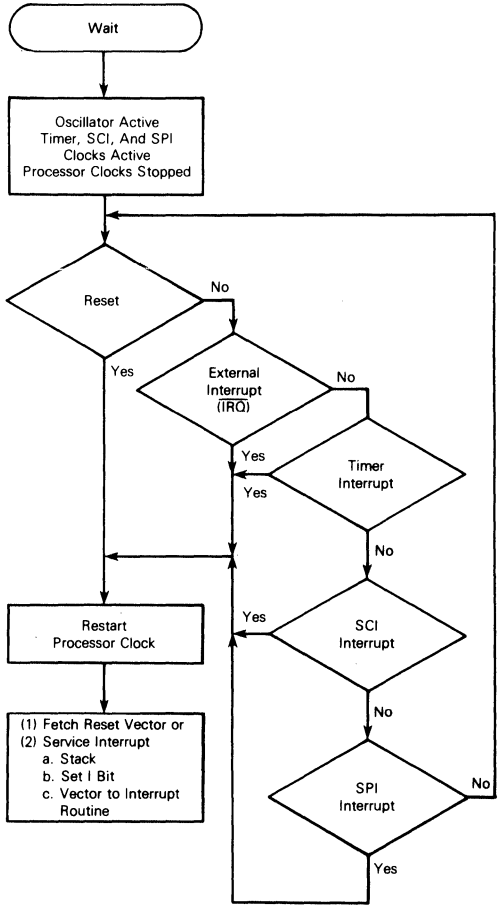


Figure 10. WAIT Function Flowchart

several microseconds to many seconds. Refer to Figure 11 for a timer block diagram.

Because the timer has a 16-bit architecture, each specific functional segment (capability) is represented by two registers. These registers contain the high and low byte of that functional segment. Generally, accessing the low byte of a specific timer function allows full control of that function; however, an access of the high byte inhibits that specific timer function until the low byte is also accessed.

NOTE

The I bit in the CCR should be set while manipulating both the high and low byte register of a specific timer function to ensure that an interrupt does not occur.

COUNTER

The key element in the programmable timer is a 16-bit, free-running counter or counter register, preceded by a prescaler that divides the internal processor clock by four. The prescaler gives the timer a resolution of 2.0 microseconds if the internal bus clock is 2.0 MHz. The counter is incremented during the low portion of the internal bus clock. Software can read the counter at any time without affecting its value.

The double-byte, free-running counter can be read from either of two locations, $18–$19 (counter register) or $1A–$1B (counter alternate register). A read from only the least significant byte (LSB) of the free-running counter ($19, $1B) receives the count value at the time of the read. If a read of the free-running counter or counter alternate register first addresses the most significant byte (MSB)

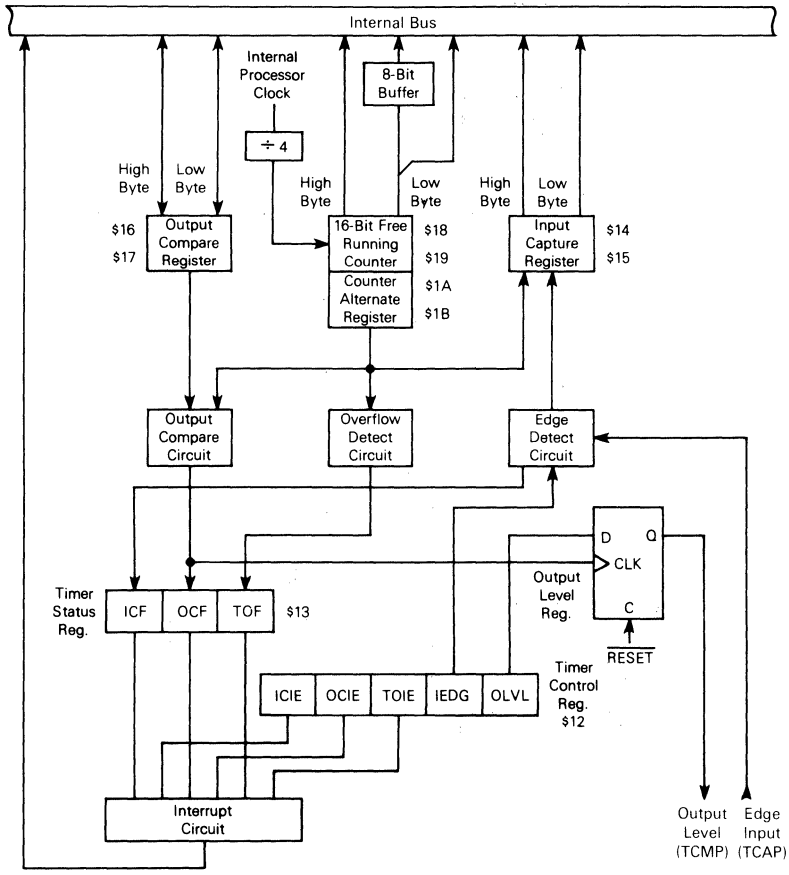


Figure 11. Timer Block Diagram

($18, $1A), the LSB ($19, $1B) is transferred to a buffer. This buffer value remains fixed after the first MSB read, even if the user reads the MSB several times. This buffer is accessed when reading the free-running counter or counter alternate register LSB ($19 or $1B) and, thus, completes a read sequence of the total counter value. In reading either the free-running counter or counter alternate register, if the MSB is read, the LSB must also be read to complete the sequence.

The counter alternate register differs from the counter register in one respect: a read of the counter register MSB can clear the timer overflow flag (TOF). Therefore, the counter alternate register can be read at any time without the possibility of missing timer overflow interrupts due to clearing of the TOF.

The free-running counter is configured to $FFFC during reset and is always a read-only register. During a power-on reset, the counter is also preset to $FFFC and begins running after the oscillator start-up delay. Because the free-running counter is 16 bits preceded by a fixed divide-by-four prescaler, the value in the free-running counter repeats every 262,144 internal bus clock cycles. When the counter rolls over from $FFFF to $0000, the TOF bit is set. An interrupt can also be enabled when counter rollover occurs by setting its interrupt enable bit (TOIE).

OUTPUT COMPARE REGISTER

The 16-bit output compare register is made up of two 8-bit registers at locations $16 (MSB) and $17 (LSB). The

output compare register is used for several purposes, such as indicating when a period of time has elapsed. All bits are readable and writable and are not altered by the timer hardware or reset. If the compare function is not needed, the two bytes of the output compare register can be used as storage locations.

The output compare register contents are compared with the contents of the free-running counter continually, and if a match is found, the corresponding output compare flag (OCF) bit is set and the corresponding output level (OLCL) bit is clocked to an output level register. The output compare register values and the output level bit should be changed after each successful comparison to establish a new elapsed timeout. An interrupt can also accompany a successful output compare provided the corresponding interrupt enable bit (OCIE) is set.

After a processor write cycle to the output compare register containing the MSB ($16), the output compare function is inhibited until the LSB ($17) is also written. The user must write both bytes (locations) if the MSB is written first. A write made only to the LSB ($17) will not inhibit the compare function. The free-running counter is updated every four internal bus clock cycles. The minimum time required to update the output compare register is a function of the program rather than the internal hardware.

The processor can write to either byte of the output compare register without affecting the other byte. The output level (OLVL) bit is clocked to the output level register regardless of whether the output compare flag (OCF) is set or clear.

INPUT CAPTURE REGISTER

Two 8-bit registers, which make up the 16-bit input capture register, are read-only and are used to latch the value of the free-running counter after the corresponding input capture edge detector senses a defined transition. The level transition which triggers the counter transfer is defined by the corresponding input edge bit (IEDG). Reset does not affect the contents of the input capture register.

The result obtained by an input capture will be one more than the value of the free-running counter on the rising edge of the internal bus clock preceding the external transition. This delay is required for internal synchronization. Resolution is one count of the free-running counter, which is four internal bus clock cycles.

The free-running counter contents are transferred to the input capture register on each proper signal transition regardless of whether the input capture flag (ICF) is set or clear. The input capture register always contains the free-running counter value that corresponds to the most recent input capture.

After a read of the input capture register ($14) MSB, the counter transfer is inhibited until the LSB ($15) is also read. This characteristic causes the time used in the input capture software routine and its interaction with the main program to determine the minimum pulse period.

A read of the input capture register LSB ($15) does not inhibit the free-running counter transfer since they occur on opposite edges of the internal bus clock.

TIMER CONTROL REGISTER (TCR) $12

The TCR is a read/write register containing five control bits. Three bits control interrupts associated with the timer status register flags ICF, OCF, and TOF.

7	6	5	4	3	2	1	0
ICIE	OCIE	TOIE	0	0	0	IEDG	OLVL
RESET:							
0	0	0	0	0	0	U	0

ICIE — Input Capture Interrupt Enable
- 1 = Interrupt enabled
- 0 = Interrupt disabled

OCIE — Output Compare Interrupt Enable
- 1 = Interrupt enabled
- 0 = Interrupt disabled

TOIE — Timer Overflow Interrupt Enable
- 1 = Interrupt enabled
- 0 = Interrupt disabled

IEDG — Input Edge

Value of input edge determines which level transition on TCAP pin will trigger free-running counter transfer to the input capture register
- 1 = Positive edge
- 0 = Negative edge

Reset does not affect te IEDG bit (U = unaffected).

OLVL — Output Level

Value of output level is clocked into output level register by the next successful output compare and will appear on the TCMP pin
- 1 = High output
- 0 = Low output

Bits 2, 3, and 4 — Not used

Always read zero

TIMER STATUS REGISTER (TSR) $13

The TSR is a read-only register containing three status flag bits.

7	6	5	4	3	2	1	0
ICF	OCF	TOF	0	0	0	0	0
RESET:							
U	U	U	0	0	0	0	0

ICF — Input Capture Flag
- 1 = Flag set when selected polarity edge is sensed by input capture edge detector
- 0 = Flag cleared when TSR and input capture low register ($15) are accessed

OCF — Output Compare Flag
- 1 = Flag set when output compare register contents match the free-running counter contents
- 0 = Flag cleared when TSR and output compare low register ($17) are accessed

TOF — Timer Overflow Flag
- 1 = Flag set when free-running counter transition from $FFFF to $0000 occurs
- 0 = Flag cleared when TSR and counter low register ($19) are accessed

Bits 0–4 — Not used

Always read zero

Accessing the timer status register satisfies the first condition required to clear status bits. The remaining step is to access the register corresponding to the status bit.

A problem can occur when using the timer overflow function and reading the free-running counter at random times to measure an elapsed time. Without incorporating the proper precautions into software, the timer overflow flag could unintentionally be cleared if:

1) The timer status register is read or written when TOF is set, and
2) The LSB of the free-running counter is read but not for the purpose of servicing the flag.

The counter alternate register at address $1A and $1B contains the same value as the free-running counter (at address $18 and $19); therefore, this alternate register can be read at any time without affecting the timer overflow flag in the timer status register.

TIMER DURING WAIT MODE

The CPU clock halts during the WAIT mode, but the timer remains active. An interrupt from the timer causes the processor to exit the WAIT mode.

TIMER DURING STOP MODE

In the STOP mode, the timer stops counting and holds the last count value if STOP is exited by an interrupt. If RESET is used, the counter is forced to $FFFC. During STOP, if at least one valid input capture edge occurs at the TCAP pin, the input capture detect circuit is armed. This does not set any timer flags nor wake up the MCU, but when the MCU does wake up, there is an active input capture flag and data from the first valid edge that occurred during the STOP mode. If RESET is used to exit STOP mode, then no input capture flag or data remains, even if a valid input capture edge occurred.

SERIAL COMMUNICATIONS INTERFACE

A full-duplex asynchronous SCI is provided with a standard NRZ format and a variety of baud rates. The SCI transmitter and receiver are functionally independent but use the same data format and baud rate. The terms baud and bit rate are used synonymously in the following description.

SCI TWO-WIRE SYSTEM FEATURES

- Standard NRZ (mark/space) format
- Advanced error detection method includes noise detection for noise duration of up to one-sixteenth bit time
- Full-duplex operation (simultaneous transmit and receive)
- Software programmable for one of 32 different baud rates
- Software-selectable word length (eight- or nine-bit words)
- Separate transmitter and receiver enable bits
- SCI may be interrupt driven
- Four separate interrupt conditions

SCI RECEIVER FEATURES

- Receiver wake-up function (idle or address bit)
- Idle line detect
- Framing error detect
- Noise detect
- Overrun detect
- Receiver data register full flag

SCI TRANSMITTER FEATURES

- Transmit data register empty flag
- Transmit complete flag
- Break send

Any SCI two-wire system requires receive data in (RDI) and transmit data out (TDO).

DATA FORMAT

Receive data in (RDI) or transmit data out (TDO) is the serial data presented between the internal data bus and the output pin (TDO) and between the input pin (RDI) and the internal data bus. Data format is as shown for the NRZ in Figure 12.

WAKE-UP FEATURE

In a typical multiprocessor configuration, the software protocol will usually identify the addressee(s) at the beginning of the message. To permit uninterested MPUs to ignore the remainder of the message, a wake-up feature is included, whereby all further SCI receiver flag (and interrupt) processing can be inhibited until its data line returns to the idle state. An SCI receiver is re-enabled by

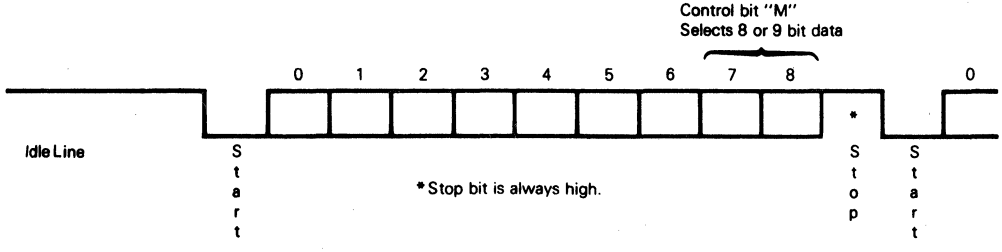


Figure 12. Data Format

an idle string of at least ten (or eleven) consecutive ones. Software for the transmitter must provide for the required idle string between consecutive messages and prevent it from occurring within messages.

A second wake-up method is available in which sleeping SCI receivers can be awakened by a logic one in the high-order bit of a received character.

RECEIVE DATA IN

Receive data in (RDI) is the serial data which is presented from the input pin via the SCI to the receive data register (RDR). While waiting for a start bit, the receiver samples the input at a rate 16 times higher than the set baud rate. This increased rate is referred to as the RT rate. When the input (idle) line is detected low, it is tested for three more sample times. If at least two of these three samples detect a logic low, a valid start bit is assumed to be detected. If in two or more samples, a logic high is detected, the line is assumed to be idle. The receive clock generator is controlled by the baud rate register (see Figure 13); however, the SCI is synchronized by the start bit independent of the transmitter. Once a valid start bit is detected, the start bit, each data bit, and the stop bit are each sampled three times. The value of the bit is determined by voting logic, which takes the value of a majority of samples. A noise flag is set when all three samples on a valid start bit, data bit, or stop bit do not agree. A noise flag is also set when the start verification samples do not agree.

START BIT DETECTION FOLLOWING A FRAMING ERROR

If there has been a framing error (FE) without detection of a break (10 zeros for 8-bit format or 11 zeros for a 9-bit format), the circuit continues to operate as if there actually were a stop bit, and the start edge will be placed artificially. The last bit received in the data shift register is inverted to a logic one, and the three logic-one start qualifiers are forced into the sample shift register during the interval when detection of a start bit is anticipated; therefore, the start bit will be accepted no sooner than it is anticipated.

If the receiver detects that a break (RDRF = 1, FE = 1, receiver data register = $00) produced the framing error, the start bit will not be artificially induced, and the receiver must actually receive a logic one before start.

TRANSMIT DATA OUT

Transmit data out (TDO) is the serial data presented from the transmit data register (TDR) via the SCI to the output pin. The transmitter generates a bit time by using a derivative of the RT clock, producing a transmission rate equal to one-sixteenth that of the receiver sample clock.

FUNCTIONAL DESCRIPTION

A block diagram of the SCI is shown in Figure 13. The user has option bits in the serial communications control register 1 (SCCR1) to determine the SCI wake-up method and data word length. Serial communications control register 2 (SCCR2) provides control bits that individually enable/disable the transmitter or receiver, enable system interrupts, and provide wake-up enable, and send break code bits. The baud rate register bits allow the user to select different baud rates, which are used as the rate control for the transmitter and receiver.

Data transmission is initiated by a write to the serial communications data register (SCDAT). Provided the transmitter is enabled, data stored in the SCDAT is transferred to the transmit data shift register. This data transfer sets the SCI status register (SCSR) transmit data register empty (TDRE) bit and generates an interrupt if the transmit interrupt is enabled. Data transfer to the transmit data shift register is synchronized with the bit rate clock. All data is transmitted LSB first. Upon completion of data transmission, the transmission complete (TC) bit is set (provided no pending data, preamble, or break code is sent), and an interrupt is generated if the transmit complete interrupt is enabled. If the transmitter is disabled, and the data, preamble, or break code has been sent, the TC bit will also be set, which will also generate an interrupt if the TCIE bit is set. If the transmitter is disabled in the middle of a transmission, that character will be completed before the transmitter gives up control of the TDO pin.

When the SCDAT is read, it contains the last data byte received, provided that the receiver is enabled. The SCSR receive data register full (RDRF) bit is set to indicate that a data byte is transferred from the input serial shift register to the SCDAT, which can cause an interrupt if the receiver interrupt is enabled. Data transfer from the input serial shift register to the SCDAT is synchronized by the receiver bit rate clock. The SCSR overrun (OR), noise flag (NF), or FE bits are set if data reception errors occur.

An idle line interrupt is generated if the idle line interrupt is enabled and the SCSR IDLE bit (which detects idle line transmission) is set. This allows a receiver that is not in the wake-up mode to detect the end of a message, the preamble of a new message, or to resynchronize with the transmitter. A valid character must be received before the idle line condition for the IDLE bit to be set and for an idle line interrupt to be generated.

REGISTERS

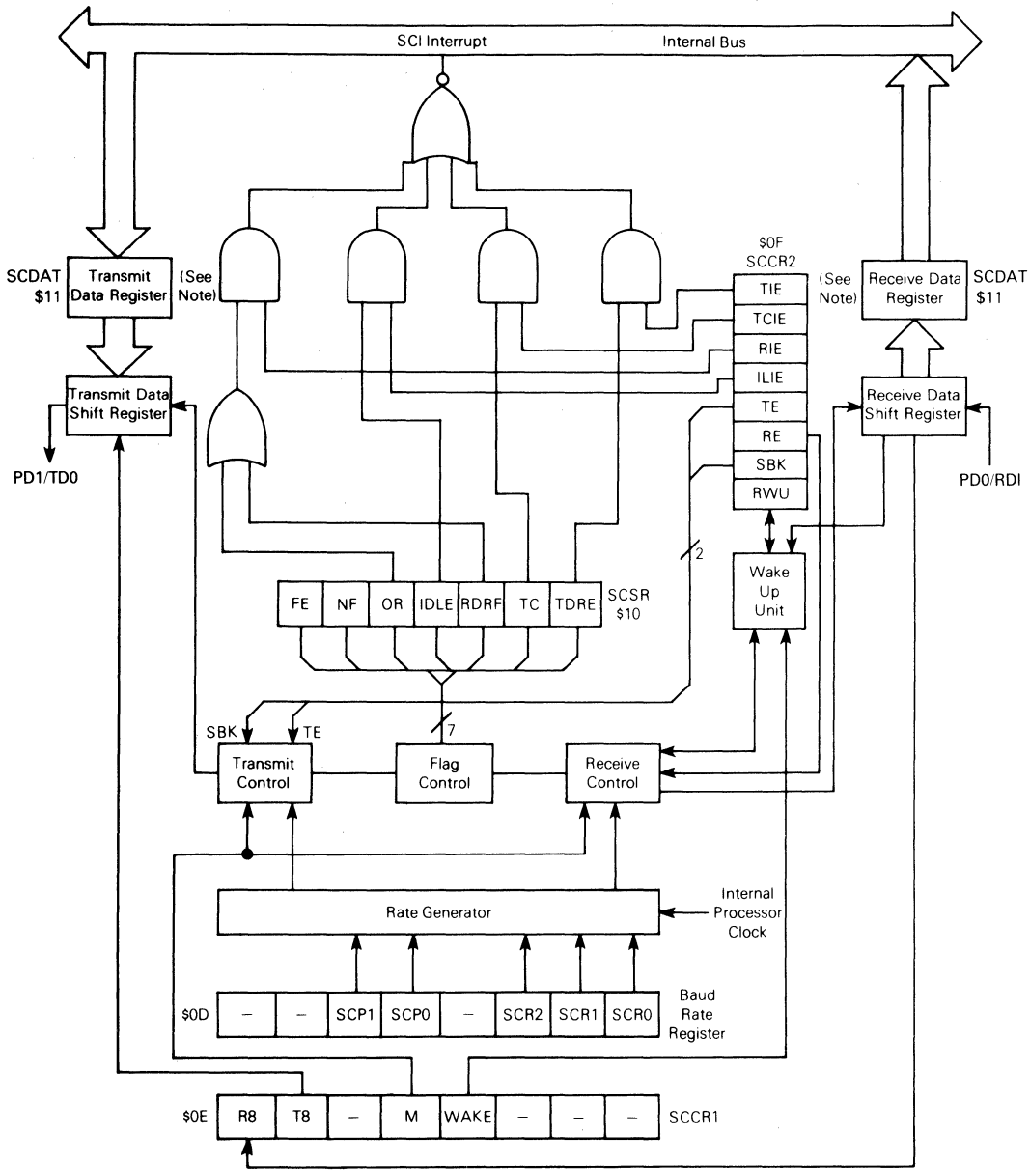
There are five registers used in the SCI; the internal configuration of these registers is discussed in the following paragraphs.

Serial Communications Data Register (SCDAT) $11

The SCDAT is a read/write register used to receive and transmit SCI data.

7	6	5	4	3	2	1	0
SCD7	SCD6	SCD5	SCD4	SCD3	SCD2	SCD1	SCD0
RESET:							
U	U	U	U	U	U	U	U

As shown in Figure 13, SCDAT functions as two separate registers. The transmit data register (TDR) provides the parallel interface from the internal data bus to the transmit shift register. The receive data register (RDR) provides the interface from the receive shift register to the internal data bus.



NOTE: The Serial Communications Data Register (SCDAT) is controlled by the internal R/W signal. It is the transmit data register when written and receive data register when read.

Figure 13. SCI Block Diagram

Serial Communications Control Register 1 (SCCR1) $0E

The SCCR1 provides control bits that determine word length and select the wake-up method.

7	6	5	4	3	2	1	0
R8	T8	—	M	WAKE	—	—	—
RESET:							
U	U	—	U	U	—	—	—

R8 — Receive Data Bit 8
R8 bit provides storage location for the ninth bit in the receive data byte (if M = 1).
T8 — Transmit Data Bit 8
T8 bit provides storage location for the ninth bit in the transmit data byte (if M = 1).
M — SCI Character Word Length
1 = one start bit, nine data bits, one stop bit
0 = one start bit, eight data bits, one stop bit
WAKE — Wake-Up Select
Wake bit selects the receiver wake-up method.
1 = Address bit (most significant bit)
0 = Idle line condition
Bits 0–2, and 5 — Not used
Can read either one or zero

The address bit is dependent on both the wake-bit and the M-bit level. Additionally, the receiver does not use the wake-up feature unless the RWU control bit in SCCR2 is set.

Wake	M	Receiver Wake-Up
0	X	Detection of an idle line allows the next data byte received to cause the receive data register to fill and produce an RDRF flag.
1	0	Detection of a received one in the eighth data bit allows an RDRF flag and associated error flags.
1	1	Detection of a received one in the ninth data bit allows an RDRF flag and associated error flags.

Serial Communications Control Register 2 (SCCR2) $0F

The SCCR2 provides control of individual SCI functions such as interrupts, transmit/receive enabling, receiver wake-up, and break code.

7	6	5	4	3	2	1	0
TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
RESET:							
0	0	0	0	0	0	0	0

TIE — Transmit Interrupt Enable
1 = SCI interrupt enabled
0 = TDRE interrupt disabled
TCIE — Transmit Complete Interrupt Enable
1 = SCI interrupt enabled
0 = TC interrupt disabled
RIE — Receive Interrupt Enable
1 = SCI interrupt enabled
0 = RDRF and OR interrupts disabled
ILIE — Idle Line Interrupt Enable
1 = SCI interrupt enabled
0 = Idle interrupt disabled
TE — Transmit Enable
1 = Transmit shift register output is applied to the TD0 line. Depending upon the SCCR1 M bit, a preamble of 10 (M = 0) or 11 (M = 1) consecutive ones is transmitted.
0 = Transmitter disabled after last byte is loaded in the SCDAT and TDRE is set. After last byte is transmitted, TD0 line becomes a high-impedance line.
RE — Receive Enable
1 = Receiver shift register input is applied to the RDI line.
0 = Receiver disabled and RDRF, IDLE, OR, NF, and FE status bits are inhibited.
RWU — Receiver Wake-Up
1 = Places receiver in sleep mode and enables wake-up function
0 = Wake-up function disabled after receiving data word with MSB set (if WAKE = 1)
Wake-up function also disabled after receiving 10 (M = 0) or 11 (M = 1) consecutive ones (if WAKE = 0)
SBK — Send Break
1 = Transmitter continually sends blocks of zeros (sets of 10 or 11) until cleared. Upon completion of break code, transmitter sends one high bit for recognition of valid start bit.
0 = Transmitter sends 10 (M = 0) or 11 (M = 1) zeros then reverts to an idle state or continues sending data. If transmitter is empty and idle, setting and clearing the SBK bit may queue up to two character times of break because the first break transfers immediately to the shift register, and the second is queued into the parallel transmit buffer.

Serial Communications Status Register (SCSR) $10

The SCSR provides inputs to the SCI interrupt logic circuits. Noise flag and framing error bits are also contained in the SCSR.

7	6	5	4	3	2	1	0
TDRE	TC	RDRF	IDLE	OR	NF	FE	—
RESET:							
1	1	0	0	0	0	0	—

TDRE — Transmit Data Register (TDR) Empty
1 = TDR contents transferred to the transmit data shift register
0 = TDR still contains data. TDRE is cleared by reading the SCSR (with TDRE = 1), followed by a write to the TDR.
TC — Transmit Complete
1 = Indicates end of data frame, preamble, or break condition has occurred
0 = TC bit cleared by reading the SCSR (with TC = 1), followed by a write to the TDR

RDRF — Receive Data Register (RDR) Full
1 = Receive data shift register contents transferred to the RDR
0 = Receive data shift register transfer did not occur. RDRF is cleared by reading the SCSR (with RDRF = 1) followed by a read of the RDR

IDLE — Idle Line Detect
1 = Indicates receiver has detected an idle line
0 = IDLE is cleared by reading the SCSR (with IDLE = 1), followed by a read of the RDR. Once IDLE is cleared, IDLE cannot be set until RDI line becomes active and idle again.

OR — Overrun Error
1 = Indicates receive data shift register data is sent to a full RDR (RDRF = 1). Data causing the overrun is lost, and RDR data is not disturbed.
0 = OR is cleared by reading the SCSR (with OR = 1), followed by a read of the RDR.

NF — Noise Flag
1 = Indicates noise is present on the receive bits, including the start and stop bits. NF is not set until RDRF = 1.
0 = NF is cleared by reading the SCSR (with NF = 1), followed by a read of the RDR.

FE — Framing Error
1 = Indicates stop bit not detected in received data character. FE is set the same time RDRF is set. If received byte causes both framing and overrun errors, processor will only recognize the overrun error. Further data transfer into the RDR is inhibited until FE is cleared.
0 = NF is cleared by reading the SCSR (with FE = 1), followed by a read of the RDR.

Bit 0 — Not used
Can read either one or zero

Baud Rate Register $0D

The baud rate register is used to select the SCI transmitter and receiver baud rate. SCP0 and SCP1 prescaler bits are used in conjunction with the SCR0 through SCR2 baud rate bits to provide multiple baud rate combinations for a given crystal frequency. Bits 3, 6, and 7 always read zero.

7	6	5	4	3	2	1	0
—	—	SCP1	SCP0	—	SCR2	SCR1	SCR0
RESET:							
—	—	0	0	—	U	U	U

SCP0 — SCI Prescaler Bit 0
SCP1 — SCI Prescaler Bit 1
Two prescaler bits are used to increase the range of standard baud rates controlled by the SCR0–SCR2 bits. Prescaler internal processor clock division versus bit levels are listed in Table 2.

SCR0 — SCI Baud Rate Bit 0
SCR1 — SCI Baud Rate Bit 1
SCR2 — SCI Baud Rate Bit 2
Three baud rate bits are used to select the baud rates of the SCI transmitter and SCI receiver. Baud rates versus bit levels are listed in Table 3.

Tables 3 and 4 tabulate the divide chain used to obtain the baud rate clock (transmit clock). The actual divider chain is controlled by the combined SCP0-SCP1 and SCR0-SCR2 bits in the baud rate register. All divided frequencies shown in Table 3 represent the final baud rate resulting from the internal processor clock division shown in the divided-by column only (prescaler division only). Table 4 lists the prescaler output divided by the action of the SCI select bits (SCR0-SCR2). For example, assume that a 9600-Hz baud rate is required with a 2.4576-MHz

Table 3. Prescaler Highest Baud Rate Frequency Output

SCP Bit		Clock* Divided By	Crystal Frequency MHz				
1	0		4.194304	4.0	2.4576	2.0	1.8432
0	0	1	131.072 kHz	125.000 kHz	76.80 kHz	62.50 kHz	57.60 kHz
0	1	3	43.691 kHz	41.666 kHz	25.60 kHz	20.833 kHz	19.20 kHz
1	0	4	32.768 kHz	31.250 kHz	19.20 kHz	15.625 kHz	14.40 kHz
1	1	13	10.082 kHz	9600 Hz	5.907 kHz	4800 Hz	4430 Hz

*Refers to the internal processor clock.

NOTE: The divided frequencies shown in Table 3 represent baud rates which are the highest transmit baud rate (Tx) that can be obtained by a specific crystal frequency and only using the prescaler division. Lower baud rates may be obtained by providing a further division using the SCI rate select bits as shown below for some representative prescaler outputs.

Table 4. Transmit Baud Rate Output for a Given Prescaler Output

SCR Bits			Divided By	Representative Highest Prescaler Baud Rate Output				
2	1	0		131.072 kHz	32.768 kHz	76.80 kHz	19.20 kHz	9600 Hz
0	0	0	1	131.072 kHz	32.768 kHz	76.80 kHz	19.20 kHz	9600 Hz
0	0	1	2	65.536 kHz	16.384 kHz	38.40 kHz	9600 Hz	4800 Hz
0	1	0	4	32.768 kHz	8.192 kHz	19.20 kHz	4800 Hz	2400 Hz
0	1	1	8	16.384 kHz	4.096 kHz	9600 Hz	2400 Hz	1200 Hz
1	0	0	16	8.192 kHz	2.048 kHz	4800 Hz	1200 Hz	600 Hz
1	0	1	32	4.096 kHz	1.024 kHz	2400 Hz	600 Hz	300 Hz
1	1	0	64	2.048 kHz	512 Hz	1200 Hz	300 Hz	150 Hz
1	1	1	128	1.024 kHz	256 Hz	600 Hz	150 Hz	75 Hz

NOTE: Table 4 illustrates how the SCI select bits can be used to provide lower transmitter baud rates by further dividing the prescaler output frequency. The five examples are only representative samples. In all cases, the baud rates shown are transmit baud rates (transmit clock), and the receive clock is 16 times higher in frequency than the actual baud rate.

external crystal. In this case, the prescaler bits (SCP0-SCP1) could be configured as a divide-by-one or a divide-by-four. If a divide-by-four prescaler is used, then the SCR0-SCR2 bits must be configured as a divide-by-two. Using the same crystal, the 9600 baud rate can be obtained with a prescaler divide-by-one and the SCR0-SCR2 bits configured for a divide-by-eight.

SERIAL PERIPHERAL INTERFACE

The serial peripheral interface (SPI) is an interface built into the MCU which allows several MCUs or MCUs plus peripherals to be interconnected within the same black box. In the SPI format, the clock is not included in the data stream and must be furnished as a separate signal. An SPI system may consist of one master MCU and several slaves (Figure 14) or MCUs that can be either masters or slaves.

Features:

- Full-duplex, three-wire synchronous transfers
- Master or slave operation
- 1.05 MHz (maximum) master bit frequency
- 2.1 MHz (maximum) slave bit frequency
- Four programmable master bit rates
- Programmable clock polarity and phase
- End-of-transmission interrupt flag
- Write collision flag protection
- Master-master mode fault protection capability

SIGNAL DESCRIPTION

The four basic signals (MOSI, MISO, SCK, and $\overline{SS}$) are described in the following paragraphs. Each signal function is described for both master and slave mode.

Master Out, Slave In

The master out, slave in (MOSI) line is configured as an output in a master device and as an input in a slave device. The MOSI line is one of two lines that transfer serial data in one direction with the most significant bit sent first.

Master In, Slave Out

The master in, slave out (MISO) line is configured as an input in a master device and as an output in a slave device. The MISO is one of two lines that transfer serial data in one direction with the most significant bit sent first. The MISO line of a slave device is placed in a high-impedance state if slave is not selected ($\overline{SS} = 1$).

Serial Clock

The serial clock (SCK) is used to synchronize both data in and out of a device via the MOSI and MISO lines. The master and slave devices can exchange a byte of information during a sequence of eight clock cycles. Since SCK is generated by the master device, this line becomes an input on a slave device.

As shown in Figure 15, four possible timing relationships may be chosen by using control bits CPOL and CPHA in the serial peripheral control register (SPCR). Both master and slave devices must operate with the same timing.

Two bits (SPR0 and SPR1) in the SPCR of the master device select the clock rate. In a slave device, SPR0 and SPR1 have no effect on SPI operation.

Slave Select

The slave select ($\overline{SS}$) input line selects a slave device. The $\overline{SS}$ line must be low prior to data transactions and must stay low for the duration of the transaction. The $\overline{SS}$ line on the master must be tied high; if the $\overline{SS}$ line goes low, a mode fault error flag (MODF) is set in the serial peripheral status register (SPSR).

When CPHA = 0, the shift clock is the OR of $\overline{SS}$ with SCK. In this clock phase mode, $\overline{SS}$ must go high between successive characters in an SPI message. When CPHA = 1, $\overline{SS}$ must go high between successive characters in an

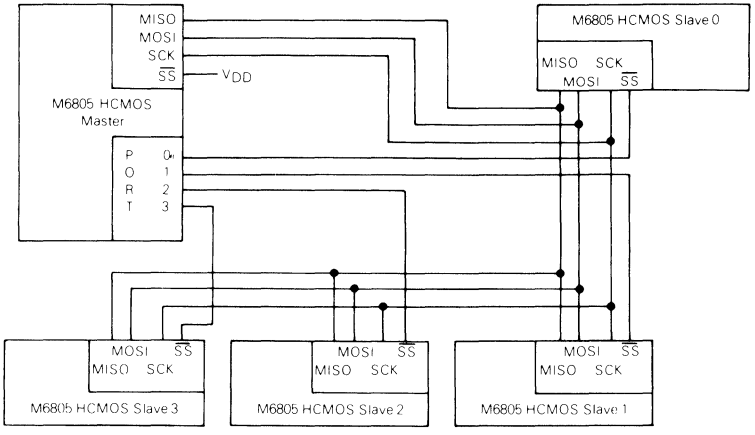


Figure 14. Master-Slave System Configuration

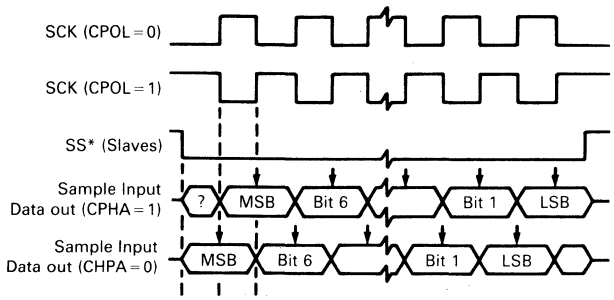


Figure 15. Data Clock Timing Diagram

SPI message. When CPHA = 1, $\overline{SS}$ may be left low for several SPI characters. In cases where there is only one SPI slave MCU, the slave MCU $\overline{SS}$ line could be tied to V_{SS} as long as CPHA = 1 clock modes are used.

3

FUNCTIONAL DESCRIPTION

A block diagram of the SPI is shown in Figure 16. In a master configuration, the CPU sends a signal to the master start logic, which originates an SPI clock (SCK) based on the internal processor clock. As a master device, data is parallel loaded into the 8-bit shift register from the internal bus during a write cycle and then serially shifted via the MOSI pin to the slave devices. During a read cycle, data is applied serially from a slave device via the MISO pin to the 8-bit shift register. Data is then parallel transferred to the read buffer and made available to the internal data bus during a CPU read cycle.

In a slave configuration, the slave start logic receives a logic low at the $\overline{SS}$ pin and a clock input at the SCK pin. This synchronizes the slave with the master. Data from the master is received serially at the slave MOSI pin and shifted into the 8-bit shift register for a parallel transfer to the read buffer. During a write cycle, data is parallel loaded into the 8-bit shift register from the internal data bus, awaiting the clocks from the master to shift out serially to the MISO pin and then to the master device.

Figure 17 illustrates the MOSI, MISO, SCK, and $\overline{SS}$ master-slave interconnections.

REGISTERS

There are three registers in the SPI that provide control, status, and data storage functions. These registers, the serial peripheral control register (SPCR), serial peripheral status register (SPSR), and serial peripheral data I/O register (SPDR), are described in the following paragraphs.

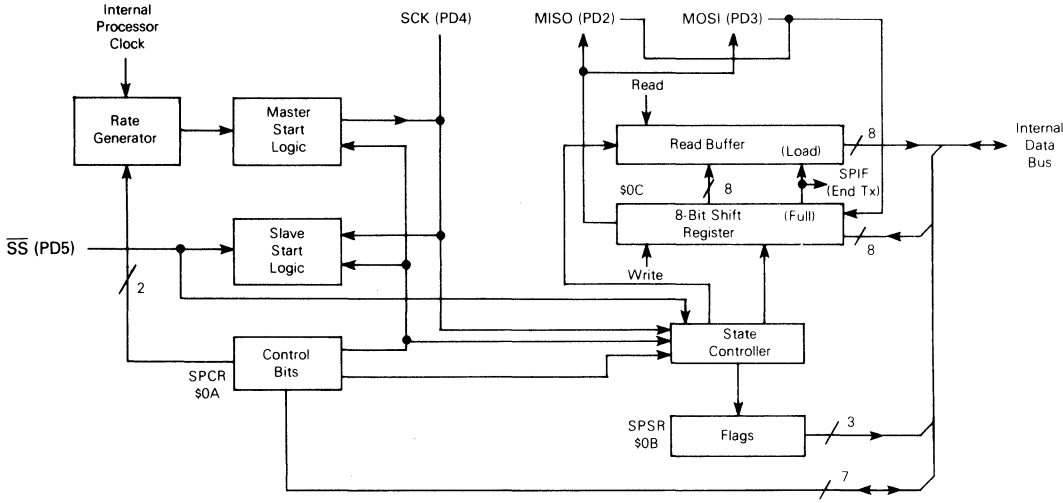


Figure 16. SPI Block Diagram

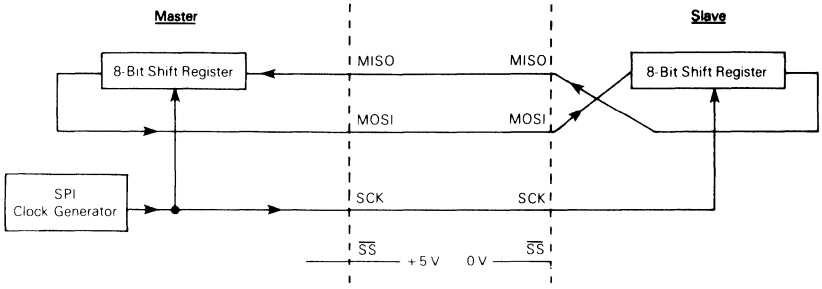


Figure 17. SPI Master-Slave Interconnections

Serial Peripheral Control Register $0A

The SPCR provides control of individual SPI functions such as interrupt and system enabling/disabling, master/slave mode select, and clock polarity/phase/rate select.

7	6	5	4	3	2	1	0
SPIE	SPE	—	MSTR	CPOL	CPHA	SPR1	SPR0
RESET:							
0	0	—	0	U	U	U	U

SPIE — Serial Peripheral Interrupt Enable
- 1 = SPI interrupt enabled
- 0 = SPI interrupt disabled

SPE — Serial Peripheral System Enable
- 1 = SPI system on
- 0 = SPI system off

MSTR — Master Mode Select
- 1 = Master mode
- 0 = Slave mode

CPOL — Clock Polarity

Clock polarity bit controls the clock value and is used in conjunction with the clock phase (CPHA) bit.
- 1 = SCK line idles high
- 0 = SCK line idles in low state

CPHA — Clock Phase

Clock phase bit along with CPOL controls the clock-data relationship between the master and slave devices. CPOL selects one of two clocking protocols.
- 1 = $\overline{SS}$ is an output enable control.
- 0 = Shift clock is the OR of SCK with $\overline{SS}$. When $\overline{SS}$ is low, first edge of SCK invokes first data sample.

SPR0, SPR1 — SPI Clock Rate Bits

Two clock rate bits are used to select one of four clock rates to be used as SCK in the master mode. In the slave mode, the two clock rate bits have no effect. Clock rate selection is shown in the following table.

Bit 5 – Not used

Can read either one or zero

SPI Clock Rate Selection

SPR1	SPR0	Internal Processor Clock Divided By
0	0	2
0	1	4
1	0	16
1	1	32

Serial Peripheral Status Register $0B

The SPSR contains three status bits.

7	6	5	4	3	2	1	0
SPIF	WCOL	—	MODF	—	—	—	—
RESET:							
0	0	—	0	—	—	—	—

SPIF — Serial Peripheral Data Transfer Flag
- 1 = Indicates data transfer completed between processor and external device. (If SPIF = 1 and SPIE = 1, SPI interrupt is enabled.)
- 0 = Clearing is accomplished by reading SPSR (with SPIF = 1) followed by SPDR access.

WCOL — Write Collision
- 1 = Indicates an attempt is made to write to SPDR while data transfer is in process.
- 0 = Clearing is accomplished by reading SPSR (with WCOL = 1), followed by SPDR access.

MODF — Mode Fault Flag
- 1 = Indicates multi-master system control conflict.
- 0 = Clearing is accomplished by reading SPSR (with MODF = 1), followed by a write to the SPCR.

Bits 0–3, and 5 — Not used

Can read either zero or one

Serial Peripheral Data I/O Register $0C

The SPDR is a read/write register used to receive and transmit SPI data.

7	6	5	4	3	2	1	0
SPD7	SPD6	SPD5	SPD4	SPD3	SPD2	SPD1	SPD0
RESET:							
U	U	U	U	U	U	U	U

A write to the SPDR places data directly into the shift register for transmission. Only a write to this register will initiate transmission/reception of another byte and will only occur in the master device. On completion of byte transmission, the SPIF status bit is set in both master and slave devices.

A read to the SPDR causes the buffer to be read. The first SPIF status bit must be cleared by the time a second data transfer from the shift register to the read buffer begins, or an overrun condition will exist. In overrun cases, the byte causing the overrun is lost.

INSTRUCTION SET

The MCU has a set of 62 basic instructions. They can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

This MCU uses all the instructions available in the M146805 CMOS Family plus one more: the unsigned multiply (MUL) instruction. This instruction allows unsigned multiplication of the contents of the accumulator (A) and the index register (X). The high-order product is then stored in the index register, and the low-order product is stored in the accumulator. A detailed definition of the MUL instruction is shown below.

Operation	X:A X*A			
Description	Multiplies the eight bits in the index register by the eight bits in the accumulator to obtain a 16-bit unsigned number in the concatenated accumulator and index register			
Condition Codes	H: Cleared I: Not affected N: Not affected Z: Not affected C: Cleared			
Source	MUL			
Form(s)	Addressing Mode Inherent	Cycles 11	Bytes 1	Opcode $42

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following instruction list.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (Twos Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST
Multiply	MUL

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two-byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Higher	BHI
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
Branch if Higher or Same	BHS
Branch if Carry Set	BCS
Branch if Lower	BLO
Branch if Not Equal	BNE
Branch if Equal	BEQ
Branch if Half Carry Clear	BHCC
Branch if Half Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Bit is Clear	BMC
Branch if Interrupt Mask Bit is Set	BMS
Branch if Interrupt Line is Low	BIL
Branch if Interrupt Line is High	BIH
Branch to Subroutine	BSR

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two-byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Higher	BHI
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
Branch if Higher or Same	BHS
Branch if Carry Set	BCS
Branch if Lower	BLO
Branch if Not Equal	BNE
Branch if Equal	BEQ
Branch if Half Carry Clear	BHCC
Branch if Half Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Bit is Clear	BMC
Branch if Interrupt Mask Bit is Set	BMS
Branch if Interrupt Line is Low	BIL
Branch if Interrupt Line is High	BIH
Branch to Subroutine	BSR

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No-Operation	NOP
Stop	STOP
Wait	WAIT

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any writable bit which resides in the first 256 bytes of the memory space where all port registers, port DDRs, timer, timer control, ROM, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear and bit test, and branch functions are all implemented with a single instruction. For test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch if Bit n is Set	BRSET n (n = 0 . . . 7)
Branch if Bit n is Clear	BRCLR n (n = 0 . . . 7)
Set Bit n	BSET n (n = 0 . . . 7)
Clear Bit n	BCLR n (n = 0 . . . 7)

OPCODE MAP SUMMARY

Table 5 is an opcode map for the instructions used on the MCU.

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single byte instructions; the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. One- or two-byte direct addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

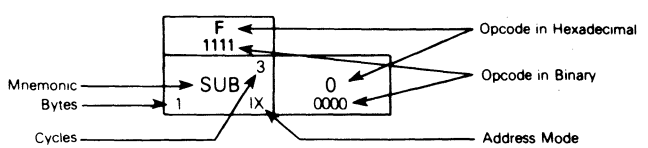
Table 5. Opcode Map

Hi / Low	Bit Manipulation BTB 0 0000	Bit Manipulation BSC 1 0001	Branch REL 2 0010	Read/Modify/Write DIR 3 0011	Read/Modify/Write INH 4 0100	Read/Modify/Write INH 5 0101	Read/Modify/Write IX1 6 0110	Read/Modify/Write IX 7 0111	Control INH 8 1000	Control INH 9 1001	Register/Memory IMM A 1010	Register/Memory DIR B 1011	Register/Memory EXT C 1100	Register/Memory IX2 D 1101	Register/Memory IX1 E 1110	Register/Memory IX F 1111	Hi / Low
0 0000	BRSET0 5 3 BTB	BSET0 5 2 BSC	BRA 3 2 REL	NEG 5 2 DIR	NEGA 3 1 INH	NEGX 3 1 INH	NEG 6 2 IX1	NEG 5 1 IX	RTI 9 1 INH		SUB 2 2 IMM	SUB 3 2 DIR	SUB 4 3 EXT	SUB 5 3 IX2	SUB 4 2 IX1	SUB 3 1 IX	0 0000
1 0001	BRCLR0 5 3 BTB	BCLR0 5 2 BSC	BRN 3 2 REL						RTS 6 1 INH		CMP 2 2 IMM	CMP 3 2 DIR	CMP 4 3 EXT	CMP 5 3 IX2	CMP 4 2 IX1	CMP 3 1 IX	1 0001
2 0010	BRSET1 5 3 BTB	BSET1 5 2 BSC	BHI 3 2 REL		MUL 11 1 INH						SBC 2 2 IMM	SBC 3 2 DIR	SBC 4 3 EXT	SBC 5 3 IX2	SBC 4 2 IX1	SBC 3 1 IX	2 0010
3 0011	BRCLR1 5 3 BTB	BCLR1 5 2 BSC	BLS 3 2 REL	COM 5 2 DIR	COMA 3 1 INH	COMX 3 1 INH	COM 6 2 IX1	COM 5 1 IX	SWI 10 1 INH		CPX 2 2 IMM	CPX 3 2 DIR	CPX 4 3 EXT	CPX 5 3 IX2	CPX 4 2 IX1	CPX 3 1 IX	3 0011
4 0100	BRSET2 5 3 BTB	BSET2 5 2 BSC	BCC 3 2 REL	LSR 5 2 DTR	LSRA 3 1 INH	LSRX 3 1 INH	LSR 6 2 IX1	LSR 5 1 IX			AND 2 2 IMM	AND 3 2 DIR	AND 4 3 EXT	AND 5 3 IX2	AND 4 2 IX1	AND 3 1 IX	4 0100
5 0101	BRCLR2 5 3 BTB	BCLR2 5 2 BSC	BCS 3 2 REL								BIT 2 2 IMM	BIT 3 2 DIR	BIT 4 3 EXT	BIT 5 3 IX2	BIT 4 2 IX1	BIT 3 1 IX	5 0101
6 0110	BRSET3 5 3 BTB	BSET3 5 2 BSC	BNE 3 2 REL	ROR 5 2 DIR	RORA 3 1 INH	RORX 3 1 INH	ROR 6 2 IX1	ROR 5 1 IX			LDA 2 2 IMM	LDA 3 2 DIR	LDA 4 3 EXT	LDA 5 3 IX2	LDA 4 2 IX1	LDA 3 1 IX	6 0110
7 0111	BRCLR3 5 3 BTB	BCLR3 5 2 BSC	BEQ 3 2 REL	ASR 5 2 DIR	ASRA 3 1 INH	ASRX 3 1 INH	ASR 6 2 IX1	ASR 5 1 IX		TAX 2 1 INH		STA 4 2 DIR	STA 5 3 EXT	STA 6 3 IX2	STA 5 2 IX1	STA 4 1 IX	7 0111
8 1000	BRSET4 5 3 BTB	BSET4 5 2 BSC	BHCC 3 2 REL	LSL 5 2 DIR	LSLA 3 1 INH	LSLX 3 1 INH	LSL 6 2 IX1	LSL 5 1 IX		CLC 2 1 INH	EOR 2 2 IMM	EOR 3 2 DIR	EOR 4 3 EXT	EOR 5 3 IX2	EOR 4 2 IX1	EOR 3 1 IX	8 1000
9 1001	BRCLR4 5 3 BTB	BCLR4 5 2 BSC	BHCS 3 2 REL	ROL 5 2 DIR	ROLA 3 1 INH	ROLX 3 1 INH	ROL 6 2 IX1	ROL 5 1 IX		SEC 2 1 INH	ADC 2 2 IMM	ADC 3 2 DIR	ADC 4 3 EXT	ADC 5 3 IX2	ADC 4 2 IX1	ADC 3 1 IX	9 1001
A 1010	BRSET5 5 3 BTB	BSET5 5 2 BSC	BPL 3 2 REL	DEC 5 2 DIR	DECA 3 1 INH	DECX 3 1 INH	DEC 6 2 IX1	DEC 5 1 IX		CLI 2 1 INH	ORA 2 2 IMM	ORA 3 2 DIR	ORA 4 3 EXT	ORA 5 3 IX2	ORA 4 2 IX1	ORA 3 1 IX	A 1010
B 1011	BRCLR5 5 3 BTB	BCLR5 5 2 BSC	BMI 3 2 REL							SEI 2 1 INH	ADD 2 2 IMM	ADD 3 2 DIR	ADD 4 3 EXT	ADD 5 3 IX2	ADD 4 2 IX1	ADD 3 1 IX	B 1011
C 1100	BRSET6 5 3 BTB	BSET6 5 2 BSC	BMC 3 2 REL	INC 5 2 DIR	INCA 3 1 INH	INCX 3 1 INH	INC 6 2 IX1	INC 5 1 IX		RSP 2 1 INH		JMP 2 2 DIR	JMP 3 3 EXT	JMP 4 3 IX2	JMP 3 2 IX1	JMP 2 1 IX	C 1100
D 1101	BRCLR6 5 3 BTB	BCLR6 5 2 BSC	BMS 3 2 REL	TST 4 2 DIR	TSTA 3 1 INH	TSTX 3 1 INH	TST 5 2 IX1	TST 4 1 IX		NOP 2 1 INH	BSR 6 2 REL	JSR 5 2 DIR	JSR 6 3 EXT	JSR 7 3 IX2	JSR 6 2 IX1	JSR 5 1 IX	D 1101
E 1110	BRSET7 5 3 BTB	BSET7 5 2 BSC	BIL 3 2 REL						STOP 2 1 INH		LDX 2 2 IMM	LDX 3 2 DIR	LDX 4 3 EXT	LDX 5 3 IX2	LDX 4 2 IX1	LDX 3 1 IX	E 1110
F 1111	BRCLR7 5 3 BTB	BCLR7 5 2 BSC	BIH 3 2 REL	CLR 5 2 DIR	CLRA 3 1 INH	CLRX 3 1 INH	CLR 6 2 IX1	CLR 5 1 IX	WAIT 2 1 INH	TXA 2 1 INH		STX 4 2 DIR	STX 5 3 EXT	STX 6 3 IX2	STX 5 2 IX1	STX 4 1 IX	F 1111

Abbreviations for Address Modes

INH	Inherent
A	Accumulator
X	Index Register
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction. When using the Motorola assembler, the user need not specify whether an instruction uses direct or extended addressing. The assembler automatically selects the shortest form of the instruction.

RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −126 to +129 from the opcode address. The programmer need not calculate the offset when using the Motorola assembler, since it calculates the proper offset and checks to see that it is within the span of the branch.

INDEXED, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. This addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this two-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This address mode can be used in a manner similar to indexed, 8-bit offset except that this three-byte instruction allows tables to be anywhere in memory. As with direct and extended addressing, the Motorola assembler determines the shortest form of indexed addressing.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode, and the byte following the opcode specifies the direct addressing of the byte in which the specified bit is to be set or cleared. Any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single two-byte instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit that is to be tested and its condition (set or clear), is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS (Voltages referenced to V_{SS})

Rating	Symbol	Value	Unit
Supply Voltage	V_{DD}	−0.3 to +7.0	V
Input Voltage	V_{in}	V_{SS} −0.3 to V_{DD} +0.3	V
Self-Check Mode ($\overline{IRQ}$ Pin Only)	V_{in}	V_{SS} −0.3 to 2×V_{DD} +0.3	V
Current Drain Per Pin Excluding V_{DD} and V_{SS}	I	25	mA
Operating Temperature Range MC68HC05C3P, FN MC68HC05C3CP, CFN MC68HC05C3VP, VFN MC68HC05C3MP, MFN	T_A	T_L to T_H 0 to +70 −40 to +85 −40 to +105 −40 to +125	°C
Storage Temperature Range	T_{stg}	−65 to +150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation, it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \le (V_{in}$ or $V_{out}) \le V_{DD}$. Reliability of operation is enhanced if unused inputs are connected to an appropriate logic voltage level (e.g., either V_{SS} or V_{DD}).

3

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic Plastic Leaded Chip Carrier (PLCC)	θ_{JA}	 60 70	°C W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{I/O}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
$P_{I/O}$ = Power Dissipation on Input and Output Pins — User Determined

For most applications $P_{I/O} \ll P_{INT}$ and can be neglected.

The following is an approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected):

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

V_{DD} = 4.5 V

Pins	R1	R2	C
PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4	3.26 kΩ	2.38 kΩ	50 pF
PD0, PD5, PD7	1.9 kΩ	2.26 kΩ	200 pF

V_{DD} = 3.0 V

Pins	R1	R2	C
PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4	10.91 kΩ 6.32 kΩ	50 pF	
PD0, PD5, PD7	6 kΩ	6 kΩ	200 pF

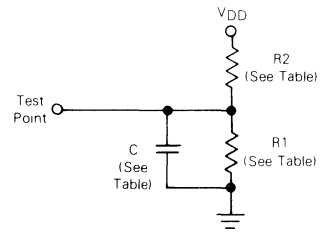


Figure 18. Equivalent Test Load

DC ELECTRICAL CHARACTERISTICS

(V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = T_L to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage, I_{Load} = 10.0 µA	V_{OL} V_{OH}	— V_{DD} − 0.1	— —	0.1 —	V
Output High Voltage (I_{Load} = 0.8 mA) PA0-PA7, PB0-PB7, PC0-PC7, TCMP (see Figure 19) (I_{Load} = 1.6 mA) PD1-PD4 (see Figure 20)	V_{OH}	 V_{DD} − 0.8 V_{DD} − 0.8	 — —	 — —	V
Output Low Voltage (see Figure 21) (I_{Load} = 1.6 mA) PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4, TCMP	V_{OL}	—	—	0.4	V
Input High Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IH}	0.7 × V_{DD}	—	V_{DD}	V
Input Low Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IL}	V_{SS}	—	0.2 × V_{DD}	V
Data Retention Mode (0° to 70°C)	V_{RM}	2.0	—	—	V
Supply Current (see Notes) Run (see Figures 22 and 23) Wait (see Figures 22 and 23) Stop (see Figure 23) 25°C 0° to 70°C (Standard) −40° to +85°C −40° to +125°C	I_{DD}	 — — — — — —	 3.5 1.6 2.0 — — —	 7.0 4.0 50 140 180 250	 mA mA µA µA µA µA
I/O Ports Hi-Z Leakage Current PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4	I_{IL}	—	—	±10	µA
Input Current $\overline{RESET}$, IRQ, TCAP, OSC1, PD0, PD5, PD7	I_{in}	—	—	±1	µA
Capacitance Ports (as Input or Output) $\overline{RESET}$, IRQ, TCAP, PD0-PD5, PD7	 C_{out} C_{in}	 — —	 — —	 12 8	pF

NOTES:
1. All values shown reflect average measurements.
2. Typical values at midpoint of voltage range, 25°C only.
3. Wait I_{DD}: Only timer system active (SPE = TE = RE = 0). If SPI, SCI active (SPE = TE = RE = 1) add 10% current draw.
4. Run (Operating) I_{DD}, Wait I_{DD}: Measured using external square wave clock source (f_{osc} = 4.2 MHz), all inputs 0.2 V from rail; no dc loads, less than 50 pF on all outputs, C_L = 20 pF on OSC2.
5. Wait, Stop I_{DD}: All ports configured as inputs, V_{IL} = 0.2 V, V_{IH} = V_{DD} − 0.2 V.
6. Stop I_{DD} measured with OSC1 = V_{SS}.
7. Standard temperature range is 0° to 70°C. Extended temperature versions and a 25°C only version are available.
8. Wait I_{DD} is affected linearly by the OSC2 capacitance.

DC ELECTRICAL CHARACTERISTICS

(V_{DD} = 3.3 Vdc ± 0.3 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage, $I_{Load} \leq 10.0\ \mu A$	V_{OL} V_{OH}	— $V_{DD} - 0.1$	— —	0.1 —	V
Output High Voltage (I_{Load} = 0.2 mA) PA0-PA7, PB0-PB7, PC0-PC7, TCMP (see Figure 19) (I_{Load} = 0.4 mA) PD1-PD4 (see Figure 20)	V_{OH}	 $V_{DD} - 0.3$ $V_{DD} - 0.3$	 — —	 — —	V
Output Low Voltage (see Figure 21) (I_{Load} = 0.4 mA) PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4, TCMP	V_{OL}	—	—	0.3	V
Input High Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IH}	$0.7 \times V_{DD}$	—	V_{DD}	V
Input Low Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IL}	V_{SS}	—	$0.2 \times V_{DD}$	V
Data Retention Mode (0° to 70°C)	V_{RM}	2.0	—	—	V
Supply Current (see Notes) Run (see Figures 22 and 24) Wait (see Figures 22 and 24) Stop (see Figure 24) 25°C 0° to 70°C (Standard) −40° to +85°C −40° to +125°C	I_{DD}	 — — — — — —	 1.0 0.5 1.0 — — —	 2.5 1.4 30 80 120 175	 mA mA μA μA μA μA
I/O Ports Hi-Z Leakage Current PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4	I_{IL}	—	—	±10	μA
Input Current $\overline{RESET}$, $\overline{IRQ}$, TCAP, OSC1, PD0, PD5, PD7	I_{in}	—	—	±1	μA
Capacitance Ports (as Input or Output) $\overline{RESET}$, $\overline{IRQ}$, TCAP, PD0-PD5, PD7	 C_{out} C_{in}	 — —	 — —	 12 8	pF

NOTES:
1. All values shown reflect average measurements.
2. Typical values at midpoint of voltage range, 25°C only.
3. Wait I_{DD}: Only timer system active (SPE = TE = RE = 0). If SPI, SCI active (SPE = TE = RE = 1) add 10% current draw.
4. Run (Operating) I_{DD}, Wait I_{DD}: Measured using external square wave clock source (f_{osc} = 4.2 MHz), all inputs 0.2 V from rail; no dc loads, less than 50 pF on all outputs, C_L = 20 pF on OSC2.
5. Wait, Stop I_{DD}: All ports configured as inputs, V_{IL} = 0.2 V, $V_{IH} = V_{DD} - 0.2$ V.
6. Stop I_{DD} measured with OSC1 = V_{SS}.
7. Standard temperature range is 0° to 70°C. Extended temperature versions and a 25°C only version are available.
8. Wait I_{DD} is affected linearly by the OSC2 capacitance.

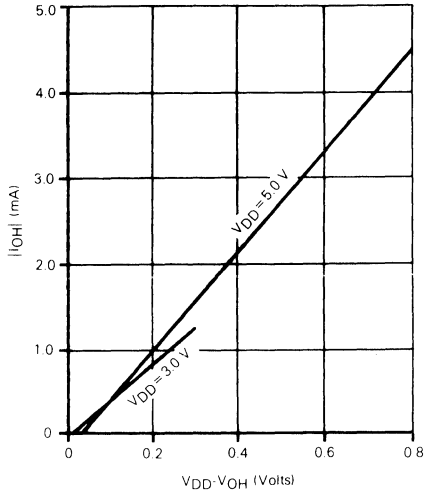


Figure 19. Typical V_{OH} vs I_{OH} for Ports A, B, C, and TCMP

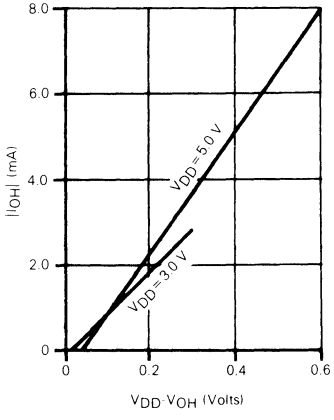


Figure 20. Typical V_{OH} vs I_{OH} for PD1-PD4

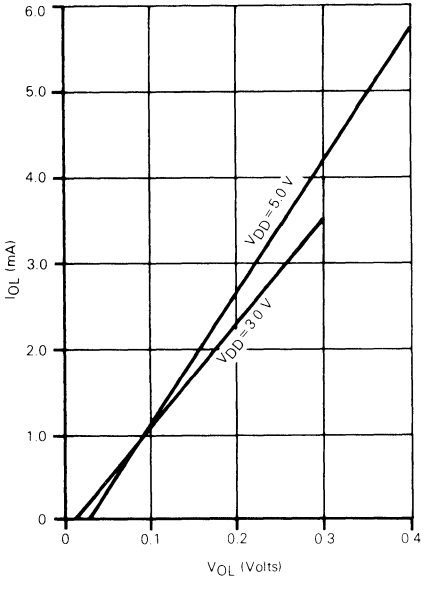


Figure 21. Typical V_{OL} vs I_{OL} for All Ports

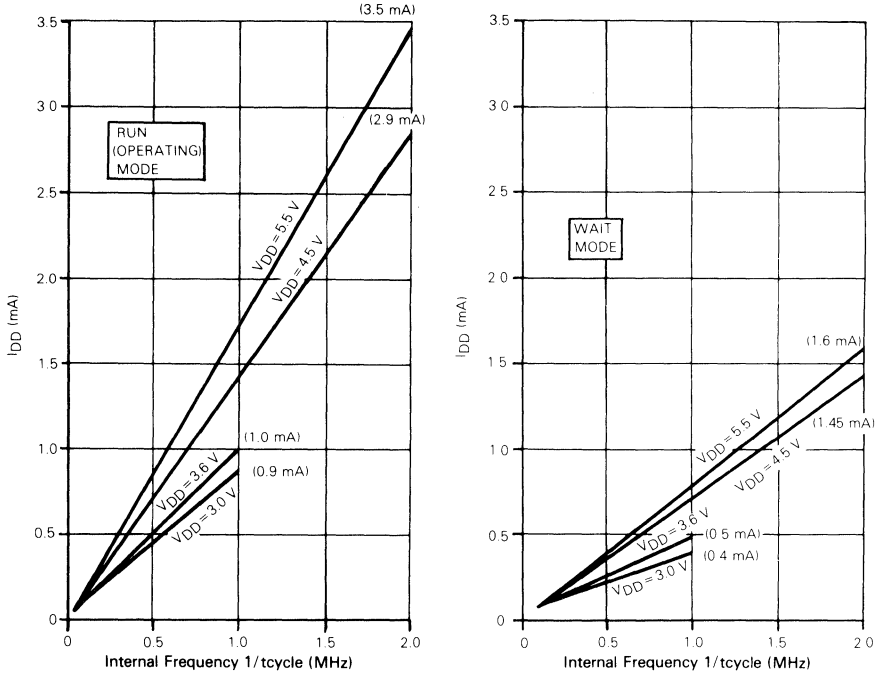


Figure 22. Typical Current vs Internal Frequency for Run and Wait Modes

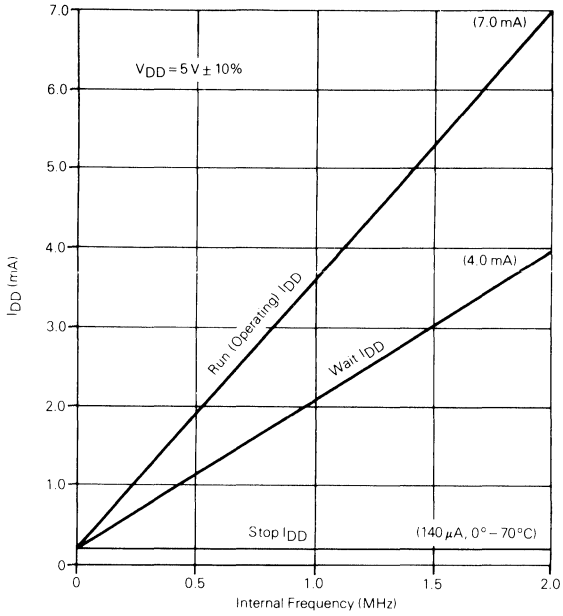


Figure 23. Maximum I_{DD} vs Frequency for $V_{DD} = 5.0$ Vdc

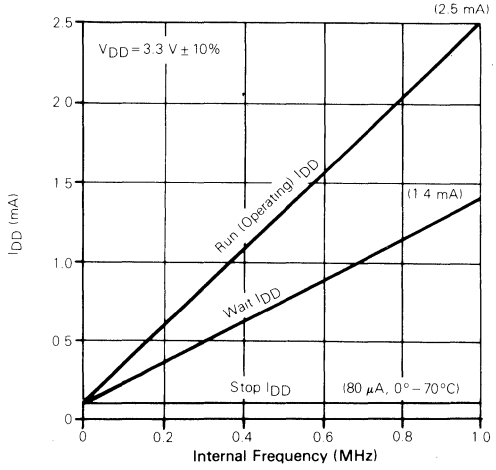


Figure 24. Maximum I_{DD} vs Frequency for $V_{DD} = 3.3$ Vdc

CONTROL TIMING

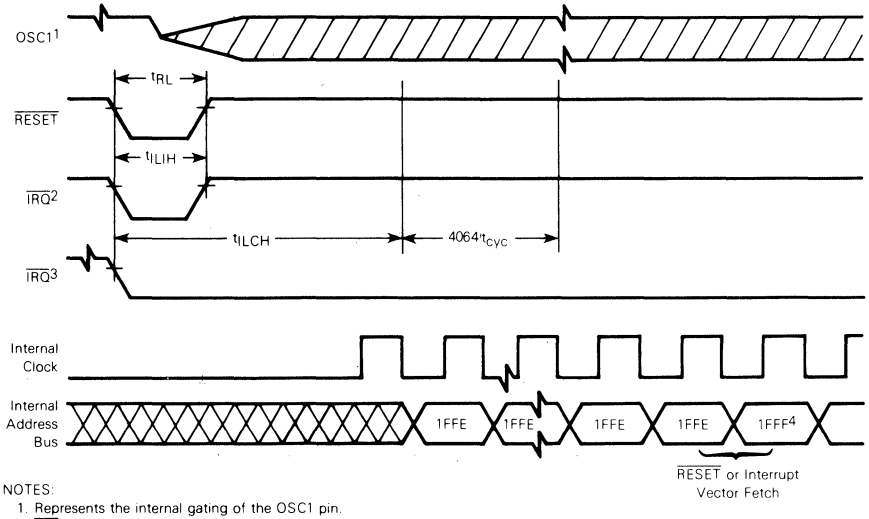
(V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation Crystal Option External Clock Option	f_{osc}	 — dc	 4.2 4.2	MHz
Internal Operating Frequency Crystal ($f_{osc} \div 2$) External Clock ($f_{osc} \div 2$)	f_{op}	 — dc	 2.1 2.1	MHz
Cycle Time (see Figure 28)	t_{cyc}	480	—	ns
Crystal Oscillator Startup Time (see Figure 28)	t_{OXOV}	—	100	ms
Stop Recovery Startup Time (Crystal Oscillator) (see Figure 25)	t_{ILCH}	—	100	ms
$\overline{RESET}$ Pulse Width (see Figure 28)	t_{RL}	1.5	—	t_{cyc}
Timer Resolution** Input Capture Pulse Width (see Figure 26) Input Capture Pulse Period (see Figure 26)	 t_{RESL} t_{TH}, t_{TL} t_{TLTL}	 4.0 125 ***	 — — —	 t_{cyc} ns t_{cyc}
Interrupt Pulse Width Low (Edge-Triggered) (see Figure 8)	t_{ILIH}	125	—	ns
Interrupt Pulse Period (see Figure 8)	t_{ILIL}	*	—	t_{cyc}
OSC1 Pulse Width	t_{OH}, t_{OL}	90	—	ns

*The minimum period t_{ILIL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus 21 t_{cyc}.

**Since a 2-bit prescaler in the timer must count four internal cycles (t_{cyc}), this is the limiting minimum factor in determining the timer resolution.

***The minimum period t_{TLTL} should not be less than the number of cycle times it takes to execute the capture interrupt service routine plus 24 t_{cyc}.



NOTES:
1. Represents the internal gating of the OSC1 pin.
2. $\overline{IRQ}$ pin edge-sensitive mask option.
3. $\overline{IRQ}$ pin level and edge-sensitive mask option.
4. $\overline{RESET}$ vector address shown for timing example.

Figure 25. Stop Recovery Timing Diagram

CONTROL TIMING

(V_{DD} = 3.3 Vdc ± 0.3 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation Crystal Option External Clock Option	f_{osc}	 — dc	 2.0 2.0	MHz
Internal Operating Frequency Crystal ($f_{osc} \div 2$) External Clock ($f_{osc} \div 2$)	f_{op}	 — dc	 1.0 1.0	MHz
Cycle Time (see Figure 28)	t_{cyc}	1000	—	ns
Crystal Oscillator Startup Time (see Figure 28)	t_{OXOV}	—	100	ms
Stop Recovery Startup Time (Crystal Oscillator) (see Figure 25)	t_{ILCH}	—	100	ms
$\overline{\text{RESET}}$ Pulse Width — Excluding Power-Up (see Figure 28)	t_{RL}	1.5	—	t_{cyc}
Timer Resolution** Input Capture Pulse Width (see Figure 26) Input Capture Pulse Period (see Figure 26)	 t_{RESL} t_{TH}, t_{TL} t_{TLTL}	 4.0 250 ***	 — — —	 t_{cyc} ns t_{cyc}
Interrupt Pulse Width Low (Edge-Triggered) (see Figure 8)	t_{ILIH}	250	—	ns
Interrupt Pulse Period (see Figure 8)	t_{ILIL}	*	—	t_{cyc}
OSC1 Pulse Width	t_{OH}, t_{OL}	200	—	ns

*The minimum period t_{ILIL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus 21 t_{cyc}.

**Since a 2-bit prescaler in the timer must count four internal cycles (t_{cyc}), this is the limiting minimum factor in determining the timer resolution.

***The minimum period t_{TLTL} should not be less than the number of cycle times it takes to execute the capture interrupt service routine plus 24 t_{cyc}.

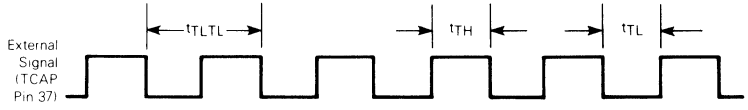


Figure 26. Timer Relationships

SERIAL PERIPHERAL INTERFACE (SPI) TIMING

(V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H) (see Figure 27)

Num.	Characteristic	Symbol	Min	Max	Unit
	Operating Frequency Master Slave	 $f_{op(m)}$ $f_{op(s)}$	 dc dc	 0.5 2.1	 f_{op} MHz
1	Cycle Time Master Slave	 $t_{cyc(m)}$ $t_{cyc(s)}$	 2.0 480	 — —	 t_{cyc} ns
2	Enable Lead Time Master Slave	 $t_{lead(m)}$ $t_{lead(s)}$	 * 240	 — —	 ns ns
3	Enable Lag Time Master Slave	 $t_{lag(m)}$ $t_{lag(s)}$	 * 240	 — —	 ns ns
4	Clock (SCK) High Time Master Slave	 $t_{w(SCKH)m}$ $t_{w(SCKH)s}$	 340 190	 — —	 ns ns
5	Clock (SCK) Low Time Master Slave	 $t_{w(SCKL)m}$ $t_{w(SCKL)s}$	 340 190	 — —	 ns ns
6	Data Setup Time (Inputs) Master Slave	 $t_{su(m)}$ $t_{su(s)}$	 100 100	 — —	 ns ns
7	Data Hold Time (Inputs) Master Slave	 $t_{h(m)}$ $t_{h(s)}$	 100 100	 — —	 ns ns
8	Access Time (Time to Data Active from High-Impedance State) Slave	 t_a	 0	 120	 ns
9	Disable Time (Hold Time to High-Impedance State) Slave	 t_{dis}	 —	 240	 ns
10	Data Valid Master (Before Capture Edge) Slave (After Enable Edge)**	 $t_{v(m)}$ $t_{v(s)}$	 0.25 —	 — 240	 $t_{cyc(m)}$ ns
11	Data Hold Time (Outputs) Master (After Capture Edge) Slave (After Enable Edge)	 $t_{ho(m)}$ $t_{ho(s)}$	 0.25 0	 — —	 $t_{cyc(m)}$ ns
12	Rise Time (20% V_{DD} to 70% V_{DD}, C_L = 200 pF) SPI Outputs (SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and SS)	 t_{rm} t_{rs}	 — —	 100 2.0	 ns µs
13	Fall Time (70% V_{DD} to 20% V_{DD}, C_L = 200 pF) SPI Outputs (SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and SS)	 t_{fm} t_{fs}	 — —	 100 2.0	 ns µs

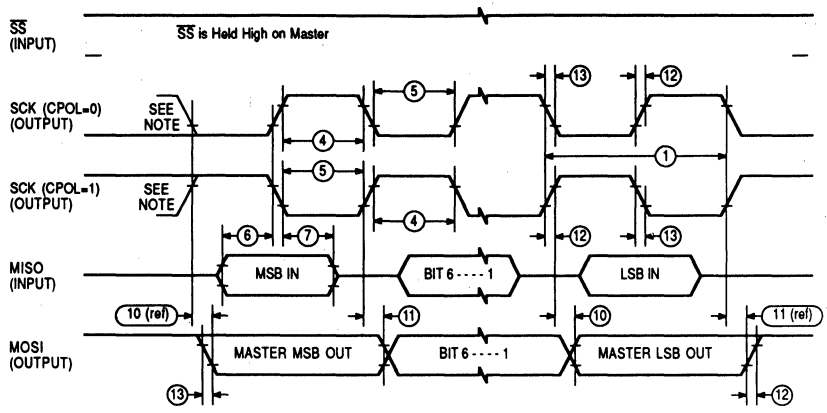
*Signal production depends on software.
**Assumes 200 pF load on all SPI pins.

SERIAL PERIPHERAL INTERFACE (SPI) TIMING

(V_{DD} = 3.3 Vdc ± 0.3 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H) (see Figure 27)

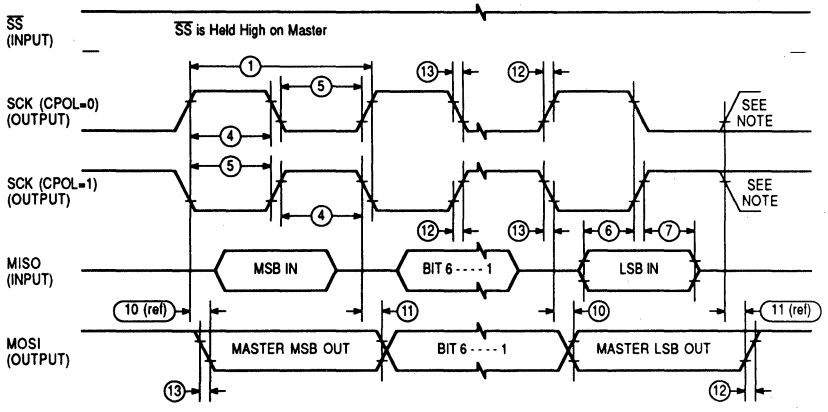
Num.	Characteristic	Symbol	Min	Max	Unit
	Operating Frequency Master Slave	 $f_{op(m)}$ $f_{op(s)}$	 dc dc	 0.5 1.0	 f_{op} MHz
1	Cycle Time Master Slave	 $t_{cyc(m)}$ $t_{cyc(s)}$	 2.0 1.0	 — —	 t_{cyc} µs
2	Enable Lead Time Master Slave	 $t_{lead(m)}$ $t_{lead(s)}$	 * 500	 — —	 ns ns
3	Enable Lag Time Master Slave	 $t_{lag(m)}$ $t_{lag(s)}$	 * 500	 — —	 ns ns
4	Clock (SCK) High Time Master Slave	 $t_{w(SCKH)m}$ $t_{w(SCKH)s}$	 720 400	 — —	 µs ns
5	Clock (SCK) Low Time Master Slave	 $t_{w(SCKL)m}$ $t_{w(SCKL)s}$	 720 400	 — —	 µs ns
6	Data Setup Time (Inputs) Master Slave	 $t_{su(m)}$ $t_{su(s)}$	 200 200	 — —	 ns ns
7	Data Hold Time (Inputs) Master Slave	 $t_{h(m)}$ $t_{h(s)}$	 200 200	 — —	 ns ns
8	Access Time (Time to Data Active from High-Impedance State) Slave	 t_a	 0	 250	 ns
9	Disable Time (Hold Time to High-Impedance State) Slave	 t_{dis}	 —	 500	 ns
10	Data Valid Master (Before Capture Edge) Slave (After Enable Edge)**	 $t_{v(m)}$ $t_{v(s)}$	 0.25 —	 — 500	 $t_{cyc(m)}$ ns
11	Data Hold Time (Outputs) Master (After Capture Edge) Slave (After Enable Edge)	 $t_{ho(m)}$ $t_{ho(s)}$	 0.25 0	 — —	 $t_{cyc(m)}$ ns
12	Rise Time (20% V_{DD} to 70% V_{DD}, C_L = 200 pF) SPI Outputs (SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and SS)	 t_{rm} t_{rs}	 — —	 200 2.0	 ns µs
13	Fall Time (70% V_{DD} to 20% V_{DD}, C_L = 200 pF) SPI Outputs (SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and SS)	 t_{fm} t_{fs}	 — —	 200 2.0	 ns µs

*Signal production depends on software.
**Assumes 200 pF load on all SPI pins.



NOTE: This first clock edge is generated internally but is not seen at the SCK pin.

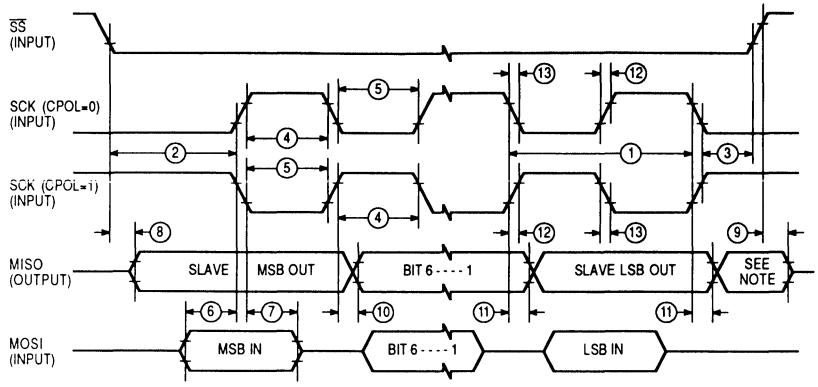
a) SPI MASTER TIMING (CPHA = 0)



NOTE: This last clock edge is generated internally but is not seen at the SCK pin.

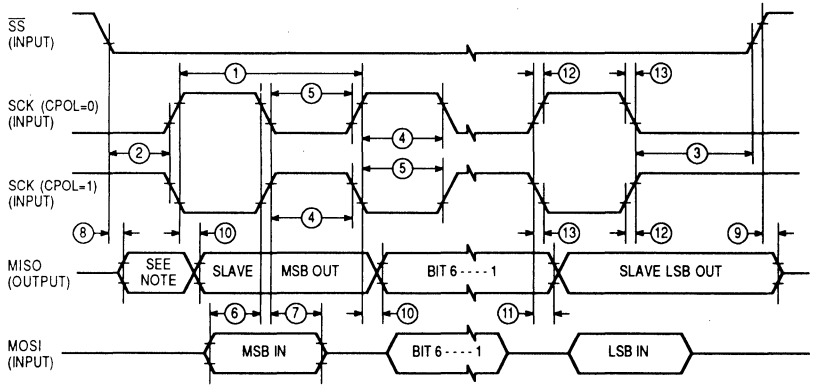
b) SPI MASTER TIMING (CPHA = 1)

Figure 27. SPI Timing Diagrams (Sheet 1 of 2)



NOTE: Not defined but normally MSB of character just received.

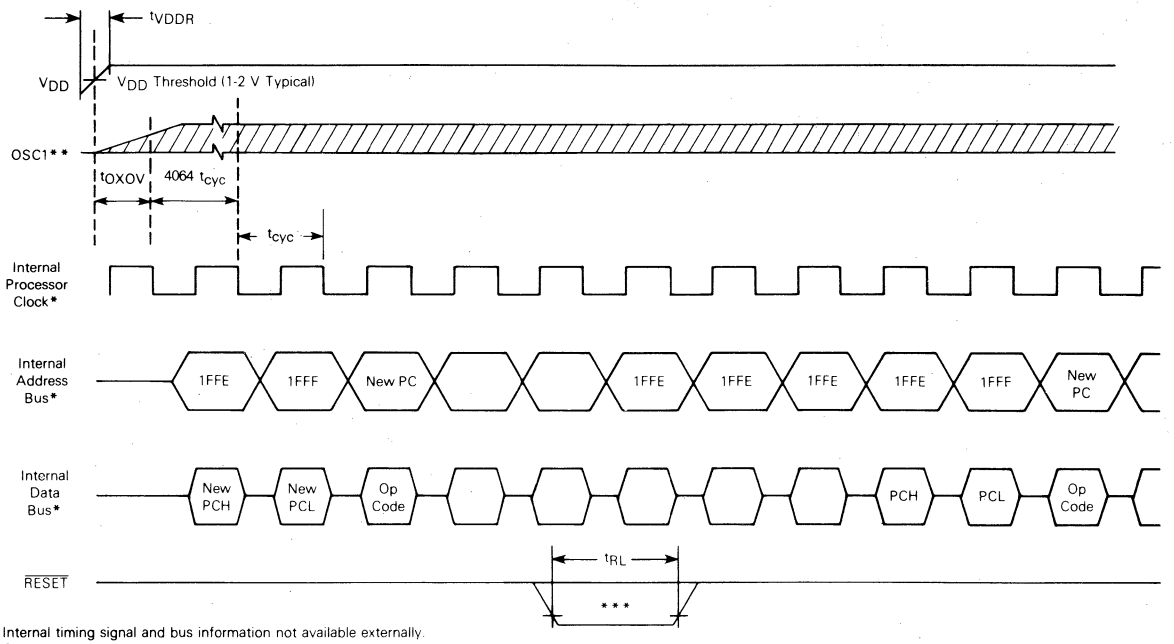
c) SPI SLAVE TIMING (CPHA = 0)



NOTE: Not defined but normally LSB of character previously transmitted.

d) SPI SLAVE TIMING (CPHA = 1)

Figure 27. SPI Timing Diagrams (Sheet 2 of 2)



*Internal timing signal and bus information not available externally.
**OSC1 line is not meant to represent frequency. It is only used to represent time.
***The next rising edge of the internal processor clock following the rising edge of $\overline{\text{RESET}}$ initiates the reset sequence.

Figure 28. Power-On Reset and $\overline{\text{RESET}}$

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola in the following media:

MDOS™, disk file
MS™-DOS/PC-DOS disk file (360K)
EPROM(s) 2764, MCM68764, MCM68766, or EEPROM MC68HC805C4

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, a sales person, or a Motorola representative.

FLEXIBLE DISKS

A flexible disk (MS-DOS/PC-DOS disk file), programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. The diskette should be clearly labeled with the customer's name, data, project or product name, and the name of the file containing the pattern.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to expedite the process in case of any difficulty with the pattern file.

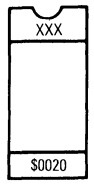
MS-DOS/PC-DOS Disk File

MS-DOS is Microsoft's Disk Operating System. PC-DOS is the IBM® Personal Computer (PC) Disk Operating System. Disk media submitted must be a standard density (360K) double-sided 5 1/4 inch compatible floppy diskette. The diskette must contain object file code in Motorola's S-record format. The S-record format is a character-based object file format generated by M6805 cross assemblers and linkers on IBM PC style machines.

EPROMs

A 2764, 68764, or 68766 type EPROM, programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. Since all program and data space information will fit on one 2764, 68764, or 68766 EPROM device, the EPROM must be programmed as described in the following paragraphs.

For an MC68HC805C4 MCU start the page zero, user ROM at EEPROM address $0020 through $004F. Start the user ROM at EEPROM address $0100 through $0BFF with vectors from $1FF4 to $1FFF. All unused bytes, including the user's space, must be set to zero. For shipment to Motorola, EPROMs should be placed in a conductive IC carrier and packed securely. Styrofoam is not acceptable for shipment.



xxx = Customer ID

Verification Media

All original pattern media (EPROMs or floppy disks) are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked, and the verification form should be completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for the creation of the customer mask. To aid in the verification process, Motorola will program *customer supplied* blank EPROM(s) or DOS disks from the data file used to create the custom mask.

ROM VERIFICATION UNITS (RVUs)

Ten MCUs containing the customer's ROM pattern will be sent for program verification. These units will have been made using the custom mask, but are for the purpose of ROM verification only. For expediency, the MCUs are unmarked, packaged in ceramic, and tested with five volts at room temperature. These RVUs are free with the minimum order quantity, but are not production parts. RVUs are not backed or guaranteed by Motorola Quality Assurance.

ORDERING INFORMATION

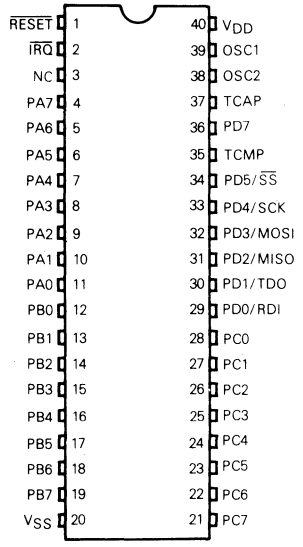
The following table provides ordering information pertaining to the package type, temperature, and MC order numbers for the MC68HC05C3 device.

Package Type	Temperature	MC Order Number
Plastic (P Suffix)	0°C to +70°C	MC68HC05C3P
	−40°C to +85°C	MC68HC05C3CP
	−40°C to +105°C	MC68HC05C3VP
	−40°C to +125°C	MC68HC05C3MP
PLCC (FN Suffix)	0°C to +70°C	MC68HC05C3FN
	−40°C to +85°C	MC68HC05C3CFN
	−40°C to +105°C	MC68HC05C3VFN
	−40°C to +125°C	MC68HC05C3MFN

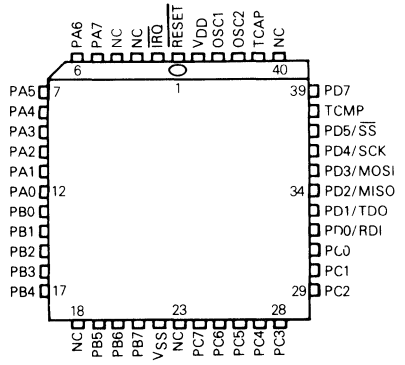
MDOS is a trademark of Motorola Inc.
MS is a trademark of Microsoft, Inc.
IBM is a registered trademark of International Business Machines Corporation.

PIN ASSIGNMENTS

40-PIN DUAL-IN-LINE PACKAGE



44-LEAD PLCC PACKAGE



NOTE : Bulk substrate tied to V_{SS}.

MC68HC05C4

Technical Summary
8-Bit Microcontroller Unit

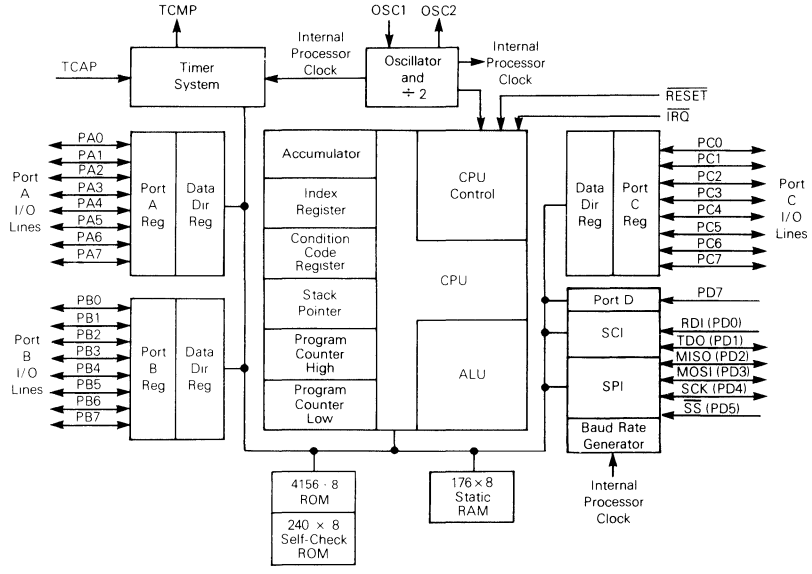
The MC68HC05C4 (HCMOS) microcontroller unit (MCU) is a member of the M68HC05 Family of microcontrollers. This high-performance, low-power MCU has parallel I/O capability with pins programmable as input or output. This publication contains condensed information on the MCU; for more detailed information, contact your local Motorola sales office.

The following block diagram depicts the hardware features; additional features available on the MCU are as follows:

- On-Chip Oscillator with RC or Crystal/Ceramic Resonator Mask Options
- Memory-Mapped I/O
- 176 Bytes of On-Chip RAM
- 4156 Bytes of User ROM
- 24 Bidirectional I/O Lines and 7 Input-Only Lines
- Serial Communications Interface (SCI) System
- Serial Peripheral Interface (SPI) System
- Self-Check Mode
- Power-Saving STOP, WAIT, and Data Retention Modes
- Single 3.0- to 5.5-Volt Supply (2-Volt Data Retention Mode)
- Fully Static Operation
- 8×8 Unsigned Multiply Instruction

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

The signal descriptions of the MCU are discussed in the following paragraphs.

V_{DD} AND V_{SS}

Power is supplied to the microcontroller using these two pins. V_{DD} is the positive supply, and V_{SS} is ground.

$\overline{IRQ}$

This pin is a programmable option that provides two different choices of interrupt triggering sensitivity. Refer to **INTERRUPTS** for more detail.

OSC1, OSC2

These pins provide control input for an on-chip clock oscillator circuit. A crystal, a ceramic resonator, a resistor/capacitor combination, or an external signal connects to these pins providing a system clock. A mask option selects either a crystal/ceramic resonator or a resistor/capacitor as the frequency determining element. The oscillator frequency is two times the internal bus rate.

RC Oscillator

With this option, a resistor is connected to the oscillator pins as shown in Figure 1(d). The relationship between R and f_{osc} is shown in Figure 2.

Crystal

The circuit shown in Figure 1(b) is recommended when using a crystal. Using an external CMOS oscillator is recommended when crystals outside the specified ranges are to be used. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and start-up stabilization time. Refer to **ELECTRICAL SPECIFICATIONS** for V_{DD} specifications.

3

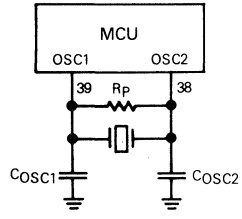
Crystal

	2 MHz	4 MHz	Units
R_{SMAX}	400	75	Ω
C_0	5	7	pF
C_1	0.008	0.012	μF
C_{OSC1}	15-40	15-30	pF
C_{OSC2}	15-30	15-25	pF
R_P	10	10	MΩ
Q	30	40	K

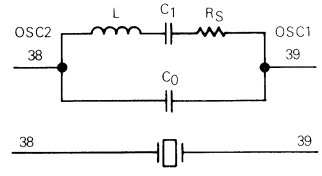
Ceramic Resonator

	2-4 MHz	Units
R_S (typical)	10	Ω
C_0	40	pF
C_1	4.3	pF
C_{OSC1}	30	pF
C_{OSC2}	30	pF
R_P	1-10	MΩ
Q	1250	–

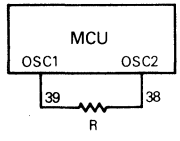
(a) Crystal/Ceramic Resonator Parameters



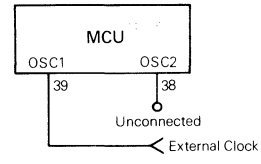
(b) Crystal/Ceramic Resonator Oscillator Connections



(c) Equivalent Crystal Circuit



(d) RC Oscillator Connections



(e) External Clock Source Connections (For Crystal Mask Option Only)

Figure 1. Oscillator Connections

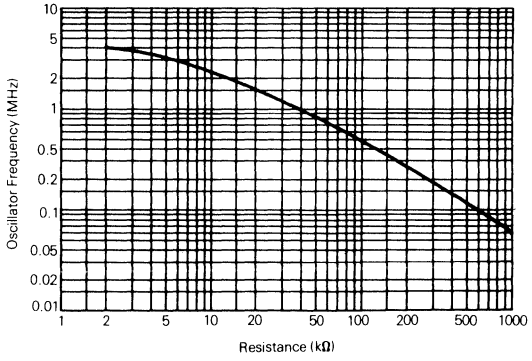


Figure 2. Typical Frequency vs Resistance for RC Oscillator Option Only

Ceramic Resonator

A ceramic resonator may be used in place of the crystal in cost-sensitive applications. The circuit in Figure 1(b) is recommended when using a ceramic resonator. Figure 1(a) lists the recommended capacitance and resistance values. The manufacturer of the resonator considered should be consulted for specific information on resonator operation.

External Clock

An external clock should be applied to the OSC1 input with the OSC2 input not connected, as shown in Figure 1(e). This option may only be used with the crystal oscillator mask option.

INPUT CAPTURE (TCAP)

This pin controls the input capture feature for the on-chip programmable timer.

OUTPUT COMPARE (TCMP)

This pin provides an output for the output compare feature of the on-chip timer.

$\overline{\text{RESET}}$

This pin is used to reset the MCU and provide an orderly start-up procedure by pulling $\overline{\text{RESET}}$ low.

INPUT/OUTPUT PORTS (PA0-PA7, PB0-PB7, PC0-PC7)

These 24 lines are arranged into three 8-bit ports (A, B, and C). These ports are programmable as either inputs or outputs under software control of the data direction registers. Refer to **PROGRAMMING** for additional information.

FIXED INPUT PORT (PD0-PD5, PD7)

These seven lines comprise port D, a fixed input port. All special functions that are enabled (SPI, SCI) affect this port. Refer to **PROGRAMMING** for additional information.

PROGRAMMING

Input/output port programming, fixed input port programming, and serial port programming are discussed in the following paragraphs.

INPUT/OUTPUT PORT PROGRAMMING

Any port pin is programmable as either an input or an output under software control of the corresponding data direction register (DDR). Each port bit can be selected as output or input by writing the corresponding bit in the port DDR to a logic one for output and logic zero for input. On reset, all DDRs are initialized to logic zero to put the ports in the input mode. The port output registers are not initialized on reset but may be written to before setting the DDR bits to avoid undefined levels.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This port write may be used to initialize the data registers and avoid undefined outputs. Refer to Figure 3 for typical port circuitry and to Table 1 for a list of the I/O pin functions.

Table 1. I/O Pin Functions

R/$\overline{\text{W}}$*	DDR	I/O Pin Functions
0	0	The I/O pin is in input mode. Data is written into the output data latch.
0	1	Data is written into the output data latch and output to the I/O pin.
1	0	The state of the I/O pin is read.
1	1	The I/O pin is in an output mode. The output data latch is read.

*R/$\overline{\text{W}}$ is an internal signal.

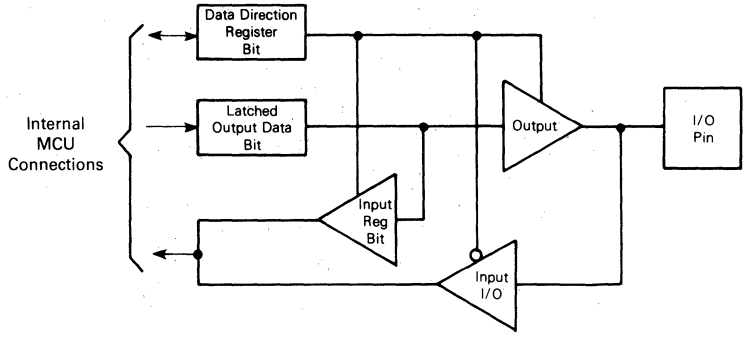


Figure 3. Typical Port I/O Circuit

3

FIXED INPUT PORT PROGRAMMING

Port D is a fixed input port (PD0-PD5, PD7) that monitors the external pins whenever the SCI or SPI is disabled. After reset, all seven bits become valid inputs because all special function drivers are disabled. For example, with the SCI enabled, PD0 and PD1 inputs will read zero. With the SPI disabled, PD2 through PD5 will read the state of the pin at the time of the read operation.

NOTE

Any unused inputs and I/O ports should be tied to an appropriate logic level (e.g., either V_{DD} or V_{SS}).

SERIAL PORT (SCI AND SPI) PROGRAMMING

The SCI and SPI use the port D pins for their functions. The SCI requires two pins (PD0-PD1) for its receive data input (RDI) and transmit data output (TD0), respectively. The SPI function requires four of the pins (PD2-PD5) for its serial data input/output (MISO), serial data output/input (MOSI), serial clock (SCK), and slave select ($\overline{SS}$), respectively.

MEMORY

The MCU is capable of addressing 8192 bytes of memory and I/O registers, as shown in Figure 4. The locations consist of user ROM, user RAM, self-check ROM, control registers, and I/O. The user-defined reset and interrupt vectors are located from $1FF4 to $1FFF.

The shared stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer decrements during pushes and increments during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

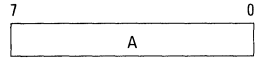
Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

REGISTERS

The MCU contains the registers described in the following paragraphs.

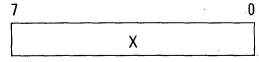
ACCUMULATOR (A)

The accumulator is a general-purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



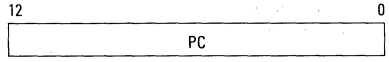
INDEX REGISTER (X)

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16-bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



PROGRAM COUNTER (PC)

The program counter is a 13-bit register that contains the address of the next byte to be fetched.



STACK POINTER (SP)

The stack pointer is a 13-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set to location $00FF. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

When accessing memory, the seven most significant bits are permanently set to 0000011. These seven bits are appended to the six least significant register bits to produce an address within the range of $00FF to $00C0.

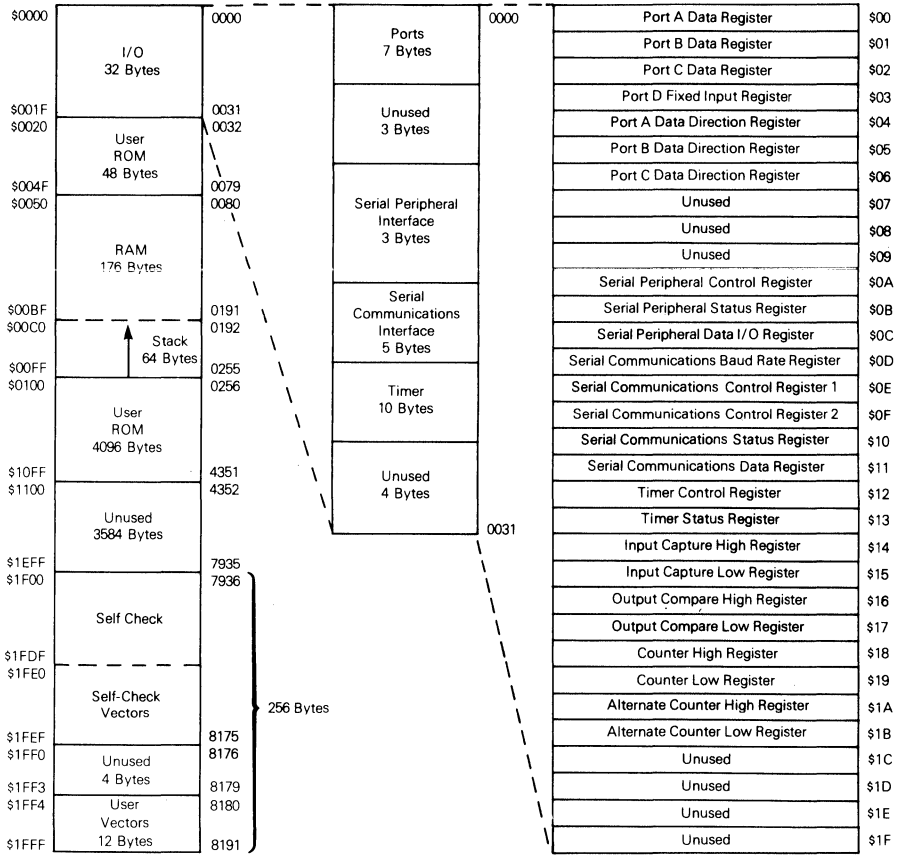
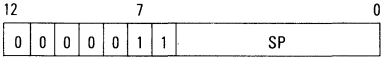


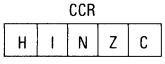
Figure 4. Memory Map

Subroutines and interrupts may use up to 64 (decimal) locations. If 64 locations are exceeded, the stack pointer wraps around and loses the previously stored information. A subroutine call occupies two locations on the stack; an interrupt uses five locations.



CONDITION CODE REGISTER (CCR)

The CCR is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program, and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.



Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, the timer and external interrupt is masked (disabled). If an interrupt occurs while this bit is set, the interrupt is latched and processed as soon as the interrupt bit is cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic one).

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/Borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions and during shifts and rotates.

SELF-CHECK

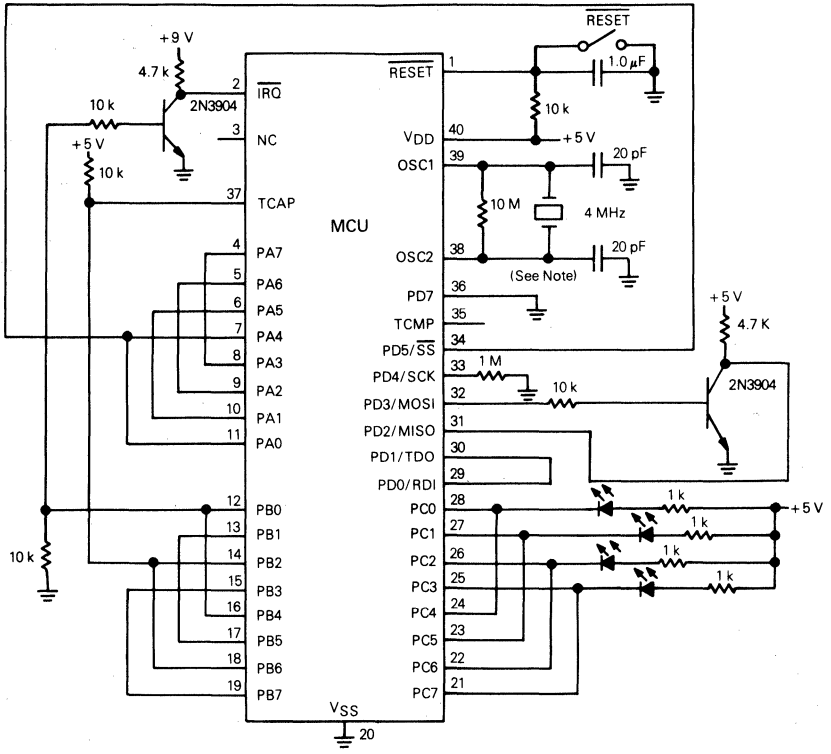
The self-check capability provides the ability to determine if the device is functional. Self-check is performed using the circuit shown in Figure 5. Port C pins PC0-PC3 are monitored for the self-check results. After reset, the following seven tests are performed automatically:

I/O — Exercise of ports A, B, and C
RAM — Counter test for each RAM byte
ROM — Exclusive OR with odd ones parity result
Timer — Tracks counter register and checks OCF flag
Interrupts — Tests external, timer, SCI and SPI interrupts
SCI — Transmission test; checks RDRF, TDRE, TC, and FE flags
SPI — Transmission test; checks SPIF, WCOL, and MODF flags

Self-check results (using the LEDs as monitors) are shown in Table 2. The following subroutines are available to the user and do not require any external hardware.

TIMER TEST SUBROUTINE

This subroutine returns with the Z bit cleared if any error is detected; otherwise, the Z bit is set. The timer test subroutine is called at location $1FOE. The output compare register is first set to the current timer state. Because the timer is free running and has only a divide-by-four prescaler, each timer count cannot be tested. The test reads the timer once every 10 counts (40 cycles) and checks for correct counting. The test tracks the counter until the timer wraps around, triggering the output compare flag in the timer status register. RAM locations $0050 and $0051 are overwritten. Upon return to the user's program, X=40. If the test passed, A=0.



NOTE: The RC Oscillator Option may also be used in this circuit.

Figure 5. Self-Check Circuit Schematic Diagram

Table 2. Self-Check Results

PC3	PC2	PC1	PC0	Remarks
1	0	0	1	Bad I/O
1	0	1	0	Bad RAM
1	0	1	1	Bad Timer
1	1	0	0	Bad SCI
1	1	0	1	Bad ROM
1	1	1	0	Bad SPI
1	1	1	1	Bad Interrupts or $\overline{IRQ}$ Request
Flashing				Good Device
All Others				Bad Device, Bad Port C, etc.

0 indicates LED is on; 1 indicates LED is off.

ROM CHECKSUM SUBROUTINE

This subroutine returns with the Z bit cleared if any error is detected; otherwise, the Z bit is set. The ROM checksum subroutine is called at location $1F93 with RAM location $0053 equal to $01 and A = 0. A short routine is set up and executed in RAM to compute a checksum of the entire ROM pattern. RAM locations $0050 through $0053 are overwritten. Upon return to the user's program, X = 0. If the test passed, A = 0.

RESETS

The MCU can be reset two ways: by initial power-up and by the external reset input ($\overline{RESET}$). The $\overline{RESET}$ input consists mainly of a Schmitt trigger that senses the $\overline{RESET}$ line logic level.

POWER-ON RESET (POR)

An internal reset is generated on power-up to allow the internal clock generator to stabilize. The power-on reset is strictly for power turn-on conditions and should not be used to detect a drop in the power supply voltage. There is a 4064 internal processor clock cycle (t_{CYC}) delay after the oscillator becomes active. If the $\overline{RESET}$ pin is low at the end of 4046 t_{CYC}, the MCU will remain in the reset condition until $\overline{RESET}$ goes high.

EXTERNAL RESET INPUT

The MCU is reset when a logic zero is applied to the $\overline{RESET}$ input for a period of one and one-half machine cycles (t_{CYC}).

INTERRUPTS

The MCU can be interrupted five different ways: the four maskable hardware interrupts ($\overline{IRQ}$, SPI, SCI, and timer) and the nonmaskable software interrupt instruction (SWI).

Interrupts cause the processor to save register contents on the stack and to set the interrupt mask (I bit) to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack and normal processing to resume. The stacking order is shown in Figure 6.

Unlike $\overline{RESET}$, hardware interrupts do not cause the current instruction execution to be halted but are considered pending until the current instruction is complete.

NOTE

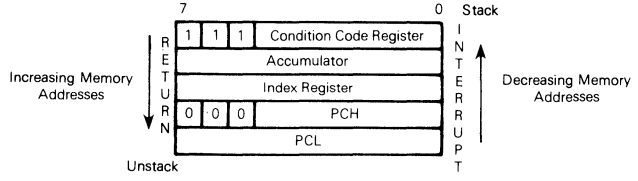
The current instruction is the one already fetched and being operated on.

When the current instruction is complete, the processor checks all pending hardware interrupts. If unmasked (I bit clear) and if the corresponding interrupt enable bit is set, the processor proceeds with interrupt processing; otherwise, the next instruction is fetched and executed.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction, regardless of the I-bit state. Refer to Figure 7 for the reset and interrupt instruction processing sequence.

TIMER INTERRUPT

There are three different timer interrupt flags that cause a timer interrupt whenever they are set and enabled. The interrupt flags are in the timer status register (TSR), and the enable bits are in the timer control register (TCR). Refer to **TIMER** for more information.



NOTE: Since the Stack Pointer decrements during pushes, the PCL is stacked first, followed by PCH, etc. Pulling from the stack is in the reverse order.

Figure 6. Interrupt Stacking Order

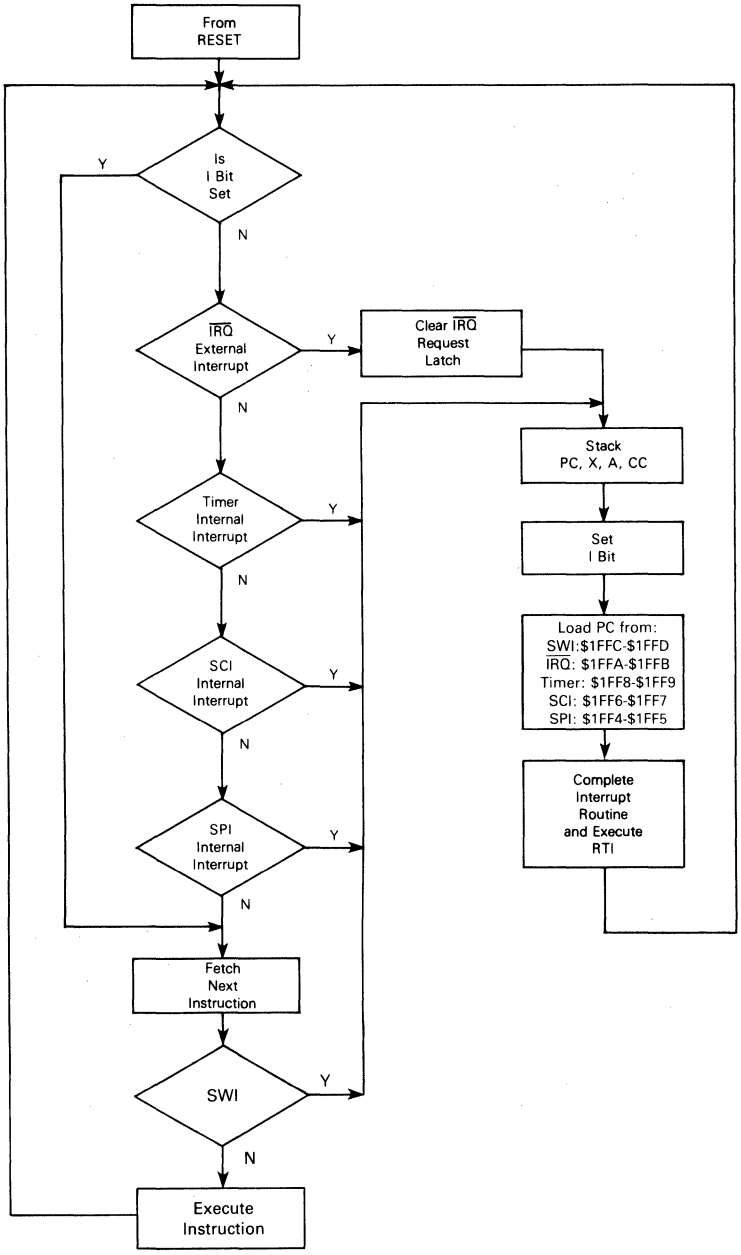


Figure 7. Reset and Interrupt Processing Flowchart

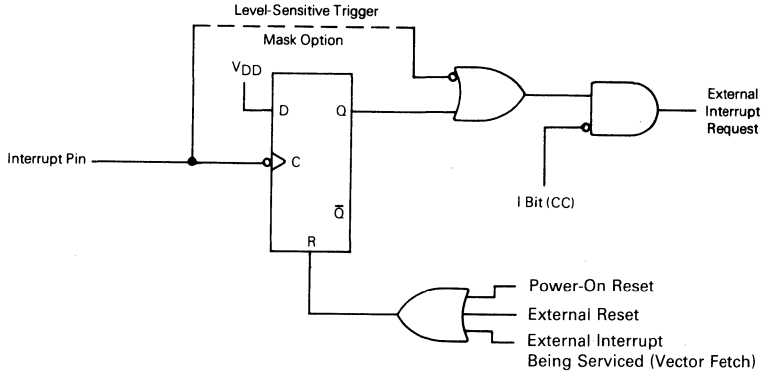
EXTERNAL INTERRUPT

If the interrupt mask bit (I bit) of the CCR is set, all interrupts are disabled. Clearing the I bit enables the external interrupt. The external interrupt is internally synchronized and then latched on the falling edge of $\overline{IRQ}$. The action of the external interrupt is identical to the timer interrupt with the exception that the interrupt request input at $\overline{IRQ}$ is latched internally and the service routine address is specified by the contents of $1FFA and $1FFB.

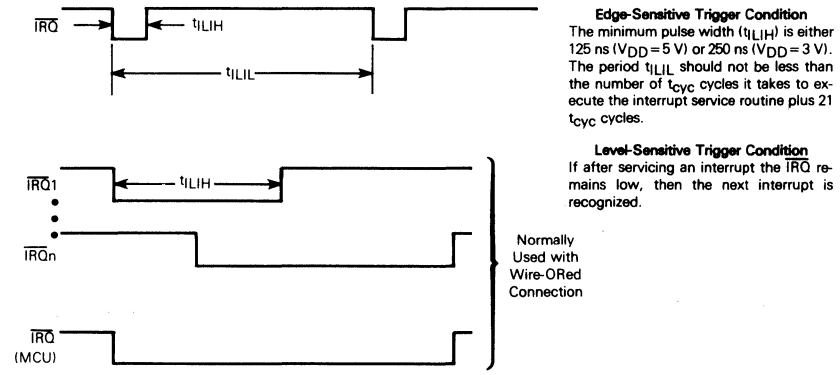
Either a level-sensitive and edge-sensitive trigger, or an edge-sensitive-only trigger are available as a mask option. Figure 8 shows both a functional internal diagram and a mode timing diagram for the interrupt line. The timing diagram shows two treatments of the interrupt line to the processor. The first method shows a single pulse on the interrupt line spaced far enough apart to be serviced. The minimum time between pulses is a function of the length of the interrupt service. Once a pulse occurs, the next pulse should not occur until an RTI occurs. This time (t_{ILIL}) is obtained by adding 21 instruction cycles to the total number of cycles it takes to complete the service routine (not including the RTI instruction). The second method shows many interrupt lines "wire-ORed" to form the interrupts at the processor. If the interrupt line remains low after servicing an interrupt, then the next interrupt is recognized.

NOTE

The internal interrupt latch is cleared in the first part of the interrupt service routine; therefore, one external interrupt pulse could be latched and serviced as soon as the I bit is cleared.



(a) Interrupt Internal Function Diagram



(b) Interrupt Mode Diagram

Figure 8. External Interrupt

SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. If the I bit is zero, SWI executes after the other interrupts. The SWI operation is similar to the hardware interrupts. The interrupt service routine address is specified by the contents of memory locations $1FFC and $1FFD.

SCI INTERRUPTS

An interrupt in the SCI occurs when one of the interrupt flag bits in the serial communications status register is set, provided the I bit in the CCR is clear and the enable bit in the serial communications control register 2 is set. Software in the serial interrupt service routine must determine the cause and priority of the SCI interrupt by examining the interrupt flags and status bits in the SCI status register.

SPI INTERRUPTS

An interrupt in the SPI occurs when one of the interrupt flag bits in the serial peripheral status register is set, provided the I bit in the CCR is clear and the enable bit in the serial peripheral control register is set. Software in the serial peripheral interrupt service routine must determine the cause and priority of the SPI interrupt by examining the interrupt flag bits in the SPI status register.

LOW-POWER MODES

STOP

The STOP instruction places the MCU in its lowest power consumption mode. In the STOP mode, the internal oscillator is turned off, halting all internal processing including timer, SCI, and SPI operation (refer to Figure 9).

During the STOP mode, the TCR bits are altered to remove any pending timer interrupt request and to disable any further timer interrupts. The timer prescaler is cleared. The I bit in the CCR is cleared to enable external interrupts. All other registers and memory remain unaltered. All input/output lines remain unchanged. The processor can be brought out of the STOP mode only by an external interrupt or reset.

SCI during STOP Mode

When the MCU enters the STOP mode, the baud rate generator stops, halting all SCI activity. If the STOP instruction is executed during a transmitter transfer, that transfer is halted. If a low input to the $\overline{IRQ}$ pin is used to exit STOP mode, the transfer resumes. If the SCI receiver is receiving data and the STOP mode is entered, received data sampling stops because the baud rate generator stops, and all subsequent data is lost. For these reasons, all SCI transfers should be in the idle state when the STOP instruction is executed.

SPI during Stop Mode

When the MCU enters the STOP mode, the baud rate generator stops, terminating all master mode SPI operations. If the STOP instruction is executed during an SPI transfer, that transfer halts until the MCU exits the STOP mode by a low signal on the $\overline{IRQ}$ pin. If reset is used to exit the STOP mode, then the SPI control and status bits are cleared, and the SPI is disabled. If the MCU is in the slave mode when the STOP instruction is executed, the slave SPI continues to operate and can still accept data and clock information in addition to transmitting its own data back to a master device.

At the end of a possible transmission with a slave SPI in the STOP mode, no flags are set until a low on the $\overline{IRQ}$ pin wakes up the MCU. Caution should be observed when operating the SPI as a slave during the STOP mode because the protective circuitry (WCOL, MODF, etc.) is inactive.

WAIT

The WAIT instruction places the MCU in a low-power consumption mode, but the WAIT mode consumes more

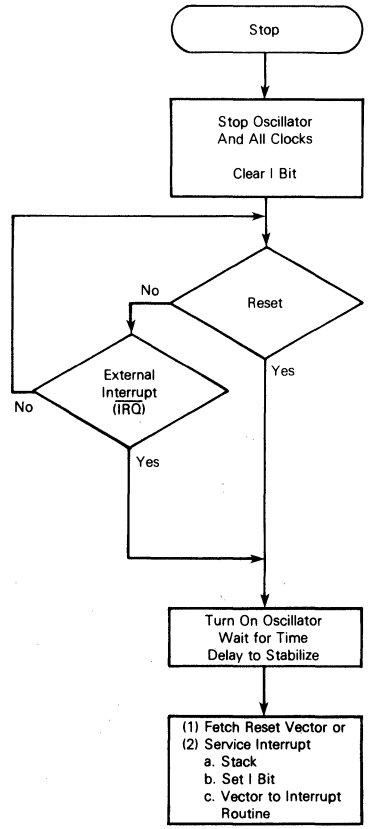


Figure 9. STOP Function Flowchart

power than the STOP mode. All CPU action is suspended, but the timer, SCI, and SPI remain active (refer to Figure 10). An interrupt from the timer, SCI, or SPI can cause the MCU to exit the WAIT mode.

During the WAIT mode, the I bit in the CCR is cleared to enable interrupts. All other registers, memory, and input/output lines remain in their previous state. The timer may be enabled to allow a periodic exit from the WAIT mode.

DATA RETENTION MODE

The contents of RAM and CPU registers are retained at supply voltages as low as 2.0 Vdc. This is called the data retention mode where the data is held, but the device is not guaranteed to operate. The MCU should be in $\overline{\text{RESET}}$ during data retention mode.

TIMER

The timer consists of a 16-bit, software-programmable counter driven by a fixed divide-by-four prescaler. This timer can be used for many purposes, including input waveform measurements while simultaneously generating an output waveform. Pulse widths can vary from several microseconds to many seconds. Refer to Figure 11 for a timer block diagram.

Because the timer has a 16-bit architecture, each specific functional segment (capability) is represented by two registers. These registers contain the high and low byte of that functional segment. Generally, accessing the low byte of a specific timer function allows full control of that function; however, an access of the high byte inhibits that specific timer function until the low byte is also accessed.

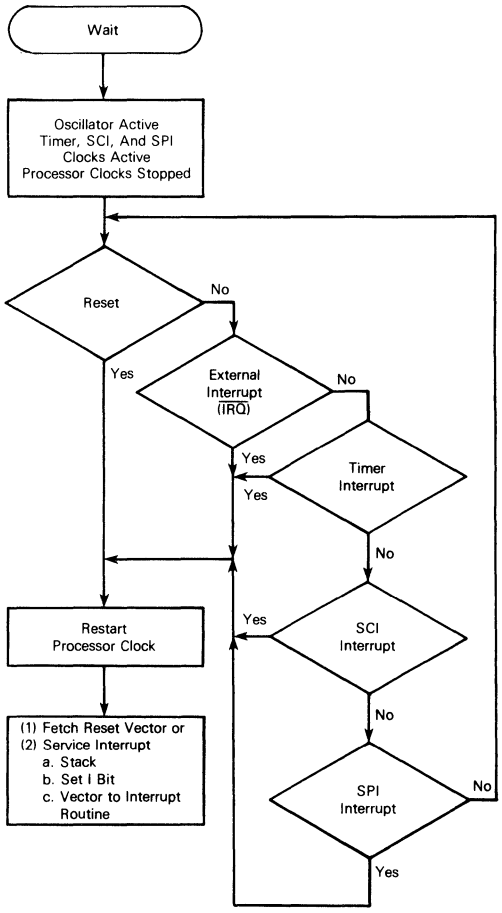


Figure 10. WAIT Function Flowchart

NOTE

The I bit in the CCR should be set while manipulating both the high and low byte register of a specific timer function to ensure that an interrupt does not occur.

COUNTER

The key element in the programmable timer is a 16-bit, free-running counter or counter register, preceded by a prescaler that divides the internal processor clock by four. The prescaler gives the timer a resolution of 2.0 microseconds if the internal bus clock is 2.0 MHz. The counter is incremented during the low portion of the internal bus clock. Software can read the counter at any time without affecting its value.

The double-byte, free-running counter can be read from either of two locations, $18–$19 (counter register) or $1A–$1B (counter alternate register). A read from only the least significant byte (LSB) of the free-running counter ($19, $1B) receives the count value at the time of the read. If a read of the free-running counter or counter alternate register first addresses the most significant byte (MSB) ($18, $1A), the LSB ($19, $1B) is transferred to a buffer. This buffer value remains fixed after the first MSB read, even if the user reads the MSB several times. This buffer is accessed when reading the free-running counter or counter alternate register LSB ($19 or $1B) and, thus, completes a read sequence of the total counter value. In reading either the free-running counter or counter alternate register, if the MSB is read, the LSB must also be read to complete the sequence.

The counter alternate register differs from the counter register in one respect: a read of the counter register MSB can clear the timer overflow flag (TOF). Therefore, the counter alternate register can be read at any time without the possibility of missing timer overflow interrupts due to clearing of the TOF.

The free-running counter is configured to $FFFC during reset and is always a read-only register. During a power-on reset, the counter is also preset to $FFFC and begins

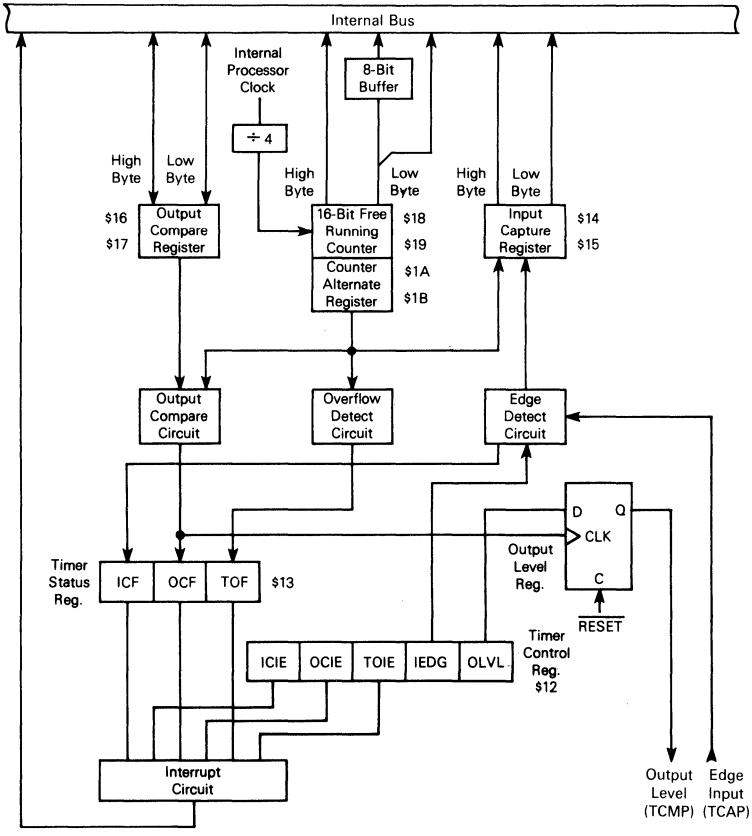


Figure 11. Timer Block Diagram

running after the oscillator start-up delay. Because the free-running counter is 16 bits preceded by a fixed divide-by-four prescaler, the value in the free-running counter repeats every 262,144 internal bus clock cycles. When the counter rolls over from $FFFF to $0000, the TOF bit is set. An interrupt can also be enabled when counter rollover occurs by setting its interrupt enable bit (TOIE).

OUTPUT COMPARE REGISTER

The 16-bit output compare register is made up of two 8-bit registers at locations $16 (MSB) and $17 (LSB). The output compare register is used for several purposes, such as indicating when a period of time has elapsed. All bits are readable and writable and are not altered by the timer hardware or reset. If the compare function is not needed, the two bytes of the output compare register can be used as storage locations.

The output compare register contents are compared with the contents of the free-running counter continually, and if a match is found, the corresponding output compare flag (OCF) bit is set and the corresponding output level (OLCL) bit is clocked to an output level register. The output compare register values and the output level bit should be changed after each successful comparison to establish a new elapsed timeout. An interrupt can also accompany a successful output compare provided the corresponding interrupt enable bit (OCIE) is set.

After a processor write cycle to the output compare register containing the MSB ($16), the output compare function is inhibited until the LSB ($17) is also written. The user must write both bytes (locations) if the MSB is written first. A write made only to the LSB ($17) will not inhibit the compare function. The free-running counter is updated every four internal bus clock cycles. The minimum time required to update the output compare register is a function of the program rather than the internal hardware.

The processor can write to either byte of the output compare register without affecting the other byte. The output level (OLVL) bit is clocked to the output level register regardless of whether the output compare flag (OCF) is set or clear.

INPUT CAPTURE REGISTER

Two 8-bit registers, which make up the 16-bit input capture register, are read-only and are used to latch the value of the free-running counter after the corresponding input capture edge detector senses a defined transition. The level transition which triggers the counter transfer is defined by the corresponding input edge bit (IEDG). Reset does not affect the contents of the input capture register.

The result obtained by an input capture will be one more than the value of the free-running counter on the rising edge of the internal bus clock preceding the external transition. This delay is required for internal synchronization. Resolution is one count of the free-running counter, which is four internal bus clock cycles.

The free-running counter contents are transferred to the input capture register on each proper signal transition regardless of whether the input capture flag (ICF) is set or clear. The input capture register always contains the free-running counter value that corresponds to the most recent input capture.

After a read of the input capture register ($14) MSB, the counter transfer is inhibited until the LSB ($15) is also read. This characteristic causes the time used in the input capture software routine and its interaction with the main program to determine the minimum pulse period.

A read of the input capture register LSB ($15) does not inhibit the free-running counter transfer since they occur on opposite edges of the internal bus clock.

TIMER CONTROL REGISTER (TCR) $12

The TCR is a read/write register containing five control bits. Three bits control interrupts associated with the timer status register flags ICF, OCF, and TOF.

7	6	5	4	3	2	1	0
ICIE	OCIE	TOIE	0	0	0	IEDG	OLVL
RESET: 0	0	0	0	0	0	U	0

ICIE — Input Capture Interrupt Enable
1 = Interrupt enabled
0 = Interrupt disabled

OCIE — Output Compare Interrupt Enable
1 = Interrupt enabled
0 = Interrupt disabled

TOIE — Timer Overflow Interrupt Enable
1 = Interrupt enabled
0 = Interrupt disabled

IEDG — Input Edge
Value of input edge determines which level transition on TCAP pin will trigger free-running counter transfer to the input capture register
1 = Positive edge
0 = Negative edge
Reset does not affect te IEDG bit (U = unaffected).

OLVL — Output Level
Value of output level is clocked into output level register by the next successful output compare and will appear on the TCMP pin
1 = High output
0 = Low output

Bits 2, 3, and 4 — Not used
Always read zero

TIMER STATUS REGISTER (TSR) $13

The TSR is a read-only register containing three status flag bits.

7	6	5	4	3	2	1	0
ICF	OCF	TOF	0	0	0	0	0
RESET: U	U	U	0	0	0	0	0

ICF — Input Capture Flag
1 = Flag set when selected polarity edge is sensed by input capture edge detector
0 = Flag cleared when TSR and input capture low register ($15) are accessed

OCF — Output Compare Flag
1 = Flag set when output compare register contents match the free-running counter contents
0 = Flag cleared when TSR and output compare low register ($17) are accessed
TOF — Timer Overflow Flag
1 = Flag set when free-running counter transition from $FFFF to $0000 occurs
0 = Flag cleared when TSR and counter low register ($19) are accessed
Bits 0–4 — Not used
Always read zero

Accessing the timer status register satisfies the first condition required to clear status bits. The remaining step is to access the register corresponding to the status bit.

A problem can occur when using the timer overflow function and reading the free-running counter at random times to measure an elapsed time. Without incorporating the proper precautions into software, the timer overflow flag could unintentionally be cleared if:

1) The timer status register is read or written when TOF is set, and
2) The LSB of the free-running counter is read but not for the purpose of servicing the flag.

The counter alternate register at address $1A and $1B contains the same value as the free-running counter (at address $18 and $19); therefore, this alternate register can be read at any time without affecting the timer overflow flag in the timer status register.

TIMER DURING WAIT MODE

The CPU clock halts during the WAIT mode, but the timer remains active. An interrupt from the timer causes the processor to exit the WAIT mode.

TIMER DURING STOP MODE

In the STOP mode, the timer stops counting and holds the last count value if STOP is exited by an interrupt. If RESET is used, the counter is forced to $FFFC. During STOP, if at least one valid input capture edge occurs at the TCAP pin, the input capture detect circuit is armed. This does not set any timer flags nor wake up the MCU, but when the MCU does wake up, there is an active input capture flag and data from the first valid edge that occurred during the STOP mode. If RESET is used to exit STOP mode, then no input capture flag or data remains, even if a valid input capture edge occurred.

SERIAL COMMUNICATIONS INTERFACE

A full-duplex asynchronous SCI is provided with a standard NRZ format and a variety of baud rates. The SCI transmitter and receiver are functionally independent but use the same data format and baud rate. The terms baud and bit rate are used synonymously in the following description.

SCI TWO-WIRE SYSTEM FEATURES

- Standard NRZ (mark/space) format
- Advanced error detection method includes noise detection for noise duration of up to one-sixteenth bit time
- Full-duplex operation (simultaneous transmit and receive)
- Software programmable for one of 32 different baud rates
- Software-selectable word length (eight- or nine-bit words)
- Separate transmitter and receiver enable bits
- SCI may be interrupt driven
- Four separate interrupt conditions

SCI RECEIVER FEATURES

- Receiver wake-up function (idle or address bit)
- Idle line detect
- Framing error detect
- Noise detect
- Overrun detect
- Receiver data register full flag

SCI TRANSMITTER FEATURES

- Transmit data register empty flag
- Transmit complete flag
- Break send

Any SCI two-wire system requires receive data in (RDI) and transmit data out (TDO).

DATA FORMAT

Receive data in (RDI) or transmit data out (TDO) is the serial data presented between the internal data bus and the output pin (TDO) and between the input pin (RDI) and the internal data bus. Data format is as shown for the NRZ in Figure 12.

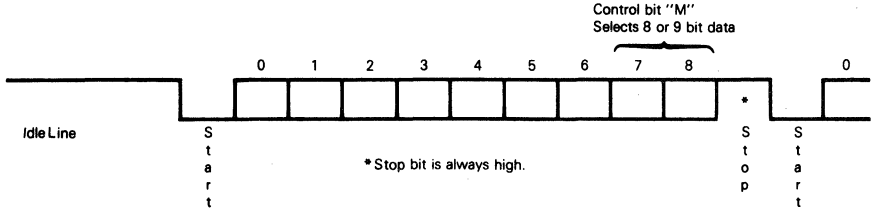


Figure 12. Data Format

WAKE-UP FEATURE

In a typical multiprocessor configuration, the software protocol will usually identify the addressee(s) at the beginning of the message. To permit uninterested MPUs to ignore the remainder of the message, a wake-up feature is included, whereby all further SCI receiver flag (and interrupt) processing can be inhibited until its data line returns to the idle state. An SCI receiver is re-enabled by an idle string of at least ten (or eleven) consecutive ones. Software for the transmitter must provide for the required idle string between consecutive messages and prevent it from occurring within messages.

A second wake-up method is available in which sleeping SCI receivers can be awakened by a logic one in the high-order bit of a received character.

RECEIVE DATA IN

Receive data in (RDI) is the serial data which is presented from the input pin via the SCI to the receive data register (RDR). While waiting for a start bit, the receiver samples the input at a rate 16 times higher than the set baud rate. This increased rate is referred to as the RT rate. When the input (idle) line is detected low, it is tested for three more sample times. If at least two of these three samples detect a logic low, a valid start bit is assumed to be detected. If in two or more samples, a logic high is detected, the line is assumed to be idle. The receive clock generator is controlled by the baud rate register (see Figure 13); however, the SCI is synchronized by the start bit independent of the transmitter. Once a valid start bit is detected, the start bit, each data bit, and the stop bit are each sampled three times. The value of the bit is determined by voting logic, which takes the value of a majority of samples. A noise flag is set when all three samples on a valid start bit, data bit, or stop bit do not agree. A noise flag is also set when the start verification samples do not agree.

START BIT DETECTION FOLLOWING A FRAMING ERROR

If there has been a framing error (FE) without detection of a break (10 zeros for 8-bit format or 11 zeros for a 9-bit format), the circuit continues to operate as if there actually were a stop bit, and the start edge will be placed artificially. The last bit received in the data shift register is inverted to a logic one, and the three logic-one start qualifiers are forced into the sample shift register during the interval when detection of a start bit is anticipated; therefore, the start bit will be accepted no sooner than it is anticipated.

If the receiver detects that a break (RDRF = 1, FE = 1, receiver data register = $00) produced the framing error, the start bit will not be artificially induced, and the receiver must actually receive a logic one before start.

TRANSMIT DATA OUT

Transmit data out (TDO) is the serial data presented from the transmit data register (TDR) via the SCI to the output pin. The transmitter generates a bit time by using a derivative of the RT clock, producing a transmission rate equal to one-sixteenth that of the receiver sample clock.

FUNCTIONAL DESCRIPTION

A block diagram of the SCI is shown in Figure 13. The user has option bits in the serial communications control register 1 (SCCR1) to determine the SCI wake-up method and data word length. Serial communications control register 2 (SCCR2) provides control bits that individually enable/disable the transmitter or receiver, enable system interrupts, and provide wake-up enable, and send break code bits. The baud rate register bits allow the user to select different baud rates, which are used as the rate control for the transmitter and receiver.

Data transmission is initiated by a write to the serial communications data register (SCDAT). Provided the transmitter is enabled, data stored in the SCDAT is transferred to the transmit data shift register. This data transfer sets the SCI status register (SCSR) transmit data register empty (TDRE) bit and generates an interrupt if the transmit interrupt is enabled. Data transfer to the transmit data shift register is synchronized with the bit rate clock. All data is transmitted LSB first. Upon completion of data transmission, the transmission complete (TC) bit is set (provided no pending data, preamble, or break code is sent), and an interrupt is generated if the transmit complete interrupt is enabled. If the transmitter is disabled, and the data, preamble, or break code has been sent, the TC bit will also be set, which will also generate an interrupt if the TCIE bit is set. If the transmitter is disabled in the middle of a transmission, that character will be completed before the transmitter gives up control of the TDO pin.

When the SCDAT is read, it contains the last data byte received, provided that the receiver is enabled. The SCSR receive data register full (RDRF) bit is set to indicate that a data byte is transferred from the input serial shift register to the SCDAT, which can cause an interrupt if the receiver interrupt is enabled. Data transfer from the input serial shift register to the SCDAT is synchronized by the receiver bit rate clock. The SCSR overrun (OR), noise flag (NF), or FE bits are set if data reception errors occur.

An idle line interrupt is generated if the idle line interrupt is enabled and the SCSR IDLE bit (which detects idle line transmission) is set. This allows a receiver that is not in the wake-up mode to detect the end of a message, the preamble of a new message, or to resynchronize with the transmitter. A valid character must be received before the idle line condition for the IDLE bit to be set and for an idle line interrupt to be generated.

REGISTERS

There are five registers used in the SCI; the internal configuration of these registers is discussed in the following paragraphs.

Serial Communications Data Register (SCDAT) $11

The SCDAT is a read/write register used to receive and transmit SCI data.

7	6	5	4	3	2	1	0
SCD7	SCD6	SCD5	SCD4	SCD3	SCD2	SCD1	SCD0
RESET:							
U	U	U	U	U	U	U	U

As shown in Figure 13, SCDAT functions as two separate registers. The transmit data register (TDR) provides the parallel interface from the internal data bus to the transmit shift register. The receive data register (RDR) provides the interface from the receive shift register to the internal data bus.

Serial Communications Control Register 1 (SCCR1) $OE

The SCCR1 provides control bits that determine word length and select the wake-up method.

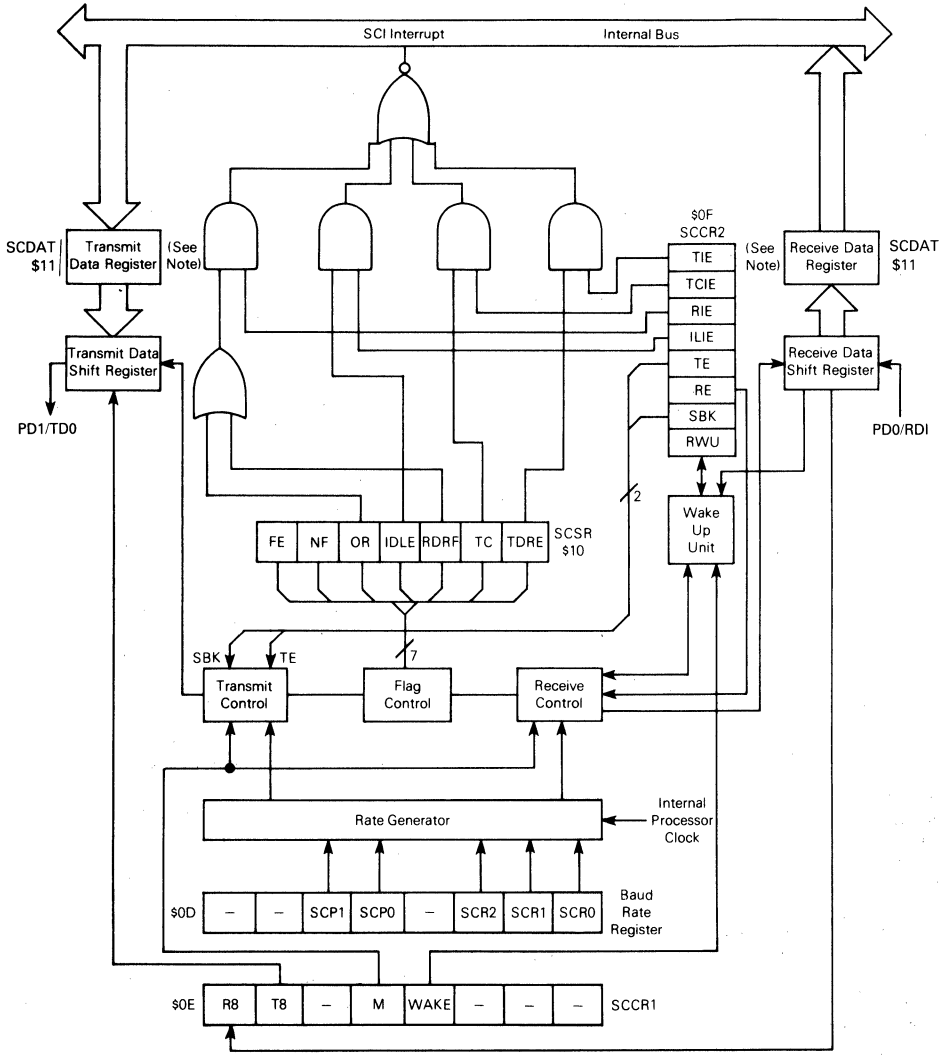
7	6	5	4	3	2	1	0
R8	T8	—	M	WAKE	—	—	—
RESET:							
U	U	—	U	U	—	—	—

R8 — Receive Data Bit 8

R8 bit provides storage location for the ninth bit in the receive data byte (if M = 1).

T8 — Transmit Data Bit 8

T8 bit provides storage location for the ninth bit in the transmit data byte (if M = 1).



NOTE: The Serial Communications Data Register (SCDAT) is controlled by the internal R/W signal. It is the transmit data register when written and receive data register when read.

Figure 13. SCI Block Diagram

M — SCI Character Word Length
1 = one start bit, nine data bits, one stop bit
0 = one start bit, eight data bits, one stop bit
WAKE — Wake-Up Select
Wake bit selects the receiver wake-up method.
1 = Address bit (most significant bit)
0 = Idle line condition
Bits 0–2, and 5 — Not used
Can read either one or zero

The address bit is dependent on both the wake-bit and the M-bit level. Additionally, the receiver does not use the wake-up feature unless the RWU control bit in SCCR2 is set.

Wake	M	Receiver Wake-Up
0	X	Detection of an idle line allows the next data byte received to cause the receive data register to fill and produce an RDRF flag.
1	0	Detection of a received one in the eighth data bit allows an RDRF flag and associated error flags.
1	1	Detection of a received one in the ninth data bit allows an RDRF flag and associated error flags.

Serial Communications Control Register 2 (SCCR2) $0F

The SCCR2 provides control of individual SCI functions such as interrupts, transmit/receive enabling, receiver wake-up, and break code.

7	6	5	4	3	2	1	0
TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
RESET: 0	0	0	0	0	0	0	0

TIE — Transmit Interrupt Enable
1 = SCI interrupt enabled
0 = TDRE interrupt disabled
TCIE — Transmit Complete Interrupt Enable
1 = SCI interrupt enabled
0 = TC interrupt disabled
RIE — Receive Interrupt Enable
1 = SCI interrupt enabled
0 = RDRF and OR interrupts disabled
ILIE — Idle Line Interrut Enable
1 = SCI interrupt enabled
0 = Idle interrupt disabled
TE — Transmit Enable
1 = Transmit shift register output is applied to the TD0 line. Depending upon the SCCR1 M bit, a preamble of 10 (M = 0) or 11 (M = 1) consecutive ones is transmitted.
0 = Transmitter disabled after last byte is loaded in the SCDAT and TDRE is set. After last byte is transmitted, TD0 line becomes a high-impedance line.
RE — Receive Enable
1 = Receiver shift register input is applied to the RDI line.
0 = Receiver disabled and RDRF, IDLE, OR, NF, and FE status bits are inhibited.
RWU — Receiver Wake-Up
1 = Places receiver in sleep mode and enables wake-up function
0 = Wake-up function disabled after receiving data word with MSB set (if WAKE = 1)
Wake-up function also disabled after receiving 10 (M = 0) or 11 (M = 1) consecutive ones (if WAKE = 0)
SBK — Send Break
1 = Transmitter continually sends blocks of zeros (sets of 10 or 11) until cleared. Upon completion of break code, transmitter sends one high bit for recognition of valid start bit.
0 = Transmitter sends 10 (M = 0) or 11 (M = 1) zeros then reverts to an idle state or continues sending data. If transmitter is empty and idle, setting and clearing the SBK bit may queue up to two character times of break because the first break transfers immediately to the shift register, and the second is queued into the parallel transmit buffer.

Serial Communications Status Register (SCSR) $10

The SCSR provides inputs to the SCI interrupt logic circuits. Noise flag and framing error bits are also contained in the SCSR.

7	6	5	4	3	2	1	0
TDRE	TC	RDRF	IDLE	OR	NF	FE	—
RESET: 1	1	0	0	0	0	0	—

TDRE — Transmit Data Register (TDR) Empty
1 = TDR contents transferred to the transmit data shift register
0 = TDR still contains data. TDRE is cleared by reading the SCSR (with TDRE = 1), followed by a write to the TDR.
TC — Transmit Complete
1 = Indicates end of data frame, preamble, or break condition has occurred
0 = TC bit cleared by reading the SCSR (with TC = 1), followed by a write to the TDR
RDRF — Receive Data Register (RDR) Full
1 = Receive data shift register contents transferred to the RDR
0 = Receive data shift register transfer did not occur. RDRF is cleared by reading the SCSR (with RDRF = 1) followed by a read of the RDR
IDLE — Idle Line Detect
1 = Indicates receiver has detected an idle line
0 = IDLE is cleared by reading the SCSR (with IDLE = 1), followed by a read of the RDR. Once IDLE is cleared, IDLE cannot be set until RDI line becomes active and idle again.
OR — Overrun Error
1 = Indicates receive data shift register data is sent to a full RDR (RDRF = 1). Data causing the overrun is lost, and RDR data is not disturbed.
0 = OR is cleared by reading the SCSR (with OR = 1), followed by a read of the RDR.

NF — Noise Flag
1 = Indicates noise is present on the receive bits, including the start and stop bits. NF is not set until RDRF = 1.
0 = NF is cleared by reading the SCSR (with NF = 1), followed by a read of the RDR.

FE — Framing Error
1 = Indicates stop bit not detected in received data character. FE is set the same time RDRF is set. If received byte causes both framing and overrun errors, processor will only recognize the overrun error. Further data transfer into the RDR is inhibited until FE is cleared.
0 = NF is cleared by reading the SCSR (with FE = 1), followed by a read of the RDR.

Bit 0 — Not used
Can read either one or zero

Baud Rate Register $0D

The baud rate register is used to select the SCI transmitter and receiver baud rate. SCP0 and SCP1 prescaler bits are used in conjunction with the SCR0 through SCR2 baud rate bits to provide multiple baud rate combinations for a given crystal frequency. Bits 3, 6, and 7 always read zero.

7	6	5	4	3	2	1	0
—	—	SCP1	SCP0	—	SCR2	SCR1	SCR0
RESET:							
—	—	0	0	—	U	U	U

SCP0 — SCI Prescaler Bit 0
SCP1 — SCI Prescaler Bit 1
Two prescaler bits are used to increase the range of standard baud rates controlled by the SCR0–SCR2 bits. Prescaler internal processor clock division versus bit levels are listed in Table 2.

SCR0 — SCI Baud Rate Bit 0
SCR1 — SCI Baud Rate Bit 1
SCR2 — SCI Baud Rate Bit 2
Three baud rate bits are used to select the baud rates of the SCI transmitter and SCI receiver. Baud rates versus bit levels are listed in Table 3.

Tables 3 and 4 tabulate the divide chain used to obtain the baud rate clock (transmit clock). The actual divider chain is controlled by the combined SCP0-SCP1 and SCR0-SCR2 bits in the baud rate register. All divided frequencies shown in Table 3 represent the final baud rate resulting from the internal processor clock division shown in the divided-by column only (prescaler division only). Table 4 lists the prescaler output divided by the action of the SCI select bits (SCR0-SCR2). For example, assume that a 9600-Hz baud rate is required with a 2.4576-MHz external crystal. In this case, the prescaler bits (SCP0-SCP1) could be configured as a divide-by-one or a divide-by-four. If a divide-by-four prescaler is used, then the SCR0-SCR2 bits must be configured as a divide-by-two. Using the same crystal, the 9600 baud rate can be obtained with a prescaler divide-by-one and the SCR0-SCR2 bits configured for a divide-by-eight.

Table 3. Prescaler Highest Baud Rate Frequency Output

SCP Bit		Clock* Divided By	Crystal Frequency MHz				
1	0		4.194304	4.0	2.4576	2.0	1.8432
0	0	1	131.072 kHz	125.000 kHz	76.80 kHz	62.50 kHz	57.60 kHz
0	1	3	43.691 kHz	41.666 kHz	25.60 kHz	20.833 kHz	19.20 kHz
1	0	4	32.768 kHz	31.250 kHz	19.20 kHz	15.625 kHz	14.40 kHz
1	1	13	10.082 kHz	9600 Hz	5.907 kHz	4800 Hz	4430 Hz

*Refers to the internal processor clock.

NOTE: The divided frequencies shown in Table 3 represent baud rates which are the highest transmit baud rate (Tx) that can be obtained by a specific crystal frequency and only using the prescaler division. Lower baud rates may be obtained by providing a further division using the SCI rate select bits as shown below for some representative prescaler outputs.

Table 4. Transmit Baud Rate Output for a Given Prescaler Output

SCR Bits			Divided By	Representative Highest Prescaler Baud Rate Output				
2	1	0		131.072 kHz	32.768 kHz	76.80 kHz	19.20 kHz	9600 Hz
0	0	0	1	131.072 kHz	32.768 kHz	76.80 kHz	19.20 kHz	9600 Hz
0	0	1	2	65.536 kHz	16.384 kHz	38.40 kHz	9600 Hz	4800 Hz
0	1	0	4	32.768 kHz	8.192 kHz	19.20 kHz	4800 Hz	2400 Hz
0	1	1	8	16.384 kHz	4.096 kHz	9600 Hz	2400 Hz	1200 Hz
1	0	0	16	8.192 kHz	2.048 kHz	4800 Hz	1200 Hz	600 Hz
1	0	1	32	4.096 kHz	1.024 kHz	2400 Hz	600 Hz	300 Hz
1	1	0	64	2.048 kHz	512 Hz	1200 Hz	300 Hz	150 Hz
1	1	1	128	1.024 kHz	256 Hz	600 Hz	150 Hz	75 Hz

NOTE: Table 4 illustrates how the SCI select bits can be used to provide lower transmitter baud rates by further dividing the prescaler output frequency. The five examples are only representative samples. In all cases, the baud rates shown are transmit baud rates (transmit clock), and the receive clock is 16 times higher in frequency than the actual baud rate.

SERIAL PERIPHERAL INTERFACE

The serial peripheral interface (SPI) is an interface built into the MCU which allows several MCUs or MCUs plus peripherals to be interconnected within the same black box. In the SPI format, the clock is not included in the data stream and must be furnished as a separate signal. An SPI system may consist of one master MCU and several slaves (Figure 14) or MCUs that can be either masters or slaves.

Features:

- Full-duplex, three-wire synchronous transfers
- Master or slave operation
- 1.05 MHz (maximum) master bit frequency
- 2.1 MHz (maximum) slave bit frequency
- Four programmable master bit rates
- Programmable clock polarity and phase
- End-of-transmission interrupt flag
- Write collision flag protection
- Master-master mode fault protection capability

SIGNAL DESCRIPTION

The four basic signals (MOSI, MISO, SCK, and $\overline{SS}$) are described in the following paragraphs. Each signal function is described for both master and slave mode.

Master Out, Slave In

The master out, slave in (MOSI) line is configured as an output in a master device and as an input in a slave device. The MOSI line is one of two lines that transfer serial data in one direction with the most significant bit sent first.

Master In, Slave Out

The master in, slave out (MISO) line is configured as an input in a master device and as an output in a slave device. The MISO is one of two lines that transfer serial data in one direction with the most significant bit sent first. The MISO line of a slave device is placed in a high-impedance state if slave is not selected ($\overline{SS}=1$).

Serial Clock

The serial clock (SCK) is used to synchronize both data in and out of a device via the MOSI and MISO lines. The master and slave devices can exchange a byte of information during a sequence of eight clock cycles. Since SCK is generated by the master device, this line becomes an input on a slave device.

As shown in Figure 15, four possible timing relationships may be chosen by using control bits CPOL and CPHA in the serial peripheral control register (SPCR). Both master and slave devices must operate with the same timing.

Two bits (SPR0 and SPR1) in the SPCR of the master device select the clock rate. In a slave device, SPR0 and SPR1 have no effect on SPI operation.

Slave Select

The slave select ($\overline{SS}$) input line selects a slave device. The $\overline{SS}$ line must be low prior to data transactions and must stay low for the duration of the transaction. The $\overline{SS}$ line on the master must be tied high; if the $\overline{SS}$ line goes low, a mode fault error flag (MODF) is set in the serial peripheral status register (SPSR).

When CPHA = 0, the shift clock is the OR of $\overline{SS}$ with SCK. In this clock phase mode, $\overline{SS}$ must go high between successive characters in an SPI message. When CPHA = 1, $\overline{SS}$ must go high between successive characters in an SPI message. When CPHA = 1, $\overline{SS}$ may be left low for several SPI characters. In cases where there is only one SPI slave MCU, the slave MCU $\overline{SS}$ line could be tied to V_{SS} as long as CPHA = 1 clock modes are used.

FUNCTIONAL DESCRIPTION

A block diagram of the SPI is shown in Figure 16. In a master configuration, the CPU sends a signal to the master start logic, which originates an SPI clock (SCK) based on the internal processor clock. As a master device, data is parallel loaded into the 8-bit shift register from the internal bus during a write cycle and then serially shifted

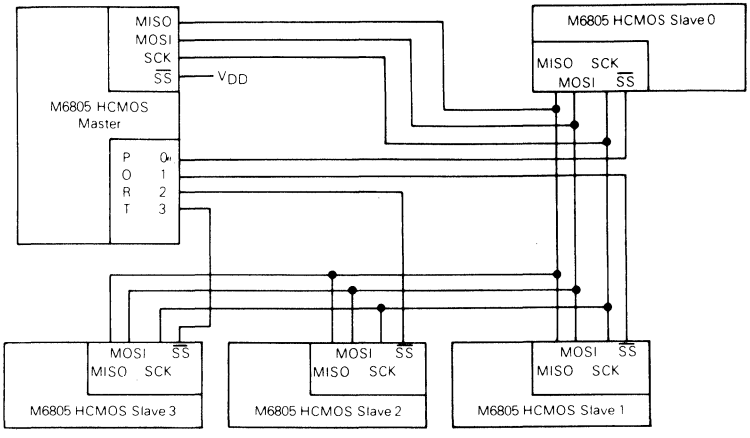


Figure 14. Master-Slave System Configuration

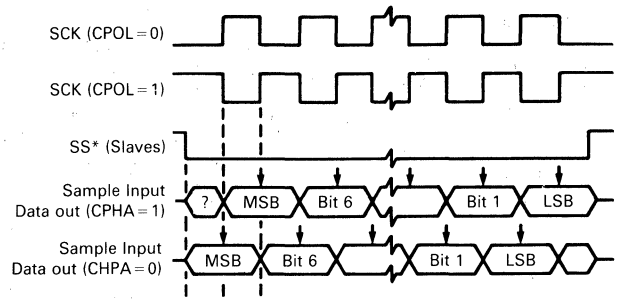


Figure 15. Data Clock Timing Diagram

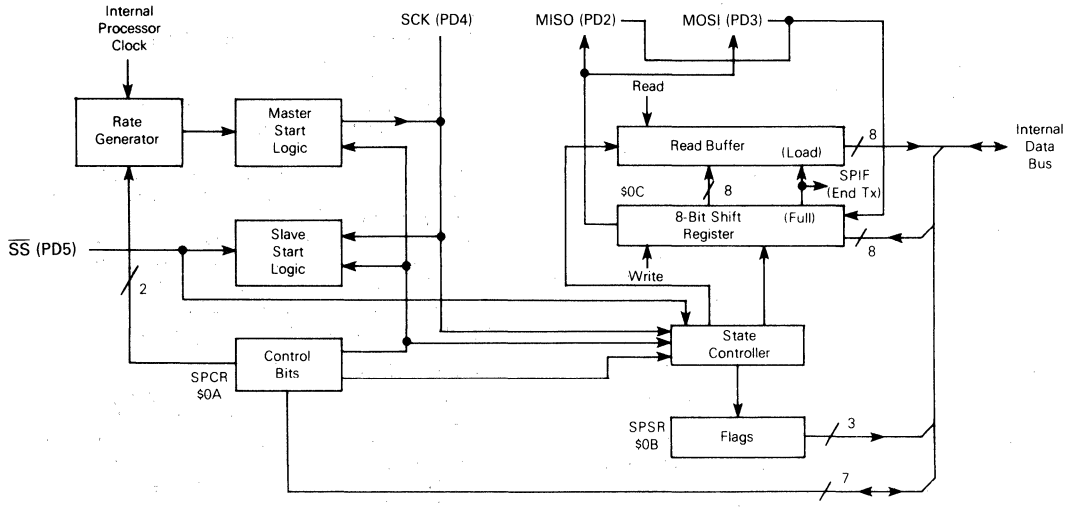


Figure 16. SPI Block Diagram

via the MOSI pin to the slave devices. During a read cycle, data is applied serially from a slave device via the MISO pin to the 8-bit shift register. Data is then parallel transferred to the read buffer and made available to the internal data bus during a CPU read cycle.

In a slave configuration, the slave start logic receives a logic low at the $\overline{SS}$ pin and a clock input at the SCK pin. This synchronizes the slave with the master. Data from the master is received serially at the slave MOSI pin and shifted into the 8-bit shift register for a parallel transfer to the read buffer. During a write cycle, data is parallel loaded into the 8-bit shift register from the internal data bus, awaiting the clocks from the master to shift out serially to the MISO pin and then to the master device.

Figure 17 illustrates the MOSI, MISO, SCK, and $\overline{SS}$ master-slave interconnections.

REGISTERS

There are three registers in the SPI that provide control, status, and data storage functions. These registers, the serial peripheral control register (SPCR), serial peripheral status register (SPSR), and serial peripheral data I/O register (SPDR), are described in the following paragraphs.

Serial Peripheral Control Register $0A

The SPCR provides control of individual SPI functions such as interrupt and system enabling/disabling, master/slave mode select, and clock polarity/phase/rate select.

7	6	5	4	3	2	1	0
SPIE	SPE	—	MSTR	CPOL	CPHA	SPR1	SPR0
RESET:							
0	0	—	0	U	U	U	U

SPIE — Serial Peripheral Interrupt Enable
1 = SPI interrupt enabled
0 = SPI interrupt disabled
SPE — Serial Peripheral System Enable
1 = SPI system on
0 = SPI system off

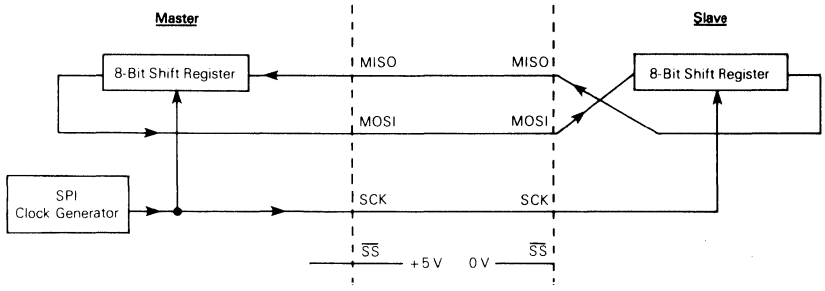


Figure 17. SPI Master-Slave Interconnections

MSTR — Master Mode Select
1 = Master mode
0 = Slave mode

CPOL — Clock Polarity
Clock polarity bit controls the clock value and is used in conjunction with the clock phase (CPHA) bit.
1 = SCK line idles high
0 = SCK line idles in low state

CPHA — Clock Phase
Clock phase bit along with CPOL controls the clock-data relationship between the master and slave devices. CPOL selects one of two clocking protocols.
1 = $\overline{SS}$ is an output enable control.
0 = Shift clock is the OR of SCK with $\overline{SS}$.
When $\overline{SS}$ is low, first edge of SCK invokes first data sample.

SPR0, SPR1 — SPI Clock Rate Bits
Two clock rate bits are used to select one of four clock rates to be used as SCK in the master mode. In the slave mode, the two clock rate bits have no effect. Clock rate selection is shown in the following table.

Bit 5 – Not used
Can read either one or zero

SPI Clock Rate Selection

SPR1	SPR0	Internal Processor Clock Divided By
0	0	2
0	1	4
1	0	16
1	1	32

Serial Peripheral Status Register $0B

The SPSR contains three status bits.

7	6	5	4	3	2	1	0
SPIF	WCOL	—	MODF	—	—	—	—
RESET:							
0	0	—	0	—	—	—	—

SPIF — Serial Peripheral Data Transfer Flag
1 = Indicates data transfer completed between processor and external device.
(If SPIF = 1 and SPIE = 1, SPI interrupt is enabled.)
0 = Clearing is accomplished by reading SPSR (with SPIF = 1) followed by SPDR access.

WCOL — Write Collision
1 = Indicates an attempt is made to write to SPDR while data transfer is in process.
0 = Clearing is accomplished by reading SPSR (with WCOL = 1), followed by SPDR access.

MODF — Mode Fault Flag
1 = Indicates multi-master system control conflict.
0 = Clearing is accomplished by reading SPSR (with MODF = 1), followed by a write to the SPCR.

Bits 0–3, and 5 — Not used
Can read either zero or one

Serial Peripheral Data I/O Register $0C

The SPDR is a read/write register used to receive and transmit SPI data.

7	6	5	4	3	2	1	0
SPD7	SPD6	SPD5	SPD4	SPD3	SPD2	SPD1	SPD0
RESET:							
U	U	U	U	U	U	U	U

A write to the SPDR places data directly into the shift register for transmission. Only a write to this register will initiate transmission/reception of another byte and will only occur in the master device. On completion of byte transmission, the SPIF status bit is set in both master and slave devices.

A read to the SPDR causes the buffer to be read. The first SPIF status bit must be cleared by the time a second data transfer from the shift register to the read buffer begins, or an overrun condition will exist. In overrun cases, the byte causing the overrun is lost.

INSTRUCTION SET

The MCU has a set of 62 basic instructions. They can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

This MCU uses all the instructions available in the M146805 CMOS Family plus one more: the unsigned multiply (MUL) instruction. This instruction allows unsigned multiplication of the contents of the accumulator

(A) and the index register (X). The high-order product is then stored in the index register, and the low-order product is stored in the accumulator. A detailed definition of the MUL instruction is shown below.

Operation	X:A X*A			
Description	Multiplies the eight bits in the index register by the eight bits in the accumulator to obtain a 16-bit unsigned number in the concatenated accumulator and index register			
Condition Codes	H: Cleared I: Not affected N: Not affected Z: Not affected C: Cleared			
Source	MUL			
Form(s)	Addressing Mode	Cycles	Bytes	Opcode
	Inherent	11	1	$42

3

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following instruction list.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (Twos Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST
Multiply	MUL

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two-byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Higher	BHI
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
Branch if Higher or Same	BHS
Branch if Carry Set	BCS
Branch if Lower	BLO
Branch if Not Equal	BNE
Branch if Equal	BEQ
Branch if Half Carry Clear	BHCC
Branch if Half Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Bit is Clear	BMC
Branch if Interrupt Mask Bit is Set	BMS
Branch if Interrupt Line is Low	BIL
Branch if Interrupt Line is High	BIH
Branch to Subroutine	BSR

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any writable bit which resides in the first 256 bytes of the memory space where all port registers, port DDRs, timer, timer control, ROM, and on-chip RAM reside. An additional feature allows the software to test and branch on the state

of any bit within these 256 locations. The bit set, bit clear and bit test, and branch functions are all implemented with a single instruction. For test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch if Bit n is Set	BRSET n (n=0 . . . 7)
Branch if Bit n is Clear	BRCLR n (n=0 . . . 7)
Set Bit n	BSET n (n=0 . . . 7)
Clear Bit n	BCLR n (n=0 . . . 7)

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No-Operation	NOP
Stop	STOP
Wait	WAIT

OPCODE MAP SUMMARY

Table 5 is an opcode map for the instructions used on the MCU.

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single byte instructions; the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. One- or two-byte direct addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction. When using the Motorola assembler, the user need not specify whether an instruction uses direct or extended addressing. The assembler automatically selects the shortest form of the instruction.

RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −126 to +129 from the opcode address. The programmer need not calculate the offset when using the Motorola assembler, since it calculates the proper offset and checks to see that it is within the span of the branch.

INDEXED, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. This addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this two-byte instruction, K would typically be in X with the address of

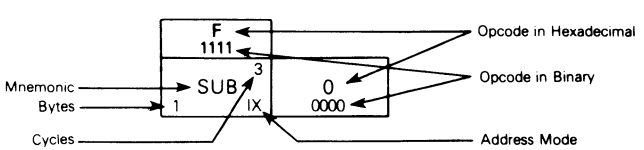
Table 5. Opcode Map

Hi / Low	Bit Manipulation		Branch	Read/Modify/Write					Control		Register/Memory					
	BTB 0 0000	BSC 1 0001	REL 2 0010	DIR 3 0011	INH 4 0100	INH 5 0101	IX1 6 0110	IX 7 0111	INH 8 1000	INH 9 1001	IMM A 1010	DIR B 1011	EXT C 1100	IX2 D 1101	IX1 E 1110	IX F 1111	Hi / Low
0 0000	BRSET0 3 BTB	BSET0 2 BSC	BRA 2 REL	NEG 5 2 DIR	NEGA 3 1 INH	NEGX 3 1 INH	NEG 6 2 IX1	NEG 5 1 IX	RTI 9 1 INH		SUB 2 2 IMM	SUB 3 2 DIR	SUB 4 3 EXT	SUB 5 3 IX2	SUB 4 2 IX1	SUB 3 1 IX	0 0000
1 0001	BRCLR0 3 BTB	BCLR0 2 BSC	BRN 2 REL						RTS 6 1 INH		CMP 2 2 IMM	CMP 3 2 DIR	CMP 4 3 EXT	CMP 5 3 IX2	CMP 4 2 IX1	CMP 3 1 IX	1 0001
2 0010	BRSET1 3 BTB	BSET1 2 BSC	BHI 2 REL		MUL 11 1 INH						SBC 2 2 IMM	SBC 3 2 DIR	SBC 4 3 EXT	SBC 5 3 IX2	SBC 4 2 IX1	SBC 3 1 IX	2 0010
3 0011	BRCLR1 3 BTB	BCLR1 2 BSC	BLS 2 REL	COM 5 2 DIR	COMA 3 1 INH	COMX 3 1 INH	COM 6 2 IX1	COM 5 1 IX	SWI 10 1 INH		CPX 2 2 IMM	CPX 3 2 DIR	CPX 4 3 EXT	CPX 5 3 IX2	CPX 4 2 IX1	CPX 3 1 IX	3 0011
4 0100	BRSET2 3 BTB	BSET2 2 BSC	BCC 2 REL	LSR 5 2 DIR	LSRA 3 1 INH	LSRX 3 1 INH	LSR 6 2 IX1	LSR 5 1 IX			AND 2 2 IMM	AND 3 2 DIR	AND 4 3 EXT	AND 5 3 IX2	AND 4 2 IX1	AND 3 1 IX	4 0100
5 0101	BRCLR2 3 BTB	BCLR2 2 BSC	BCS 2 REL								BIT 2 2 IMM	BIT 3 2 DIR	BIT 4 3 EXT	BIT 5 3 IX2	BIT 4 2 IX1	BIT 3 1 IX	5 0101
6 0110	BRSET3 3 BTB	BSET3 2 BSC	BNE 2 REL	ROR 5 2 DIR	RORA 3 1 INH	RORX 3 1 INH	ROR 6 2 IX1	ROR 5 1 IX			LDA 2 2 IMM	LDA 3 2 DIR	LDA 4 3 EXT	LDA 5 3 IX2	LDA 4 2 IX1	LDA 3 1 IX	6 0110
7 0111	BRCLR3 3 BTB	BCLR3 2 BSC	BEQ 2 REL	ASR 5 2 DIR	ASRA 3 1 INH	ASRX 3 1 INH	ASR 6 2 IX1	ASR 5 1 IX		TAX 2 1 INH		STA 4 2 DIR	STA 5 3 EXT	STA 6 3 IX2	STA 5 2 IX1	STA 4 1 IX	7 0111
8 1000	BRSET4 3 BTB	BSET4 2 BSC	BHCC 2 REL	LSL 5 2 DIR	LSLA 3 1 INH	LSLX 3 1 INH	LSL 6 2 IX1	LSL 5 1 IX		CLC 2 1 INH	EOR 2 2 IMM	EOR 3 2 DIR	EOR 4 3 EXT	EOR 5 3 IX2	EOR 4 2 IX1	EOR 3 1 IX	8 1000
9 1001	BRCLR4 3 BTB	BCLR4 2 BSC	BHCS 2 REL	ROL 5 2 DIR	ROLA 3 1 INH	ROLX 3 1 INH	ROL 6 2 IX1	ROL 5 1 IX		SEC 2 1 INH	ADC 2 2 IMM	ADC 3 2 DIR	ADC 4 3 EXT	ADC 5 3 IX2	ADC 4 2 IX1	ADC 3 1 IX	9 1001
A 1010	BRSET5 3 BTB	BSET5 2 BSC	BPL 2 REL	DEC 5 2 DIR	DECA 3 1 INH	DECX 3 1 INH	DEC 6 2 IX1	DEC 5 1 IX		CLI 2 1 INH	ORA 2 2 IMM	ORA 3 2 DIR	ORA 4 3 EXT	ORA 5 3 IX2	ORA 4 2 IX1	ORA 3 1 IX	A 1010
B 1011	BRCLR5 3 BTB	BCLR5 2 BSC	BMI 2 REL							SEI 2 1 INH	ADD 2 2 IMM	ADD 3 2 DIR	ADD 4 3 EXT	ADD 5 3 IX2	ADD 4 2 IX1	ADD 3 1 IX	B 1011
C 1100	BRSET6 3 BTB	BSET6 2 BSC	BMC 2 REL	INC 5 2 DIR	INCA 3 1 INH	INCX 3 1 INH	INC 6 2 IX1	INC 5 1 IX		RSP 2 1 INH		JMP 2 2 DIR	JMP 3 3 EXT	JMP 4 3 IX2	JMP 3 2 IX1	JMP 2 1 IX	C 1100
D 1101	BRCLR6 3 BTB	BCLR6 2 BSC	BMS 2 REL	TST 4 2 DIR	TSTA 3 1 INH	TSTX 3 1 INH	TST 5 2 IX1	TST 4 1 IX		NOP 2 1 INH	BSR 6 2 REL	JSR 5 2 DIR	JSR 6 3 EXT	JSR 7 3 IX2	JSR 6 2 IX1	JSR 5 1 IX	D 1101
E 1110	BRSET7 3 BTB	BSET7 2 BSC	BIL 2 REL						STOP 2 1 INH		LDX 2 2 IMM	LDX 3 2 DIR	LDX 4 3 EXT	LDX 5 3 IX2	LDX 4 2 IX1	LDX 3 1 IX	E 1110
F 1111	BRCLR7 3 BTB	BCLR7 2 BSC	BIH 2 REL	CLR 5 2 DIR	CLRA 3 1 INH	CLRX 3 1 INH	CLR 6 2 IX1	CLR 5 1 IX	WAIT 2 1 INH	TXA 2 1 INH		STX 4 2 DIR	STX 5 3 EXT	STX 6 3 IX2	STX 5 2 IX1	STX 4 1 IX	F 1111

Abbreviations for Address Modes

INH	Inherent
A	Accumulator
X	Index Register
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This address mode can be used in a manner similar to indexed, 8-bit offset except that this three-byte instruction allows tables to be anywhere in memory. As with direct and extended addressing, the Motorola assembler determines the shortest form of indexed addressing.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode, and the byte following the opcode specifies the direct addressing of the byte in which the specified bit is to be set or cleared. Any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single two-byte instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit that is to be tested and its condition (set or clear), is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

3

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS (Voltages referenced to V_{SS})

Rating	Symbol	Value	Unit
Supply Voltage	V_{DD}	−0.3 to +7.0	V
Input Voltage	V_{in}	V_{SS} −0.3 to V_{DD} +0.3	V
Self-Check Mode ($\overline{IRQ}$ Pin Only)	V_{in}	V_{SS} −0.3 to $2 \times V_{DD} + 0.3$	V
Current Drain Per Pin Excluding V_{DD} and V_{SS}	I	25	mA
Operating Temperature Range MC68HC05C4P, FN (Standard) MC68HC05C4CP, CFN (Extended) MC68HC05C4MP, MFN (Automotive)	T_A	T_L to T_H 0 to +70 −40 to +85 −40 to +125	°C
Storage Temperature Range	T_{stg}	−65 to +150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation, it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{DD}$. Reliability of operation is enhanced if unused inputs are connected to an appropriate logic voltage level (e.g., either V_{SS} or V_{DD}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic Plastic Leaded Chip Carrier (PLCC)	θ_{JA}	 60 70	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{I/O}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
$P_{I/O}$ = Power Dissipation on Input and Output Pins — User Determined

For most applications $P_{I/O} < P_{INT}$ and can be neglected.

The following is an approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected):

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

V_{DD} = 4.5 V

Pins	R1	R2	C
PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4	3.26 kΩ	2.38 kΩ	50 pF
PD0,PD5, PD7	1.9 kΩ	2.26 kΩ	200 pF

V_{DD} = 3.0 V

Pins	R1	R2	C
PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4	10.91 kΩ	6.32 kΩ	50 pF
PD0, PD5, PD7	6 kΩ	6 kΩ	200 pF

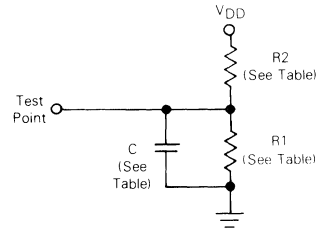


Figure 18. Equivalent Test Load

DC ELECTRICAL CHARACTERISTICS

(V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage, $I_{Load} \approx 10.0\ \mu A$	V_{OL} V_{OH}	— $V_{DD} - 0.1$	— —	0.1 —	V
Output High Voltage (I_{Load} = 0.8 mA) PA0-PA7, PB0-PB7, PC0-PC7, TCMP (see Figure 19) (I_{Load} = 1.6 mA) PD1-PD4 (see Figure 20)	V_{OH}	 $V_{DD} - 0.8$ $V_{DD} - 0.8$	 — —	 — —	V
Output Low Voltage (see Figure 21) (I_{Load} = 1.6 mA) PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4, TCMP	V_{OL}	—	—	0.4	V
Input High Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IH}	$0.7 \times V_{DD}$	—	V_{DD}	V
Input Low Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IL}	V_{SS}	—	$0.2 \times V_{DD}$	V
Data Retention Mode (0° to 70°C)	V_{RM}	2.0	—	—	V
Supply Current (see Notes) Run (see Figures 22 and 23) Wait (see Figures 22 and 23) Stop (see Figure 23) 25°C 0° to 70°C (Standard) −40° to +85°C −40° to +125°C	I_{DD}	 — — — — — —	 3.5 1.6 2.0 — — —	 7.0 4.0 50 140 180 250	 mA mA µA µA µA µA
I/O Ports Hi-Z Leakage Current PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4	I_{IL}	—	—	±10	µA
Input Current $\overline{RESET}$, $\overline{IRQ}$, TCAP, OSC1, PD0, PD5, PD7	I_{in}	—	—	±1	µA
Capacitance Ports (as Input or Output) $\overline{RESET}$, $\overline{IRQ}$, TCAP, PD0-PD5, PD7	 C_{out} C_{in}	 — —	 — —	 12 8	pF

NOTES:
1. All values shown reflect average measurements.
2. Typical values at midpoint of voltage range, 25°C only.
3. Wait I_{DD}: Only timer system active (SPE = TE = RE = 0). If SPI, SCI active (SPE = TE = RE = 1) add 10% current draw.
4. Run (Operating) I_{DD}, Wait I_{DD}: Measured using external square wave clock source (f_{osc} = 4.2 MHz), all inputs 0.2 V from rail; no dc loads, less than 50 pF on all outputs, C_L = 20 pF on OSC2.
5. Wait, Stop I_{DD}: All ports configured as inputs, V_{IL} = 0.2 V, $V_{IH} = V_{DD} - 0.2$ V.
6. Stop I_{DD} measured with OSC1 = V_{SS}.
7. Standard temperature range is 0° to 70°C. Extended temperature (−40° to +85°C, −40° to +125°C) versions and a 25°C only version are available.
8. Wait I_{DD} is affected linearly by the OSC2 capacitance.

DC ELECTRICAL CHARACTERISTICS

(V_{DD} = 3.3 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage, $I_{Load} \leq 10.0$ µA	V_{OL} V_{OH}	— $V_{DD}-0.1$	— —	0.1 —	V
Output High Voltage (I_{Load} = 0.2 mA) PA0-PA7, PB0-PB7, PC0-PC7, TCMP (see Figure 19) (I_{Load} = 1.6 mA) PD1-PD4 (see Figure 20)	V_{OH}	 $V_{DD}-0.3$ $V_{DD}-0.3$	 — —	 — —	V
Output Low Voltage (see Figure 21) (I_{Load} = 0.4 mA) PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4, TCMP	V_{OL}	—	—	0.3	V
Input High Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IH}	$0.7 \times V_{DD}$	—	V_{DD}	V
Input Low Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IL}	V_{SS}	—	$0.2 \times V_{DD}$	V
Data Retention Mode (0° to 70°C)	V_{RM}	2.0	—	—	V
Supply Current (see Notes) Run (see Figures 22 and 24) Wait (see Figures 22 and 24) Stop (see Figure 24) 25°C 0° to 70°C (Standard) −40° to +85°C −40° to +125°C	I_{DD}	 — — — — — —	 1.0 0.5 1.0 — — —	 2.5 1.4 30 80 120 175	 mA mA µA µA µA µA
I/O Ports Hi-Z Leakage Current PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4	I_{IL}	—	—	±10	µA
Input Current RESET, IRQ, TCAP, OSC1, PD0, PD5, PD7	I_{in}	—	—	±1	µA
Capacitance Ports (as Input or Output) RESET, IRQ, TCAP, PD0-PD5, PD7	 C_{out} C_{in}	 — —	 — —	 12 8	pF

NOTES:

1. All values shown reflect average measurements.
2. Typical values at midpoint of voltage range, 25°C only.
3. Wait I_{DD}: Only timer system active (SPE = TE = RE = 0). If SPI, SCI active (SPE = TE = RE = 1) add 10% current draw.
4. Run (Operating) I_{DD}, Wait I_{DD}: Measured using external square wave clock source (f_{osc} = 4.2 MHz), all inputs 0.2 V from rail; no dc loads, less than 50 pF on all outputs, C_L = 20 pF on OSC2.
5. Wait, Stop I_{DD}: All ports configured as inputs, V_{IL} = 0.2 V, $V_{IH} = V_{DD} - 0.2$ V.
6. Stop I_{DD} measured with OSC1 = V_{SS}.
7. Standard temperature range is 0° to 70°C. Extended temperature (−40° to +85°C, −40° to +125°C) versions and a 25°C only version are available.
8. Wait I_{DD} is affected linearly by the OSC2 capacitance.

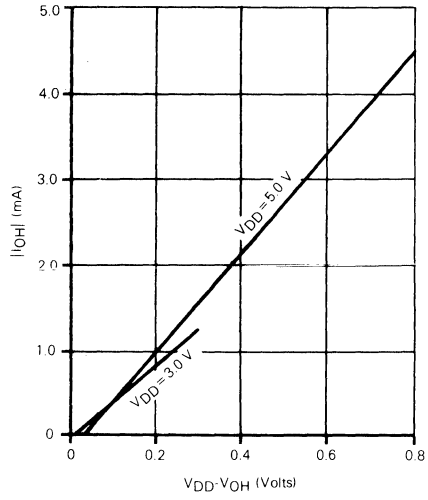


Figure 19. Typical V_{OH} vs I_{OH} for Ports A, B, C, and TCMP

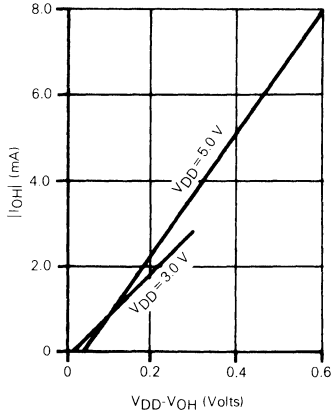


Figure 20. Typical V_{OH} vs I_{OH} for PD1-PD4

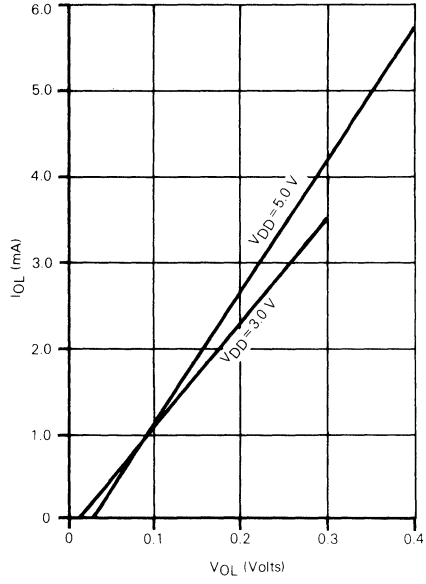


Figure 21. Typical V_{OL} vs I_{OL} for All Ports

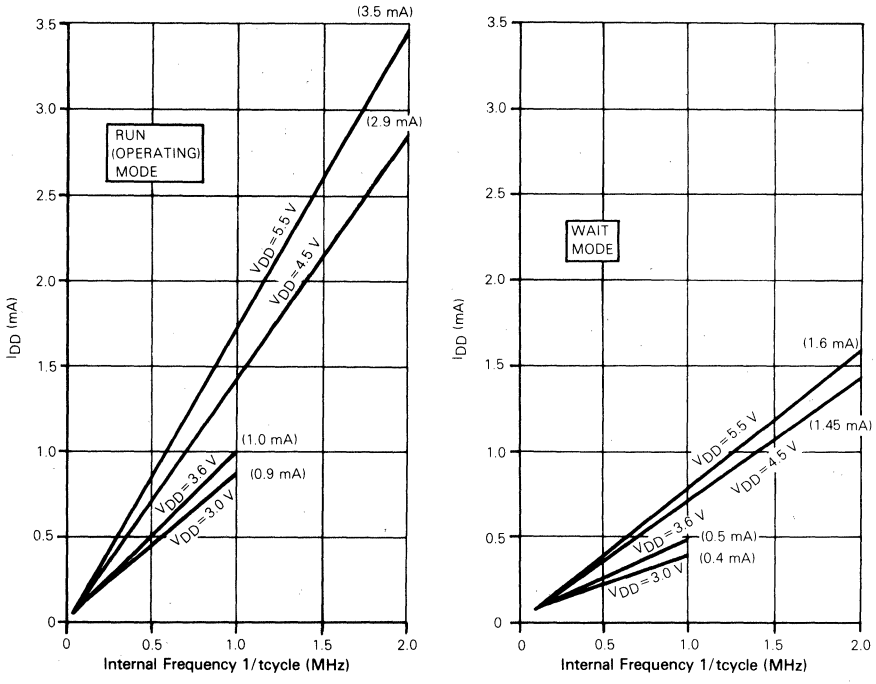


Figure 22. Typical Current vs Internal Frequency for Run and Wait Modes

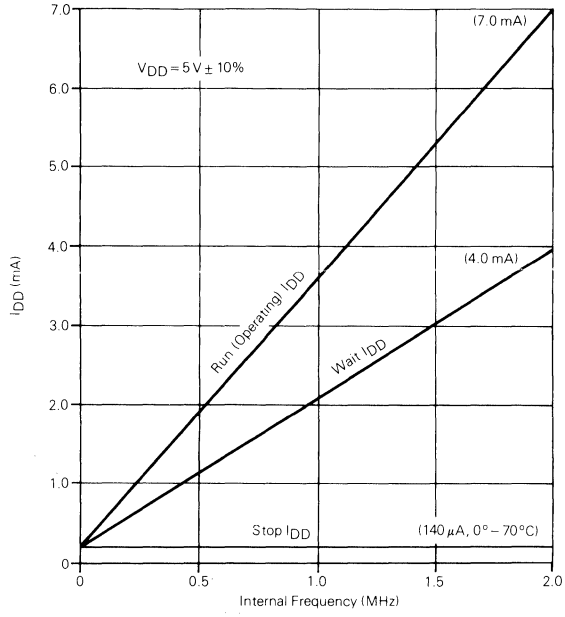


Figure 23. Maximum I_{DD} vs Frequency for $V_{DD} = 5.0$ Vdc

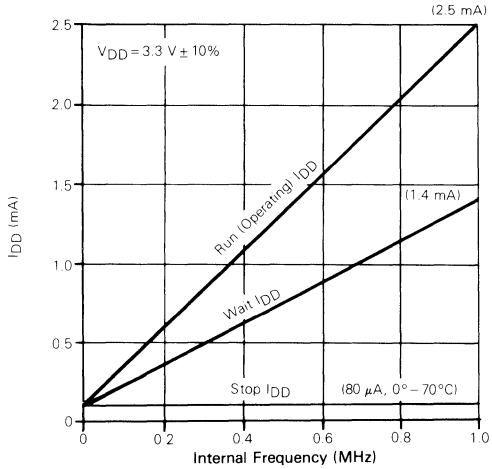


Figure 24. Maximum I_{DD} vs Frequency for $V_{DD} = 3.3$ Vdc

CONTROL TIMING

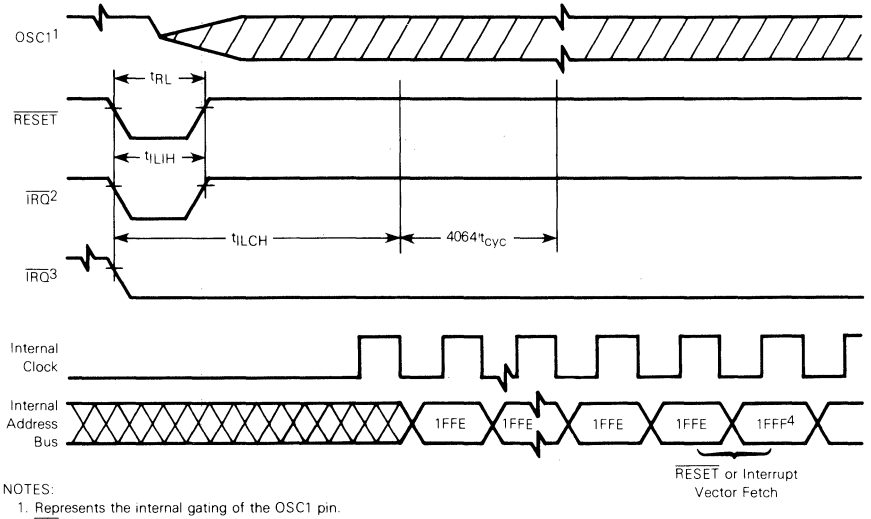
(V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation Crystal Option External Clock Option	f_{osc}	 — dc	 4.2 4.2	MHz
Internal Operating Frequency Crystal ($f_{osc} \div 2$) External Clock ($f_{osc} \div 2$)	f_{op}	 — dc	 2.1 2.1	MHz
Cycle Time (see Figure 28)	t_{cyc}	480	—	ns
Crystal Oscillator Startup Time (see Figure 28)	t_{OXOV}	—	100	ms
Stop Recovery Startup Time (Crystal Oscillator) (see Figure 25)	t_{ILCH}	—	100	ms
$\overline{\text{RESET}}$ Pulse Width (see Figure 28)	t_{RL}	1.5	—	t_{cyc}
Timer Resolution** Input Capture Pulse Width (see Figure 26) Input Capture Pulse Period (see Figure 26)	 t_{RESL} t_{TH}, t_{TL} t_{TLTL}	 4.0 125 ***	 — — —	 t_{cyc} ns t_{cyc}
Interrupt Pulse Width Low (Edge-Triggered) (see Figure 8)	t_{ILIH}	125	—	ns
Interrupt Pulse Period (see Figure 8)	t_{ILIL}	*	—	t_{cyc}
OSC1 Pulse Width	t_{OH}, t_{OL}	90	—	ns

*The minimum period t_{ILIL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus 21 t_{cyc}.

**Since a 2-bit prescaler in the timer must count four internal cycles (t_{cyc}), this is the limiting minimum factor in determining the timer resolution.

***The minimum period t_{TLTL} should not be less than the number of cycle times it takes to execute the capture interrupt service routine plus 24 t_{cyc}.



NOTES:
1. Represents the internal gating of the OSC1 pin.
2. $\overline{\text{IRQ}}$ pin edge-sensitive mask option.
3. $\overline{\text{IRQ}}$ pin level and edge-sensitive mask option.
4. $\overline{\text{RESET}}$ vector address shown for timing example.

Figure 25. Stop Recovery Timing Diagram

CONTROL TIMING

(V_{DD} = 3.3 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation Crystal Option External Clock Option	f_{osc}	 — dc	 2.0 2.0	MHz
Internal Operating Frequency Crystal ($f_{osc} \div 2$) External Clock ($f_{osc} \div 2$)	f_{op}	 — dc	 1.0 1.0	MHz
Cycle Time (see Figure 28)	t_{cyc}	1000	—	ns
Crystal Oscillator Startup Time (see Figure 28)	t_{OXOV}	—	100	ms
Stop Recovery Startup Time (Crystal Oscillator) (see Figure 25)	t_{ILCH}	—	100	ms
$\overline{\text{RESET}}$ Pulse Width — Excluding Power-Up (see Figure 28)	t_{RL}	1.5	—	t_{cyc}
Timer Resolution** Input Capture Pulse Width (see Figure 26) Input Capture Pulse Period (see Figure 26)	 t_{RESL} t_{TH}, t_{TL} t_{TLTL}	 4.0 250 ***	 — — —	 t_{cyc} ns t_{cyc}
Interrupt Pulse Width Low (Edge-Triggered) (see Figure 8)	t_{ILIH}	250	—	ns
Interrupt Pulse Period (see Figure 8)	t_{ILIL}	*	—	t_{cyc}
OSC1 Pulse Width	t_{OH}, t_{OL}	200	—	ns

*The minimum period t_{ILIL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus 21 t_{cyc}.

**Since a 2-bit prescaler in the timer must count four internal cycles (t_{cyc}), this is the limiting minimum factor in determining the timer resolution.

***The minimum period t_{TLTL} should not be less than the number of cycle times it takes to execute the capture interrupt service routine plus 24 t_{cyc}.

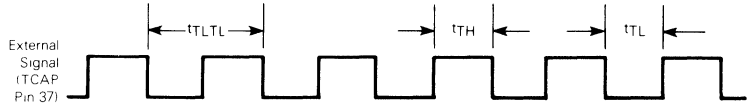


Figure 26. Timer Relationships

SERIAL PERIPHERAL INTERFACE (SPI) TIMING

(V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H) (see Figure 27)

Num.	Characteristic	Symbol	Min	Max	Unit
	Operating Frequency Master Slave	 $f_{op(m)}$ $f_{op(s)}$	 dc dc	 0.5 2.1	 f_{op} MHz
1	Cycle Time Master Slave	 $t_{cyc(m)}$ $t_{cyc(s)}$	 2.0 480	 — —	 t_{cyc} ns
2	Enable Lead Time Master Slave	 $t_{lead(m)}$ $t_{lead(s)}$	 * 240	 — —	 ns ns
3	Enable Lag Time Master Slave	 $t_{lag(m)}$ $t_{lag(s)}$	 * 240	 — —	 ns ns
4	Clock (SCK) High Time Master Slave	 $t_{w(SCKH)m}$ $t_{w(SCKH)s}$	 340 190	 — —	 ns ns
5	Clock (SCK) Low Time Master Slave	 $t_{w(SCKL)m}$ $t_{w(SCKL)s}$	 340 190	 — —	 ns ns
6	Data Setup Time (Inputs) Master Slave	 $t_{su(m)}$ $t_{su(s)}$	 100 100	 — —	 ns ns
7	Data Hold Time (Inputs) Master Slave	 $t_{h(m)}$ $t_{h(s)}$	 100 100	 — —	 ns ns
8	Access Time (Time to Data Active from High-Impedance State) Slave	 t_a	 0	 120	 ns
9	Disable Time (Hold Time to High-Impedance State) Slave	 t_{dis}	 —	 240	 ns
10	Data Valid Master (Before Capture Edge) Slave (After Enable Edge)**	 $t_{v(m)}$ $t_{v(s)}$	 0.25 —	 — 240	 $t_{cyc(m)}$ ns
11	Data Hold Time (Outputs) Master (After Capture Edge) Slave (After Enable Edge)	 $t_{ho(m)}$ $t_{ho(s)}$	 0.25 0	 — —	 $t_{cyc(m)}$ ns
12	Rise Time (20% V_{DD} to 70% V_{DD}, C_L = 200 pF) SPI Outputs (SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and SS)	 t_{rm} t_{rs}	 — —	 100 2.0	 ns µs
13	Fall Time (70% V_{DD} to 20% V_{DD}, C_L = 200 pF) SPI Outputs (SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and SS)	 t_{fm} t_{fs}	 — —	 100 2.0	 ns µs

*Signal production depends on software.
**Assumes 200 pF load on all SPI pins.

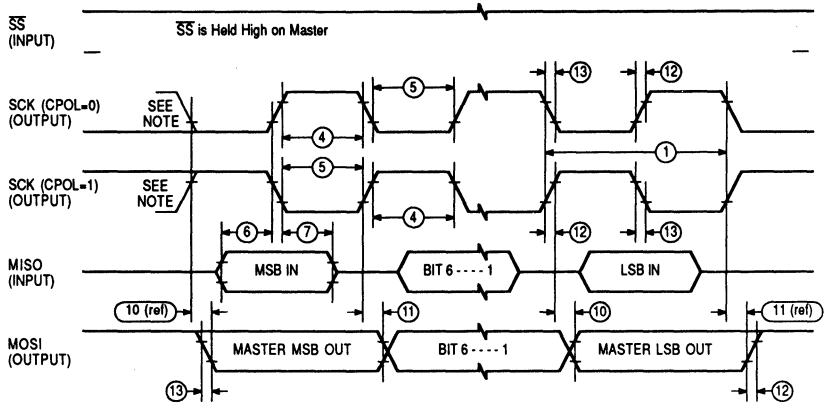
SERIAL PERIPHERAL INTERFACE (SPI) TIMING

(V_{DD} = 3.3 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H) (see Figure 27)

Num.	Characteristic	Symbol	Min	Max	Unit
	Operating Frequency Master Slave	 $f_{op(m)}$ $f_{op(s)}$	 dc dc	 0.5 1.0	 f_{op} MHz
1	Cycle Time Master Slave	 $t_{cyc(m)}$ $t_{cyc(s)}$	 2.0 1.0	 — —	 t_{cyc} µs
2	Enable Lead Time Master Siave	 $t_{lead(m)}$ $t_{lead(s)}$	 * 500	 — —	 ns ns
3	Enable Lag Time Master Slave	 $t_{lag(m)}$ $t_{lag(s)}$	 * 500	 — —	 ns ns
4	Clock (SCK) High Time Master Slave	 $t_{w(SCKH)m}$ $t_{w(SCKH)s}$	 720 400	 — —	 µs ns
5	Clock (SCK) Low Time Master Slave	 $t_{w(SCKL)m}$ $t_{w(SCKL)s}$	 720 400	 — —	 µs ns
6	Data Setup Time (Inputs) Master Slave	 $t_{su(m)}$ $t_{su(s)}$	 200 200	 — —	 ns ns
7	Data Hold Time (Inputs) Master Slave	 $t_{h(m)}$ $t_{h(s)}$	 200 200	 — —	 ns ns
8	Access Time (Time to Data Active from High-Impedance State) Slave	 t_a	 0	 250	 ns
9	Disable Time (Hold Time to High-Impedance State) Slave	 t_{dis}	 —	 500	 ns
10	Data Valid Master (Before Capture Edge) Slave (After Enable Edge)**	 $t_{v(m)}$ $t_{v(s)}$	 0.25 —	 — 500	 $t_{cyc(m)}$ ns
11	Data Hold Time (Outputs) Master (After Capture Edge) Slave (After Enable Edge)	 $t_{ho(m)}$ $t_{ho(s)}$	 0.25 0	 — —	 $t_{cyc(m)}$ ns
12	Rise Time (20% V_{DD} to 70% V_{DD}, C_L = 200 pF) SPI Outputs (SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and SS)	 t_{rm} t_{rs}	 — —	 200 2.0	 ns µs
13	Fall Time (70% V_{DD} to 20% V_{DD}, C_L = 200 pF) SPI Outputs (SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and SS)	 t_{fm} t_{fs}	 — —	 200 2.0	 ns µs

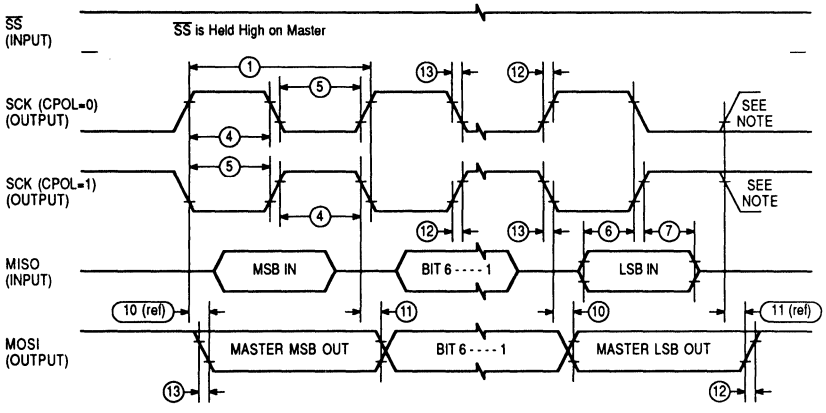
*Signal production depends on software.
**Assumes 200 pF load on all SPI pins.

3



NOTE: This first clock edge is generated internally but is not seen at the SCK pin.

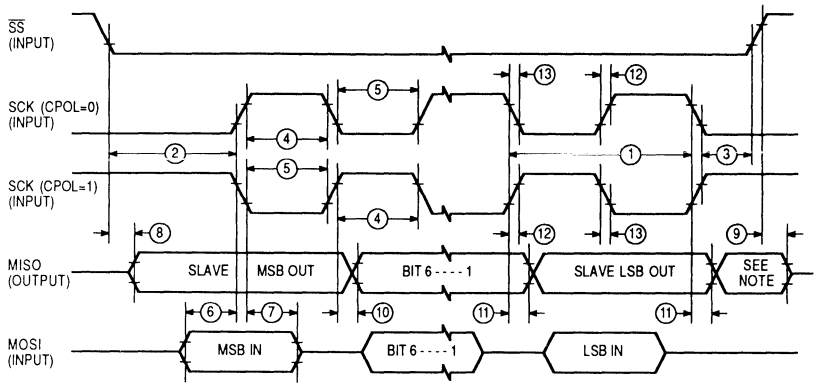
a) SPI MASTER TIMING (CPHA = 0)



NOTE: This last clock edge is generated internally but is not seen at the SCK pin.

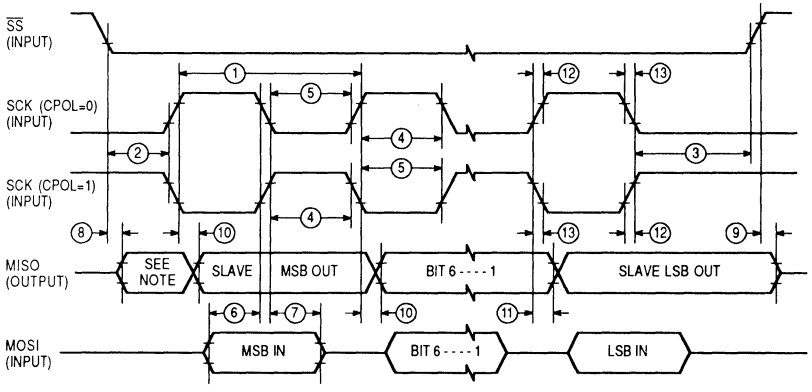
b) SPI MASTER TIMING (CPHA = 1)

Figure 27. SPI Timing Diagrams (Sheet 1 of 2)



NOTE: Not defined but normally MSB of character just received.

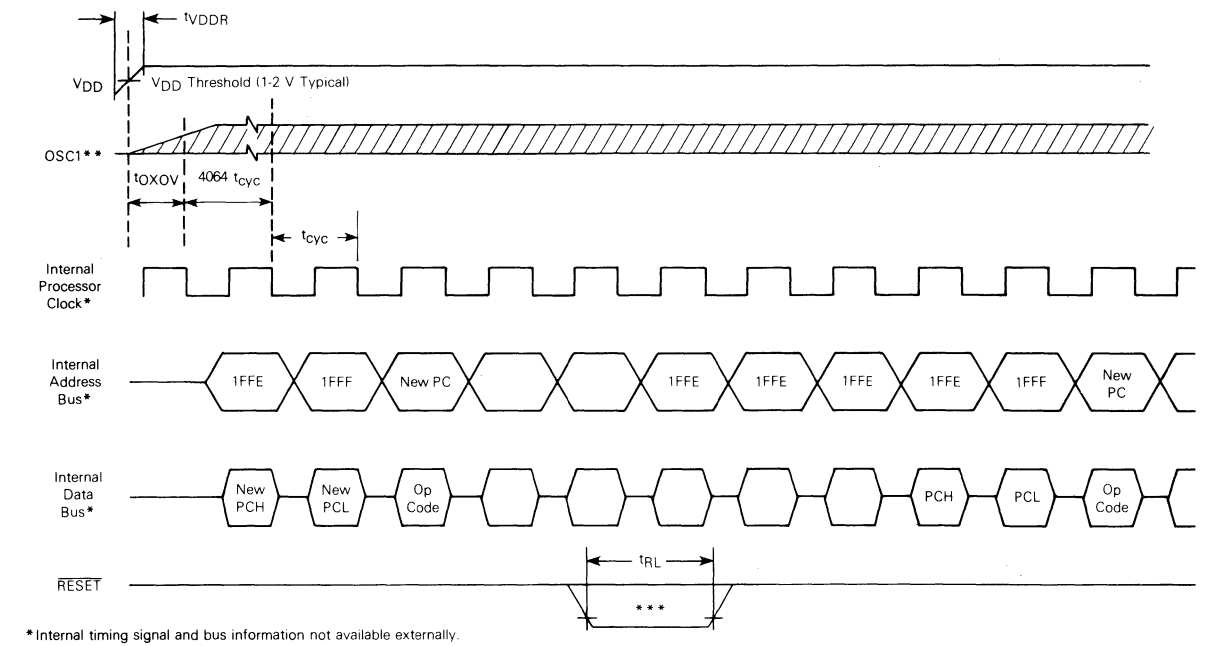
c) SPI SLAVE TIMING (CPHA = 0)



NOTE: Not defined but normally LSB of character previously transmitted.

d) SPI SLAVE TIMING (CPHA = 1)

Figure 27. SPI Timing Diagrams (Sheet 2 of 2)



*Internal timing signal and bus information not available externally.
**OSC1 line is not meant to represent frequency. It is only used to represent time.
***The next rising edge of the internal processor clock following the rising edge of $\overline{\text{RESET}}$ initiates the reset sequence.

Figure 28. Power-On Reset and $\overline{\text{RESET}}$

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola in the following media:

MDOS™, disk file
MS™-DOS/PC-DOS disk file (360K)
EPROM(s) 2764, MCM68764, MCM68766, or EEPROM MC68HC805C4

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, a sales person, or a Motorola representative.

FLEXIBLE DISKS

A flexible disk (MS-DOS/PC-DOS disk file), programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. The diskette should be clearly labeled with the customer's name, data, project or product name, and the name of the file containing the pattern.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to expedite the process in case of any difficulty with the pattern file.

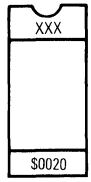
MS-DOS/PC-DOS Disk File

MS-DOS is Microsoft's Disk Operating System. PC-DOS is the IBM® Personal Computer (PC) Disk Operating System. Disk media submitted must be a standard density (360K) double-sided 5 1/4 inch compatible floppy diskette. The diskette must contain object file code in Motorola's S-record format. The S-record format is a character-based object file format generated by M6805 cross assemblers and linkers on IBM PC style machines.

EPROMs

A 2764, 68764, or 68766 type EPROM, programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. Since all program and data space information will fit on one 2764, 68764, or 68766 EPROM device, the EPROM must be programmed as described in the following paragraphs.

For an MC68HC805C4 MCU start the page zero, user ROM at EEPROM address $0020 through $004F. Start the user ROM at EEPROM address $0100 through $10FF with vectors from $1FF4 to $1FFF. All unused bytes, including the user's space, must be set to zero. For shipment to Motorola, EPROMs should be placed in a conductive IC carrier and packed securely. Styrofoam is not acceptable for shipment.



xxx = Customer ID

Verification Media

All original pattern media (EPROMs or floppy disks) are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked, and the verification form should be completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for the creation of the customer mask. To aid in the verification process, Motorola will program *customer supplied* blank EPROM(s) or DOS disks from the data file used to create the custom mask.

ROM VERIFICATION UNITS (RVUs)

Ten MCUs containing the customer's ROM pattern will be sent for program verification. These units will have been made using the custom mask, but are for the purpose of ROM verification only. For expediency, the MCUs are unmarked, packaged in ceramic, and tested with five volts at room temperature. These RVUs are free with the minimum order quantity, but are not production parts. RVUs are not backed or guaranteed by Motorola Quality Assurance.

ORDERING INFORMATION

The following table provides ordering information pertaining to the package type, temperature, and MC order numbers for the MC68HC05C4 device.

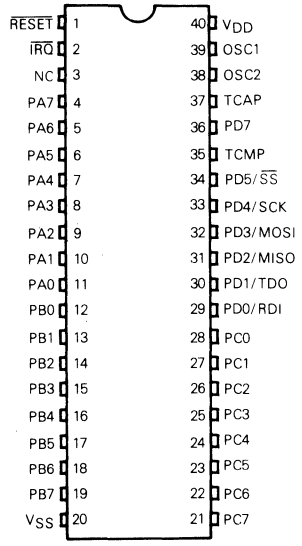
Package Type	Temperature	MC Order Number
Plastic (P Suffix)	0°C to +70°C	MC68HC05C4P
	−40°C to +85°C	MC68HC05C4CP
	−40° to +105°C	MC68HC05C4VP
	−40°C to +125°C	MC68HC05C4MP
PLCC (FN Suffix)	0°C to +70°C	MC68HC05C4FN
	−40°C to +85°C	MC68HC05C4CFN
	−40°C to +105°C	MC68HC05C4VFN
	−40°C to +125°C	MC68HC05C4MFN

3

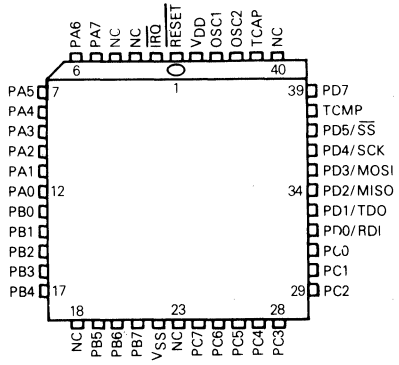
MDOS is a trademark of Motorola Inc.
MS is a trademark of Microsoft, Inc.
IBM is a registered trademark of International Business Machines Corporation.

PIN ASSIGNMENTS

40-PIN DUAL-IN-LINE PACKAGE



44-LEAD PLCC PACKAGE



NOTE: Bulk substrate tied to V_{SS}.

MC68HC05C8

Technical Summary
8-Bit Microcontroller Unit

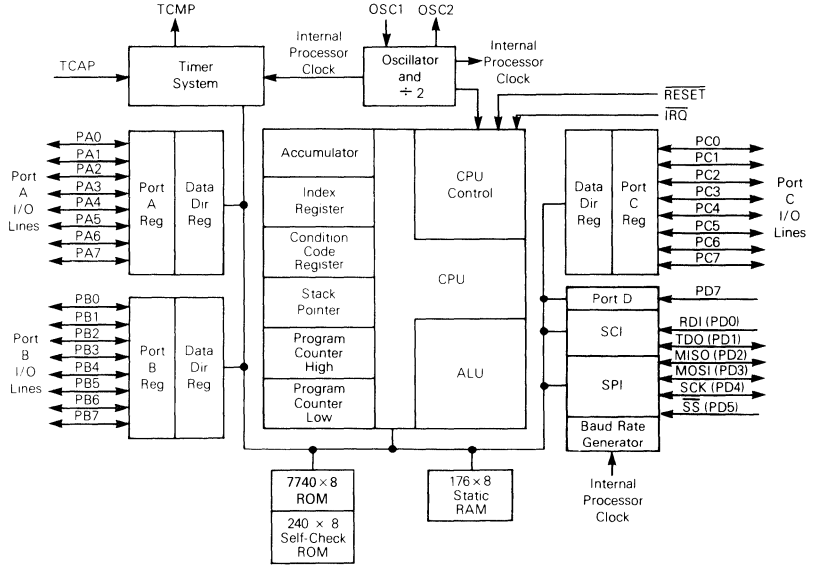
The MC68HC05C8 (HCMOS) microcontroller unit (MCU) is a member of the M68HC05 Family of microcontrollers. This high-performance, low-power MCU has parallel I/O capability with pins programmable as input or output. This publication contains condensed information on the MCU; for more detailed information, contact your local Motorola sales office.

The following block diagram depicts the hardware features; additional features available on the MCU are as follows:

- On-Chip Oscillator with RC or Crystal/Ceramic Resonator Mask Options
- Memory-Mapped I/O
- 176 Bytes of On-Chip RAM
- 7740 Bytes of User ROM
- 24 Bidirectional I/O Lines and 7 Input-Only Lines
- Serial Communications Interface (SCI) System
- Serial Peripheral Interface (SPI) System
- Self-Check Mode
- Power-Saving STOP, WAIT, and Data Retention Modes
- Single 3.0- to 5.5-Volt Supply (2-Volt Data Retention Mode)
- Fully Static Operation
- 8×8 Unsigned Multiply Instruction

3

BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

SIGNAL DESCRIPTION

The signal descriptions of the MCU are discussed in the following paragraphs.

V_{DD} AND V_{SS}

Power is supplied to the microcontroller using these two pins. V_{DD} is the positive supply, and V_{SS} is ground.

$\overline{IRQ}$

This pin is a programmable option that provides two different choices of interrupt triggering sensitivity. Refer to **INTERRUPTS** for more detail.

OSC1, OSC2

These pins provide control input for an on-chip clock oscillator circuit. A crystal, a ceramic resonator, a resistor/capacitor combination, or an external signal connects to these pins providing a system clock. A mask option selects either a crystal/ceramic resonator or a resistor/capacitor as the frequency determining element. The oscillator frequency is two times the internal bus rate.

RC Oscillator

With this option, a resistor is connected to the oscillator pins as shown in Figure 1(d). The relationship between R and f_{osc} is shown in Figure 2.

Crystal

The circuit shown in Figure 1(b) is recommended when using a crystal. Using an external CMOS oscillator is recommended when crystals outside the specified ranges are to be used. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and start-up stabilization time. Refer to **ELECTRICAL SPECIFICATIONS** for V_{DD} specifications.

Ceramic Resonator

A ceramic resonator may be used in place of the crystal in cost-sensitive applications. The circuit in Figure 1(b) is

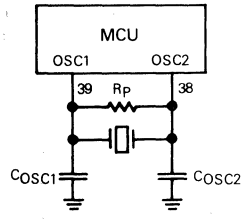
Crystal

	2 MHz	4 MHz	Units
R_{SMAX}	400	75	Ω
C_0	5	7	pF
C_1	0.008	0.012	μF
C_{OSC1}	15-40	15-30	pF
C_{OSC2}	15-30	15-25	pF
R_P	10	10	MΩ
Q	30	40	K

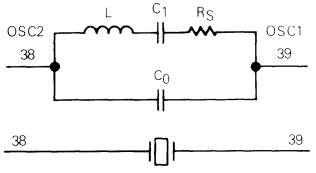
Ceramic Resonator

	2-4 MHz	Units
R_S (typical)	10	Ω
C_O	40	pF
C_1	4.3	pF
C_{OSC1}	30	pF
C_{OSC2}	30	pF
R_P	1-10	MΩ
Q	1250	—

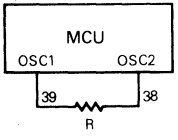
(a) Crystal/Ceramic Resonator Parameters



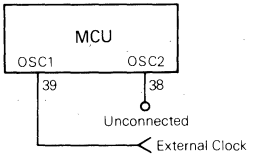
(b) Crystal/Ceramic Resonator Oscillator Connections



(c) Equivalent Crystal Circuit



(d) RC Oscillator Connections



(e) External Clock Source Connections (For Crystal Mask Option Only)

Figure 1. Oscillator Connections

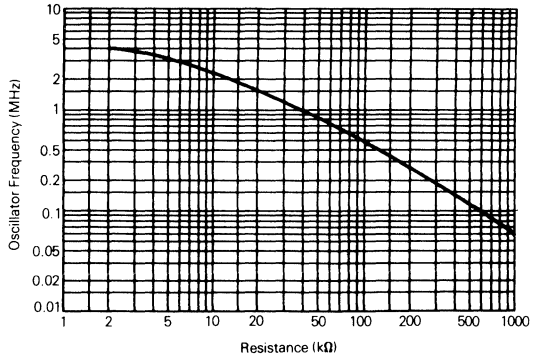


Figure 2. Typical Frequency vs Resistance for RC Oscillator Option Only

recommended when using a ceramic resonator. Figure 1(a) lists the recommended capacitance and resistance values. The manufacturer of the resonator considered should be consulted for specific information on resonator operation.

External Clock

An external clock should be applied to the OSC1 input with the OSC2 input not connected, as shown in Figure 1(e). This option may only be used with the crystal oscillator mask option.

INPUT CAPTURE (TCAP)

This pin controls the input capture feature for the on-chip programmable timer.

OUTPUT COMPARE (TCMP)

This pin provides an output for the output compare feature of the on-chip timer.

$\overline{\text{RESET}}$

This pin is used to reset the MCU and provide an orderly start-up procedure by pulling $\overline{\text{RESET}}$ low.

INPUT/OUTPUT PORTS (PA0-PA7, PB0-PB7, PC0-PC7)

These 24 lines are arranged into three 8-bit ports (A, B, and C). These ports are programmable as either inputs or outputs under software control of the data direction registers. Refer to **PROGRAMMING** for additional information.

FIXED INPUT PORT (PD0-PD5, PD7)

These seven lines comprise port D, a fixed input port. All special functions that are enabled (SPI, SCI) affect this port. Refer to **PROGRAMMING** for additional information.

PROGRAMMING

Input/output port programming, fixed input port programming, and serial port programming are discussed in the following paragraphs.

INPUT/OUTPUT PORT PROGRAMMING

Any port pin is programmable as either an input or an output under software control of the corresponding data direction register (DDR). Each port bit can be selected as output or input by writing the corresponding bit in the port DDR to a logic one for output and logic zero for input. On reset, all DDRs are initialized to logic zero to put the ports in the input mode. The port output registers are not initialized on reset but may be written to before setting the DDR bits to avoid undefined levels.

When programmed as outputs, the latched output data is readable as input data regardless of the logic levels at the output pin due to output loading. The latched output data bit may always be written. Therefore, any write to a port writes all of its data bits, even though the port DDR is set to input. This port write may be used to initialize the data registers and avoid undefined outputs. Refer to Figure 3 for typical port circuitry and to Table 1 for a list of the I/O pin functions.

Table 1. I/O Pin Functions

R/$\overline{\text{W}}$*	DDR	I/O Pin Functions
0	0	The I/O pin is in input mode. Data is written into the output data latch.
0	1	Data is written into the output data latch and output to the I/O pin.
1	0	The state of the I/O pin is read.
1	1	The I/O pin is in an output mode. The output data latch is read.

*R/$\overline{\text{W}}$ is an internal signal.

FIXED INPUT PORT PROGRAMMING

Port D is a fixed input port (PD0-PD5, PD7) that monitors the external pins whenever the SCI or SPI is disabled. After reset, all seven bits become valid inputs because all special function drivers are disabled. For example, with the SCI enabled, PD0 and PD1 inputs will read zero. With the SPI disabled, PD2 through PD5 will read the state of the pin at the time of the read operation.

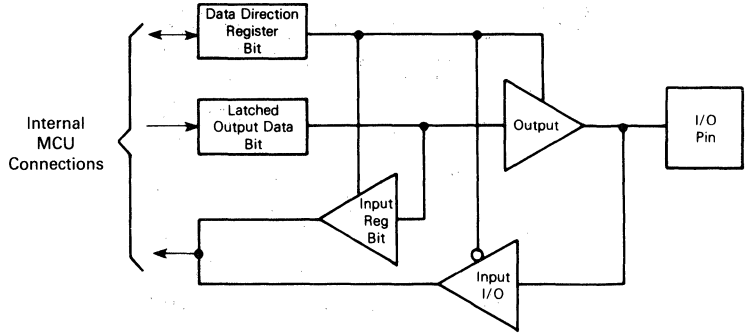


Figure 3. Typical Port I/O Circuit

NOTE

Any unused inputs and I/O ports should be tied to an appropriate logic level (e.g., either V_{DD} or V_{SS}).

SERIAL PORT (SCI AND SPI) PROGRAMMING

The SCI and SPI use the port D pins for their functions. The SCI requires two pins (PD0-PD1) for its receive data input (RDI) and transmit data output (TD0), respectively. The SPI function requires four of the pins (PD2-PD5) for its serial data input/output (MISO), serial data output/input (MOSI), serial clock (SCK), and slave select ($\overline{SS}$), respectively.

MEMORY

The MCU is capable of addressing 8192 bytes of memory and I/O registers, as shown in Figure 4. The locations consist of user ROM, user RAM, self-check ROM, control registers, and I/O. The user-defined reset and interrupt vectors are located from $1FF4 to $1FFF.

The shared stack area is used during processing of an interrupt or subroutine call to save the CPU state. The stack pointer decrements during pushes and increments during pulls. Refer to **INTERRUPTS** for additional information.

NOTE

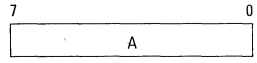
Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

REGISTERS

The MCU contains the registers described in the following paragraphs.

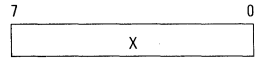
ACCUMULATOR (A)

The accumulator is a general-purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



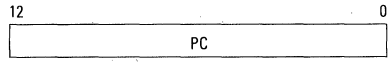
INDEX REGISTER (X)

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an 8- or 16-bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



PROGRAM COUNTER (PC)

The program counter is a 13-bit register that contains the address of the next byte to be fetched.



STACK POINTER (SP)

The stack pointer is a 13-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set to location $00FF. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

When accessing memory, the seven most significant bits are permanently set to 0000011. These seven bits are appended to the six least significant register bits to produce an address within the range of $00FF to $00C0. Subroutines and interrupts may use up to 64 (decimal) locations. If 64 locations are exceeded, the stack pointer

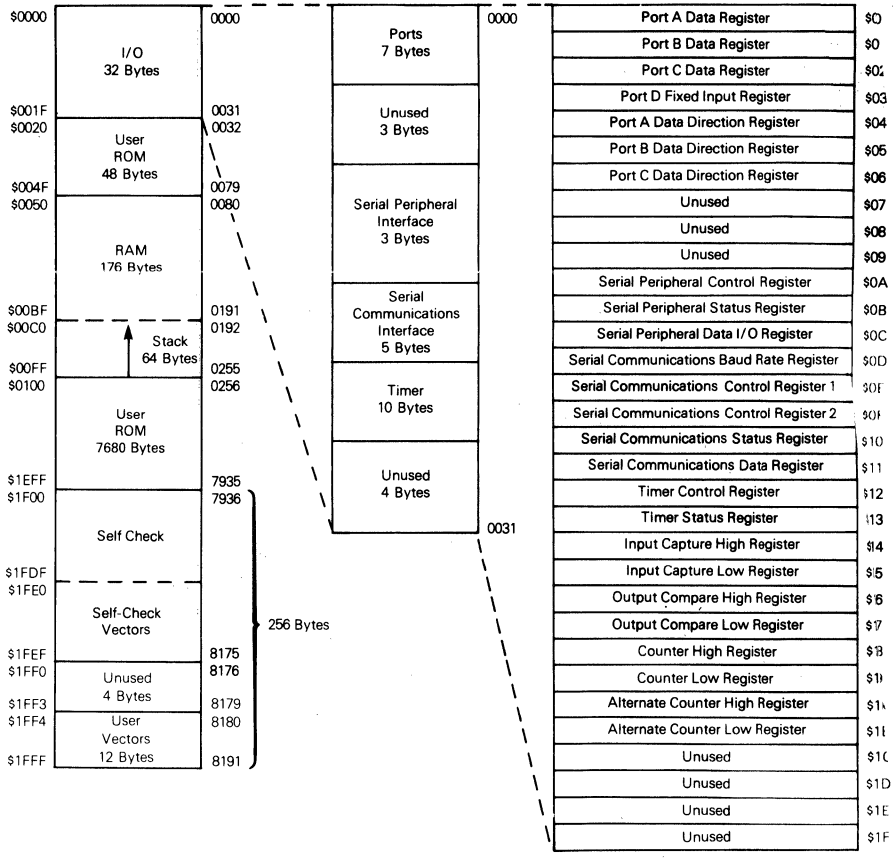
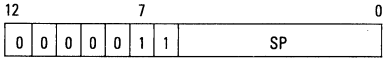


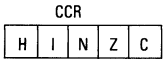
Figure 4. Memory Map

wraps around and loses the previously stored information. A subroutine call occupies two locations on the stack; an interrupt uses five locations.



CONDITION CODE REGISTER (CCR)

The CCR is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program, and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.



Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

When this bit is set, the timer and external interrupt is masked (disabled). If an interrupt occurs while this bit is set, the interrupt is latched and processed as soon as the interrupt bit is cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative (bit 7 in the result is a logic one).

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

Carry/Borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions and during shifts and rotates

SELF-CHECK

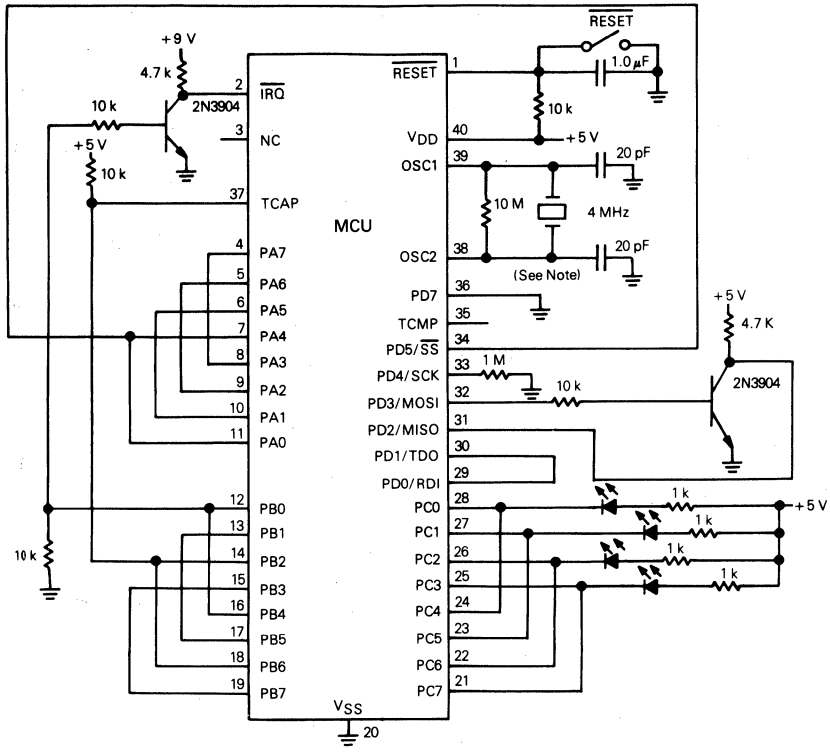
The self-check capability provides the ability to determine if the device is functional. Self-check is performed using the circuit shown in Figure 5. Port C pins PC0-PC3 are monitored for the self-check results. After reset, the following seven tests are performed automatically:

I/O — Exercise of ports A, B, and C
RAM —Counter test for each RAM byte
ROM —Exclusive OR with odd ones parity result
Timer – Tracks counter register and checks OCF flag
Interrupts — Tests external, timer, SCI and SPI interrupts
SCI — Transmission test; checks RDRF, TDRE, TC, and FE flags
SPI — Transmission test; checks SPIF, WCOL, and MODF flags

Self-check results (using the LEDs as monitors) are shown in Table 2. The following subroutines are available to the user and do not require any external hardware.

TIMER TEST SUBROUTINE

This subroutine returns with the Z bit cleared if any error is detected; otherwise, the Z bit is set. The timer test subroutine is called at location $1FOE. The output compare register is first set to the current timer state. Because the timer is free running and has only a divide-by-four prescaler, each timer count cannot be tested. The test reads the timer once every 10 counts (40 cycles) and checks for correct counting. The test tracks the counter until the timer wraps around, triggering the output compare flag in the timer status register. RAM locations $0050 and $0051 are overwritten. Upon return to the user's program, X=40. If the test passed, A=0.



NOTE: The RC Oscillator Option may also be used in this circuit.

Figure 5. Self-Check Circuit Schematic Diagram

Table 2. Self-Check Results

PC3	PC2	PC1	PC0	Remarks
1	0	0	1	Bad I/O
1	0	1	0	Bad RAM
1	0	1	1	Bad Timer
1	1	0	0	Bad SCI
1	1	0	1	Bad ROM
1	1	1	0	Bad SPI
1	1	1	1	Bad Interrupts or $\overline{IRQ}$ Request
Flashing				Good Device
All Others				Bad Device, Bad Port C, etc.

0 indicates LED is on; 1 indicates LED is off.

ROM CHECKSUM SUBROUTINE

This subroutine returns with the Z bit cleared if any error is detected; otherwise, the Z bit is set. The ROM checksum subroutine is called at location $1F93 with RAM location $0053 equal to $01 and A=0. A short routine is set up and executed in RAM to compute a checksum of the entire ROM pattern. RAM locations $0050 through $0053 are overwritten. Upon return to the user's program, X=0. If the test passed, A=0.

RESETS

The MCU can be reset two ways: by initial power-up and by the external reset input ($\overline{RESET}$). The $\overline{RESET}$ input consists mainly of a Schmitt trigger that senses the $\overline{RESET}$ line logic level.

POWER-ON RESET (POR)

An internal reset is generated on power-up to allow the internal clock generator to stabilize. The power-on reset is strictly for power turn-on conditions and should not be used to detect a drop in the power supply voltage. There is a 4064 internal processor clock cycle (t_{CYC}) delay after the oscillator becomes active. If the $\overline{RESET}$ pin is low at the end of 4046 t_{CYC}, the MCU will remain in the reset condition until $\overline{RESET}$ goes high.

EXTERNAL RESET INPUT

The MCU is reset when a logic zero is applied to the $\overline{RESET}$ input for a period of one and one-half machine cycles (t_{CYC}).

INTERRUPTS

The MCU can be interrupted five different ways: the four maskable hardware interrupts ($\overline{IRQ}$, SPI, SCI, and timer) and the nonmaskable software interrupt instruction (SWI).

Interrupts cause the processor to save register contents on the stack and to set the interrupt mask (I bit) to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack and normal processing to resume. The stacking order is shown in Figure 6.

Unlike $\overline{RESET}$, hardware interrupts do not cause the current instruction execution to be halted but are considered pending until the current instruction is complete.

NOTE

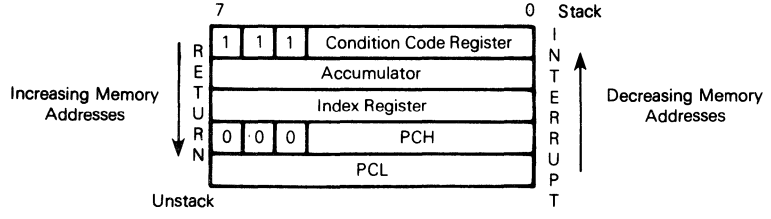
The current instruction is the one already fetched and being operated on.

When the current instruction is complete, the processor checks all pending hardware interrupts. If unmasked (I bit clear) and if the corresponding interrupt enable bit is set, the processor proceeds with interrupt processing; otherwise, the next instruction is fetched and executed.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction, regardless of the I-bit state. Refer to Figure 7 for the reset and interrupt instruction processing sequence.

TIMER INTERRUPT

There are three different timer interrupt flags that cause a timer interrupt whenever they are set and enabled. The interrupt flags are in the timer status register (TSR), and the enable bits are in the timer control register (TCR). Refer to **TIMER** for more information.



NOTE: Since the Stack Pointer decrements during pushes, the PCL is stacked first, followed by PCH, etc. Pulling from the stack is in the reverse order.

Figure 6. Interrupt Stacking Order

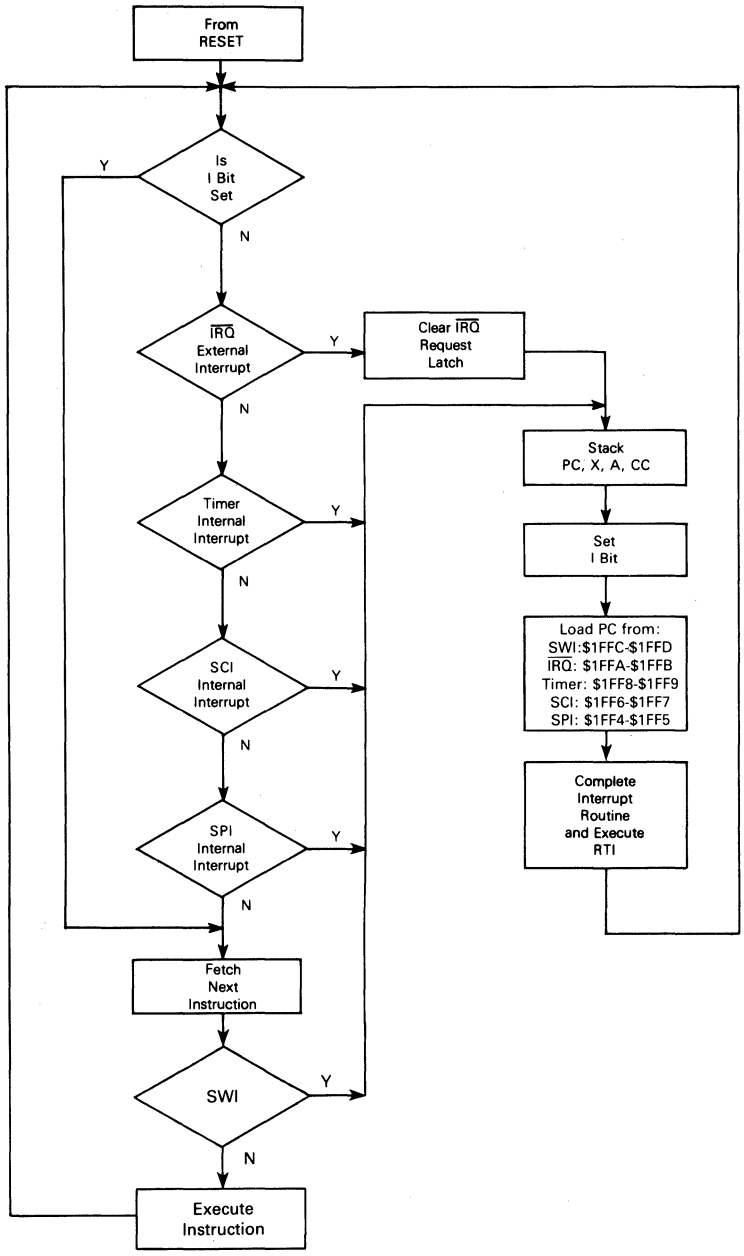


Figure 7. Reset and Interrupt Processing Flowchart

EXTERNAL INTERRUPT

If the interrupt mask bit (I bit) of the CCR is set, all interrupts are disabled. Clearing the I bit enables the external interrupt. The external interrupt is internally synchronized and then latched on the falling edge of $\overline{IRQ}$. The action of the external interrupt is identical to the timer interrupt with the exception that the interrupt request input at $\overline{IRQ}$ is latched internally and the service routine address is specified by the contents of $1FFA and $1FFB.

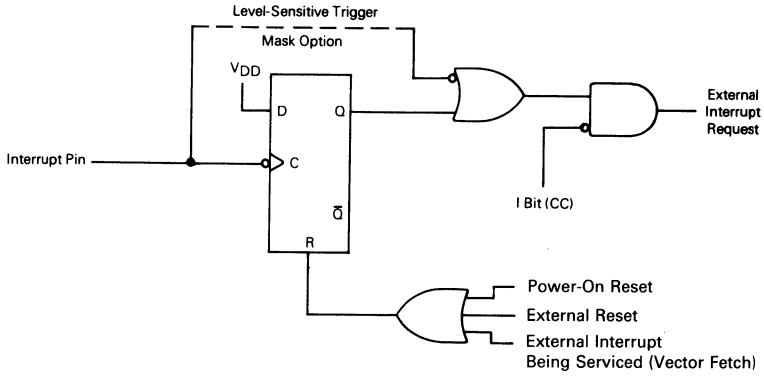
Either a level-sensitive and edge-sensitive trigger, or an edge-sensitive-only trigger are available as a mask option. Figure 8 shows both a functional internal diagram and a mode timing diagram for the interrupt line. The timing diagram shows two treatments of the interrupt line to the processor. The first method shows a single pulse on the interrupt line spaced far enough apart to be serviced. The minimum time between pulses is a function of the length of the interrupt service. Once a pulse occurs, the next pulse should not occur until an RTI occurs. This time (t_{ILIL}) is obtained by adding 21 instruction cycles to the total number of cycles it takes to complete the service routine (not including the RTI instruction). The second method shows many interrupt lines "wire-ORed" to form the interrupts at the processor. If the interrupt line remains low after servicing an interrupt, then the next interrupt is recognized.

NOTE

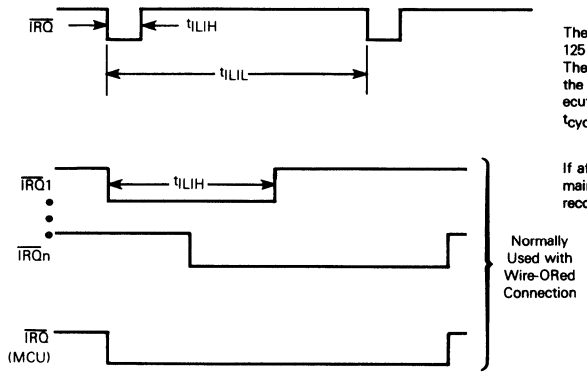
The internal interrupt latch is cleared in the first part of the interrupt service routine; therefore, one external interrupt pulse could be latched and serviced as soon as the I bit is cleared.

SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. If the I bit is zero, SWI executes after the other interrupts. The SWI



(a) Interrupt Internal Function Diagram



Edge-Sensitive Trigger Condition
The minimum pulse width (t_{ILIH}) is either 125 ns (V_{DD} = 5 V) or 250 ns (V_{DD} = 3 V). The period t_{ILIL} should not be less than the number of t_{cyc} cycles it takes to execute the interrupt service routine plus 21 t_{cyc} cycles.

Level-Sensitive Trigger Condition
If after servicing an interrupt the $\overline{IRQ}$ remains low, then the next interrupt is recognized.

(b) Interrupt Mode Diagram

Figure 8. External Interrupt

operation is similar to the hardware interrupts. The interrupt service routine address is specified by the contents of memory locations $1FFC and $1FFD.

SCI INTERRUPTS

An interrupt in the SCI occurs when one of the interrupt flag bits in the serial communications status register is set, provided the I bit in the CCR is clear and the enable bit in the serial communications control register 2 is set. Software in the serial interrupt service routine must determine the cause and priority of the SCI interrupt by examining the interrupt flags and status bits in the SCI status register.

SPI INTERRUPTS

An interrupt in the SPI occurs when one of the interrupt flag bits in the serial peripheral status register is set, provided the I bit in the CCR is clear and the enable bit in the serial peripheral control register is set. Software in the serial peripheral interrupt service routine must determine the cause and priority of the SPI interrupt by examining the interrupt flag bits in the SPI status register.

3

LOW-POWER MODES

STOP

The STOP instruction places the MCU in its lowest power consumption mode. In the STOP mode, the internal oscillator is turned off, halting all internal processing including timer, SCI, and SPI operation (refer to Figure 9).

During the STOP mode, the TCR bits are altered to remove any pending timer interrupt request and to disable any further timer interrupts. The timer prescaler is cleared. The I bit in the CCR is cleared to enable external interrupts. All other registers and memory remain unaltered. All input/output lines remain unchanged. The processor can be brought out of the STOP mode only by an external interrupt or reset.

SCI during STOP Mode

When the MCU enters the STOP mode, the baud rate generator stops, halting all SCI activity. If the STOP instruction is executed during a transmitter transfer, that transfer is halted. If a low input to the $\overline{IRQ}$ pin is used to exit STOP mode, the transfer resumes. If the SCI receiver is receiving data and the STOP mode is entered, received data sampling stops because the baud rate generator stops, and all subsequent data is lost. For these reasons, all SCI transfers should be in the idle state when the STOP instruction is executed.

SPI during Stop Mode

When the MCU enters the STOP mode, the baud rate generator stops, terminating all master mode SPI operations. If the STOP instruction is executed during an SPI transfer, that transfer halts until the MCU exits the STOP mode by a low signal on the $\overline{IRQ}$ pin. If reset is used to exit the STOP mode, then the SPI control and status bits are cleared, and the SPI is disabled. If the MCU is in the slave mode when the STOP instruction is executed, the slave SPI continues to operate and can still accept data and clock information in addition to transmitting its own data back to a master device.

At the end of a possible transmission with a slave SPI in the STOP mode, no flags are set until a low on the $\overline{IRQ}$ pin wakes up the MCU. Caution should be observed when operating the SPI as a slave during the STOP mode because the protective circuitry (WCOL, MODF, etc.) is inactive.

WAIT

The WAIT instruction places the MCU in a low-power consumption mode, but the WAIT mode consumes more power than the STOP mode. All CPU action is suspended, but the timer, SCI, and SPI remain active (refer to Figure 10). An interrupt from the timer, SCI, or SPI can cause the MCU to exit the WAIT mode.

During the WAIT mode, the I bit in the CCR is cleared to enable interrupts. All other registers, memory, and input/output lines remain in their previous state. The timer

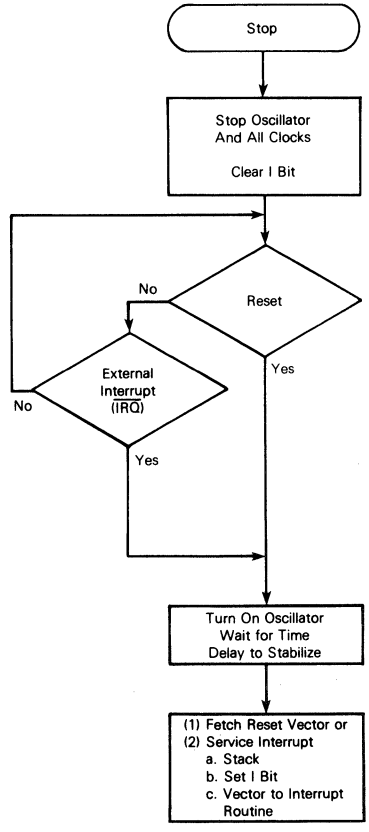


Figure 9. STOP Function Flowchart

may be enabled to allow a periodic exit from the WAIT mode.

DATA RETENTION MODE

The contents of RAM and CPU registers are retained at supply voltages as low as 2.0 Vdc. This is called the data retention mode where the data is held, but the device is not guaranteed to operate. The MCU should be in $\overline{\text{RESET}}$ during data retention mode.

TIMER

The timer consists of a 16-bit, software-programmable counter driven by a fixed divide-by-four prescaler. This timer can be used for many purposes, including input waveform measurements while simultaneously generating an output waveform. Pulse widths can vary from several microseconds to many seconds. Refer to Figure 11 for a timer block diagram.

Because the timer has a 16-bit architecture, each specific functional segment (capability) is represented by two registers. These registers contain the high and low byte of that functional segment. Generally, accessing the low byte of a specific timer function allows full control of that function; however, an access of the high byte inhibits that specific timer function until the low byte is also accessed.

NOTE

The I bit in the CCR should be set while manipulating both the high and low byte registers of a specific timer function to ensure that an interrupt does not occur.

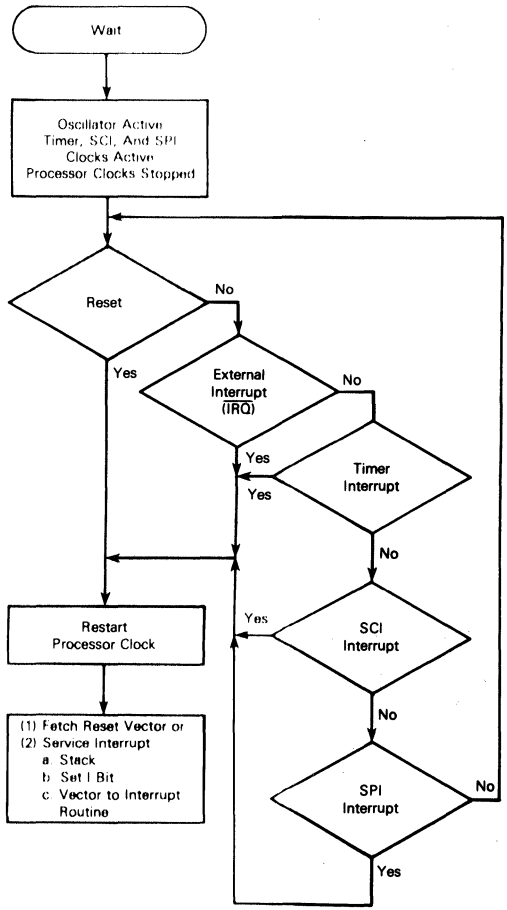


Figure 10. WAIT Function Flowchart

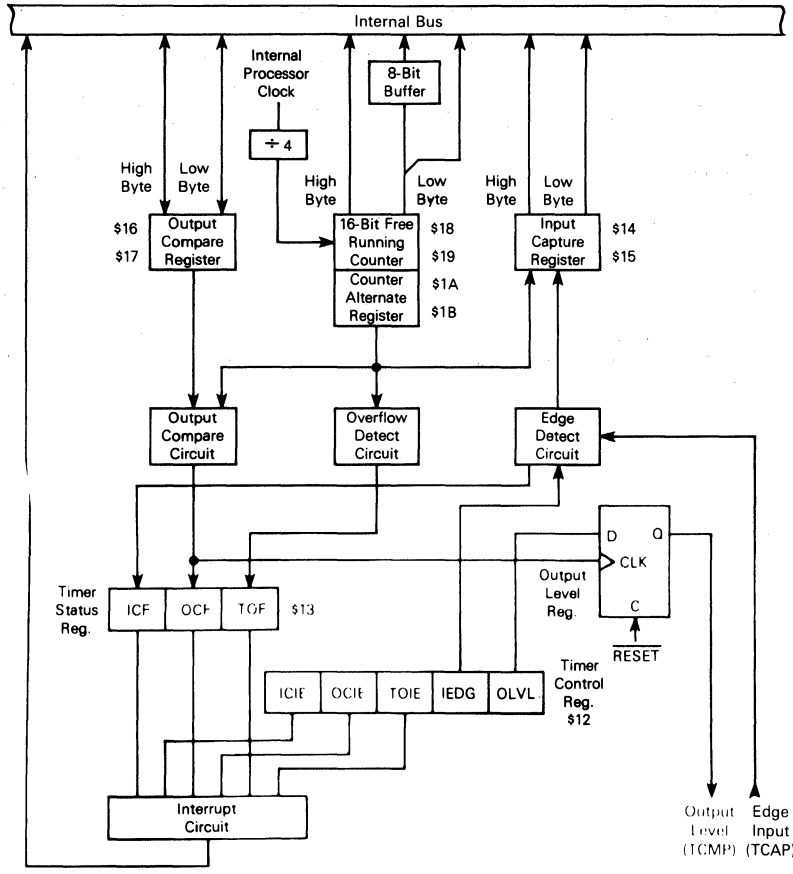


Figure 11. Timer Block Diagram

COUNTER

The key element in the programmable timer is a 16-bit, free-running counter or counter register, preceded by a prescaler that divides the internal processor clock by four. The prescaler gives the timer a resolution of 2.0 microseconds if the internal bus clock is 2.0 MHz. The counter is incremented during the low portion of the internal bus clock. Software can read the counter at any time without affecting its value.

The double-byte, free-running counter can be read from either of two locations, $18–$19 (counter register) or $1A–$1B (counter alternate register). A read from only the least significant byte (LSB) of the free-running counter ($19, $1B) receives the count value at the time of the read. If a read of the free-running counter or counter alternate register first addresses the most significant byte (MSB) ($18, $1A), the LSB ($19, $1B) is transferred to a buffer. This buffer value remains fixed after the first MSB read, even if the user reads the MSB several times. This buffer is accessed when reading the free-running counter or counter alternate register LSB ($19 or $1B) and, thus, completes a read sequence of the total counter value. In reading either the free-running counter or counter alternate register, if the MSB is read, the LSB must also be read to complete the sequence.

The counter alternate register differs from the counter register in one respect: a read of the counter register MSB can clear the timer overflow flag (TOF). Therefore, the counter alternate register can be read at any time without the possibility of missing timer overflow interrupts due to clearing of the TOF.

The free-running counter is configured to $FFFC during reset and is always a read-only register. During a power-on reset, the counter is also preset to $FFFC and begins

running after the oscillator start-up delay. Because the free-running counter is 16 bits preceded by a fixed divide-by-four prescaler, the value in the free-running counter repeats every 262,144 internal bus clock cycles. When the counter rolls over from $FFFF to $0000, the TOF bit is set. An interrupt can also be enabled when counter rollover occurs by setting its interrupt enable bit (TOIE).

OUTPUT COMPARE REGISTER

The 16-bit output compare register is made up of two 8-bit registers at locations $16 (MSB) and $17 (LSB). The output compare register is used for several purposes, such as indicating when a period of time has elapsed. All bits are readable and writable and are not altered by the timer hardware or reset. If the compare function is not needed, the two bytes of the output compare register can be used as storage locations.

The output compare register contents are compared with the contents of the free-running counter continually, and if a match is found, the corresponding output compare flag (OCF) bit is set and the corresponding output level (OLCL) bit is clocked to an output level register. The output compare register values and the output level bit should be changed after each successful comparison to establish a new elapsed timeout. An interrupt can also accompany a successful output compare provided the corresponding interrupt enable bit (OCIE) is set.

After a processor write cycle to the output compare register containing the MSB ($16), the output compare function is inhibited until the LSB ($17) is also written. The user must write both bytes (locations) if the MSB is written first. A write made only to the LSB ($17) will not inhibit the compare function. The free-running counter is updated every four internal bus clock cycles. The minimum time required to update the output compare register is a function of the program rather than the internal hardware.

The processor can write to either byte of the output compare register without affecting the other byte. The output level (OLVL) bit is clocked to the output level register regardless of whether the output compare flag (OCF) is set or clear.

INPUT CAPTURE REGISTER

Two 8-bit registers, which make up the 16-bit input capture register, are read-only and are used to latch the value of the free-running counter after the corresponding input capture edge detector senses a defined transition. The level transition which triggers the counter transfer is defined by the corresponding input edge bit (IEDG). Reset does not affect the contents of the input capture register.

The result obtained by an input capture will be one more than the value of the free-running counter on the rising edge of the internal bus clock preceding the external transition. This delay is required for internal synchronization. Resolution is one count of the free-running counter, which is four internal bus clock cycles.

The free-running counter contents are transferred to the input capture register on each proper signal transition regardless of whether the input capture flag (ICF) is set or clear. The input capture register always contains the free-running counter value that corresponds to the most recent input capture.

After a read of the input capture register ($14) MSB, the counter transfer is inhibited until the LSB ($15) is also read. This characteristic causes the time used in the input capture software routine and its interaction with the main program to determine the minimum pulse period.

A read of the input capture register LSB ($15) does not inhibit the free-running counter transfer since they occur on opposite edges of the internal bus clock.

TIMER CONTROL REGISTER (TCR) $12

The TCR is a read/write register containing five control bits. Three bits control interrupts associated with the timer status register flags ICF, OCF, and TOF.

7	6	5	4	3	2	1	0
ICIE	OCIE	TOIE	0	0	0	IEDG	OLVL
RESET:							
0	0	0	0	0	0	U	0

ICIE — Input Capture Interrupt Enable
- 1 = Interrupt enabled
- 0 = Interrupt disabled

OCIE — Output Compare Interrupt Enable
- 1 = Interrupt enabled
- 0 = Interrupt disabled

TOIE — Timer Overflow Interrupt Enable
- 1 = Interrupt enabled
- 0 = Interrupt disabled

IEDG — Input Edge

Value of input edge determines which level transition on TCAP pin will trigger free-running counter transfer to the input capture register
- 1 = Positive edge
- 0 = Negative edge

Reset does not affect te IEDG bit (U = unaffected).

OLVL — Output Level

Value of output level is clocked into output level register by the next successful output compare and will appear on the TCMP pin
- 1 = High output
- 0 = Low output

Bits 2, 3, and 4 — Not used

Always read zero

TIMER STATUS REGISTER (TSR) $13

The TSR is a read-only register containing three status flag bits.

7	6	5	4	3	2	1	0
ICF	OCF	TOF	0	0	0	0	0
RESET:							
U	U	U	0	0	0	0	0

ICF — Input Capture Flag
- 1 = Flag set when selected polarity edge is sensed by input capture edge detector
- 0 = Flag cleared when TSR and input capture low register ($15) are accessed

OCF — Output Compare Flag
1 = Flag set when output compare register contents match the free-running counter contents
0 = Flag cleared when TSR and output compare low register ($17) are accessed
TOF — Timer Overflow Flag
1 = Flag set when free-running counter transition from $FFFF to $0000 occurs
0 = Flag cleared when TSR and counter low register ($19) are accessed
Bits 0–4 — Not used
Always read zero

Accessing the timer status register satisfies the first condition required to clear status bits. The remaining step is to access the register corresponding to the status bit.

A problem can occur when using the timer overflow function and reading the free-running counter at random times to measure an elapsed time. Without incorporating the proper precautions into software, the timer overflow flag could unintentionally be cleared if:

1) The timer status register is read or written when TOF is set, and
2) The LSB of the free-running counter is read but not for the purpose of servicing the flag.

The counter alternate register at address $1A and $1B contains the same value as the free-running counter (at address $18 and $19); therefore, this alternate register can be read at any time without affecting the timer overflow flag in the timer status register.

TIMER DURING WAIT MODE

The CPU clock halts during the WAIT mode, but the timer remains active. An interrupt from the timer causes the processor to exit the WAIT mode.

TIMER DURING STOP MODE

In the STOP mode, the timer stops counting and holds the last count value if STOP is exited by an interrupt. If RESET is used, the counter is forced to $FFFC. During STOP, if at least one valid input capture edge occurs at the TCAP pin, the input capture detect circuit is armed. This does not set any timer flags nor wake up the MCU, but when the MCU does wake up, there is an active input capture flag and data from the first valid edge that occurred during the STOP mode. If RESET is used to exit STOP mode, then no input capture flag or data remains, even if a valid input capture edge occurred.

SERIAL COMMUNICATIONS INTERFACE

A full-duplex asynchronous SCI is provided with a standard NRZ format and a variety of baud rates. The SCI transmitter and receiver are functionally independent but use the same data format and baud rate. The terms baud and bit rate are used synonymously in the following description.

SCI TWO-WIRE SYSTEM FEATURES

- Standard NRZ (mark/space) format
- Advanced error detection method includes noise detection for noise duration of up to one-sixteenth bit time
- Full-duplex operation (simultaneous transmit and receive)
- Software programmable for one of 32 different baud rates
- Software-selectable word length (eight- or nine-bit words)
- Separate transmitter and receiver enable bits
- SCI may be interrupt driven
- Four separate interrupt conditions

SCI RECEIVER FEATURES

- Receiver wake-up function (idle or address bit)
- Idle line detect
- Framing error detect
- Noise detect
- Overrun detect
- Receiver data register full flag

SCI TRANSMITTER FEATURES

- Transmit data register empty flag
- Transmit complete flag
- Break send

Any SCI two-wire system requires receive data in (RDI) and transmit data out (TDO).

DATA FORMAT

Receive data in (RDI) or transmit data out (TDO) is the serial data presented between the internal data bus and the output pin (TDO) and between the input pin (RDI) and the internal data bus. Data format is as shown for the NRZ in Figure 12.

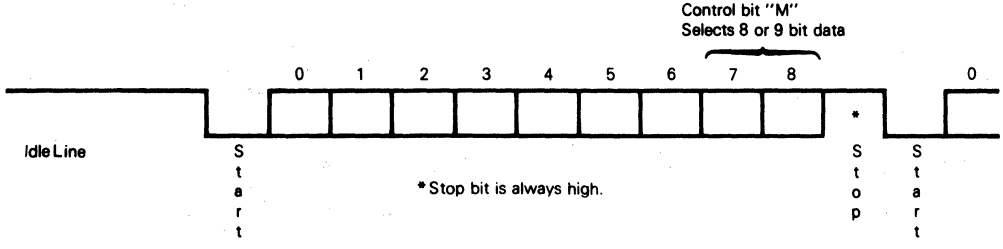


Figure 12. Data Format

WAKE-UP FEATURE

In a typical multiprocessor configuration, the software protocol will usually identify the addressee(s) at the beginning of the message. To permit uninterested MPUs to ignore the remainder of the message, a wake-up feature is included, whereby all further SCI receiver flag (and interrupt) processing can be inhibited until its data line returns to the idle state. An SCI receiver is re-enabled by an idle string of at least ten (or eleven) consecutive ones. Software for the transmitter must provide for the required idle string between consecutive messages and prevent it from occurring within messages.

A second wake-up method is available in which sleeping SCI receivers can be awakened by a logic one in the high-order bit of a received character.

RECEIVE DATA IN

Receive data in (RDI) is the serial data which is presented from the input pin via the SCI to the receive data register (RDR). While waiting for a start bit, the receiver samples the input at a rate 16 times higher than the set baud rate. This increased rate is referred to as the RT rate. When the input (idle) line is detected low, it is tested for three more sample times. If at least two of these three samples detect a logic low, a valid start bit is assumed to be detected. If in two or more samples, a logic high is detected, the line is assumed to be idle. The receive clock generator is controlled by the baud rate register (see Figure 13); however, the SCI is synchronized by the start bit independent of the transmitter. Once a valid start bit is detected, the start bit, each data bit, and the stop bit are each sampled three times. The value of the bit is determined by voting logic, which takes the value of a majority of samples. A noise flag is set when all three samples on a valid start bit, data bit, or stop bit do not agree. A noise flag is also set when the start verification samples do not agree.

START BIT DETECTION FOLLOWING A FRAMING ERROR

If there has been a framing error (FE) without detection of a break (10 zeros for 8-bit format or 11 zeros for a 9-bit format), the circuit continues to operate as if there actually were a stop bit, and the start edge will be placed artificially. The last bit received in the data shift register is inverted to a logic one, and the three logic-one start qualifiers are forced into the sample shift register during the interval when detection of a start bit is anticipated; therefore, the start bit will be accepted no sooner than it is anticipated.

If the receiver detects that a break (RDRF = 1, FE = 1, receiver data register = $00) produced the framing error, the start bit will not be artificially induced, and the receiver must actually receive a logic one before start.

TRANSMIT DATA OUT

Transmit data out (TDO) is the serial data presented from the transmit data register (TDR) via the SCI to the output pin. The transmitter generates a bit time by using a derivative of the RT clock, producing a transmission rate equal to one-sixteenth that of the receiver sample clock.

FUNCTIONAL DESCRIPTION

A block diagram of the SCI is shown in Figure 13. The user has option bits in the serial communications control register 1 (SCCR1) to determine the SCI wake-up method and data word length. Serial communications control register 2 (SCCR2) provides control bits that individually enable/disable the transmitter or receiver, enable system interrupts, and provide wake-up enable, and send break code bits. The baud rate register bits allow the user to select different baud rates, which are used as the rate control for the transmitter and receiver.

Data transmission is initiated by a write to the serial communications data register (SCDAT). Provided the transmitter is enabled, data stored in the SCDAT is transferred to the transmit data shift register. This data transfer sets the SCI status register (SCSR) transmit data register empty (TDRE) bit and generates an interrupt if the transmit interrupt is enabled. Data transfer to the transmit data shift register is synchronized with the bit rate clock. All data is transmitted LSB first. Upon completion of data transmission, the transmission complete (TC) bit is set (provided no pending data, preamble, or break code is sent), and an interrupt is generated if the transmit complete interrupt is enabled. If the transmitter is disabled, and the data, preamble, or break code has been sent, the TC bit will also be set, which will also generate an interrupt if the TCIE bit is set. If the transmitter is disabled in the middle of a transmission, that character will be completed before the transmitter gives up control of the TDO pin.

When the SCDAT is read, it contains the last data byte received, provided that the receiver is enabled. The SCSR receive data register full (RDRF) bit is set to indicate that a data byte is transferred from the input serial shift register to the SCDAT, which can cause an interrupt if the receiver interrupt is enabled. Data transfer from the input serial shift register to the SCDAT is synchronized by the receiver bit rate clock. The SCSR overrun (OR), noise flag (NF), or FE bits are set if data reception errors occur.

An idle line interrupt is generated if the idle line interrupt is enabled and the SCSR IDLE bit (which detects idle line transmission) is set. This allows a receiver that is not in the wake-up mode to detect the end of a message, the preamble of a new message, or to resynchronize with the transmitter. A valid character must be received before the idle line condition for the IDLE bit to be set and for an idle line interrupt to be generated.

REGISTERS

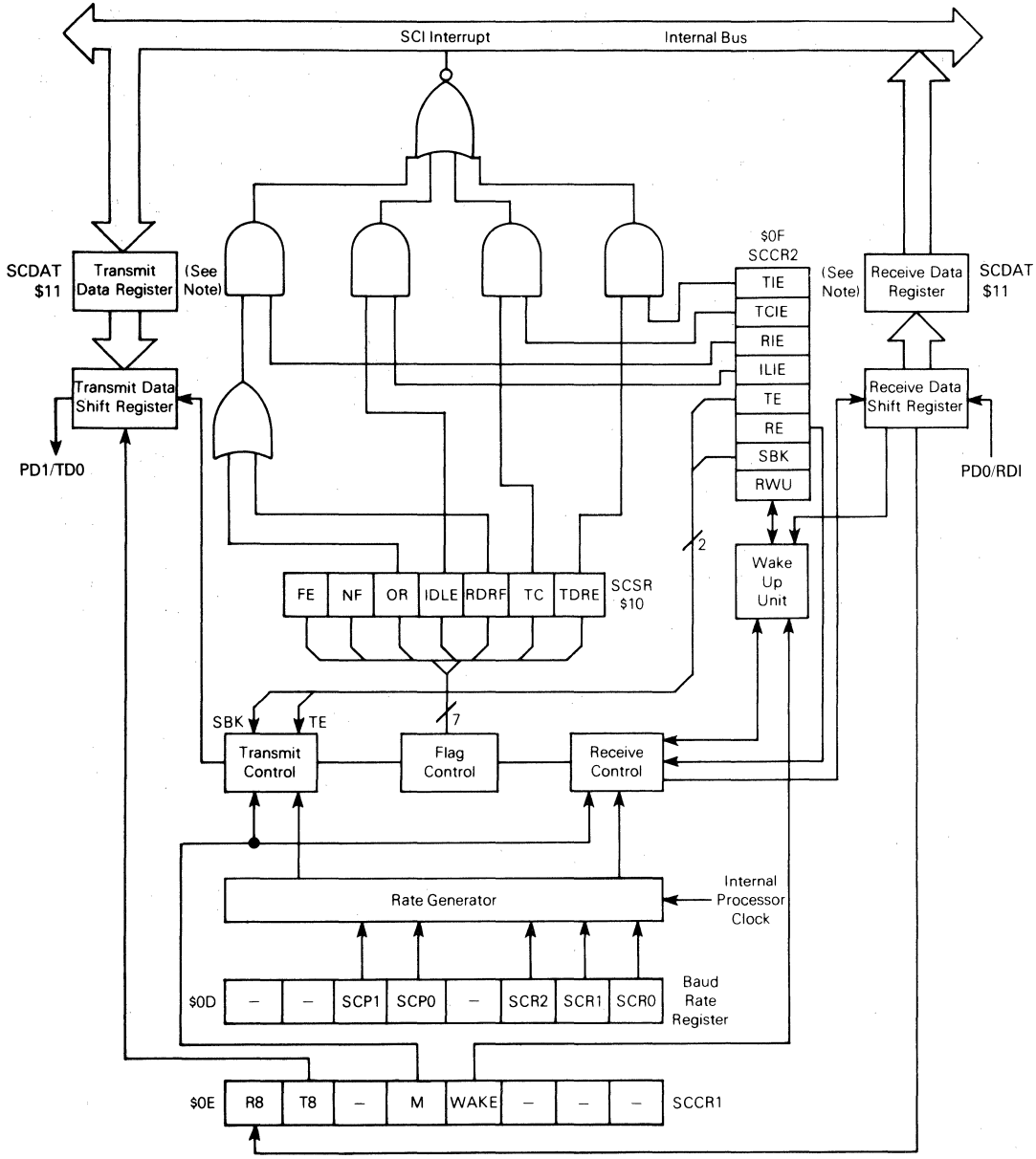
There are five registers used in the SCI; the internal configuration of these registers is discussed in the following paragraphs.

Serial Communications Data Register (SCDAT) $11

The SCDAT is a read/write register used to receive and transmit SCI data.

7	6	5	4	3	2	1	0
SCD7	SCD6	SCD5	SCD4	SCD3	SCD2	SCD1	SCD0
RESET:							
U	U	U	U	U	U	U	U

As shown in Figure 13, SCDAT functions as two separate registers. The transmit data register (TDR) provides the parallel interface from the internal data bus to the transmit shift register. The receive data register (RDR) provides the interface from the receive shift register to the internal data bus.



NOTE: The Serial Communications Data Register (SCDAT) is controlled by the internal R/W signal. It is the transmit data register when written and receive data register when read.

Figure 13. SCI Block Diagram

Serial Communications Control Register 1 (SCCR1) $0E

The SCCR1 provides control bits that determine word length and select the wake-up method.

7	6	5	4	3	2	1	0
R8	T8	—	M	WAKE	—	—	—
RESET:							
U	U	—	U	U	—	—	—

R8 — Receive Data Bit 8
R8 bit provides storage location for the ninth bit in the receive data byte (if M = 1).
T8 — Transmit Data Bit 8
T8 bit provides storage location for the ninth bit in the transmit data byte (if M = 1).
M — SCI Character Word Length
1 = one start bit, nine data bits, one stop bit
0 = one start bit, eight data bits, one stop bit
WAKE — Wake-Up Select
Wake bit selects the receiver wake-up method.
1 = Address bit (most significant bit)
0 = Idle line condition
Bits 0–2, and 5 — Not used
Can read either one or zero

The address bit is dependent on both the wake-bit and the M-bit level. Additionally, the receiver does not use the wake-up feature unless the RWU control bit in SCCR2 is set.

Wake	M	Receiver Wake-Up
0	X	Detection of an idle line allows the next data byte received to cause the receive data register to fill and produce an RDRF flag.
1	0	Detection of a received one in the eighth data bit allows an RDRF flag and associated error flags.
1	1	Detection of a received one in the ninth data bit allows an RDRF flag and associated error flags.

Serial Communications Control Register 2 (SCCR2) $0F

The SCCR2 provides control of individual SCI functions such as interrupts, transmit/receive enabling, receiver wake-up, and break code.

7	6	5	4	3	2	1	0
TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
RESET:							
0	0	0	0	0	0	0	0

TIE — Transmit Interrupt Enable
1 = SCI interrupt enabled
0 = TDRE interrupt disabled
TCIE — Transmit Complete Interrupt Enable
1 = SCI interrupt enabled
0 = TC interrupt disabled
RIE — Receive Interrupt Enable
1 = SCI interrupt enabled
0 = RDRF and OR interrupts disabled
ILIE — Idle Line Interrut Enable
1 = SCI interrupt enabled
0 = Idle interrupt disabled
TE — Transmit Enable
1 = Transmit shift register output is applied to the TD0 line. Depending upon the SCCR1 M bit, a preamble of 10 (M = 0) or 11 (M = 1) consecutive ones is transmitted.
0 = Transmitter disabled after last byte is loaded in the SCDAT and TDRE is set. After last byte is transmitted, TD0 line becomes a high-impedance line.
RE — Receive Enable
1 = Receiver shift register input is applied to the RDI line.
0 = Receiver disabled and RDRF, IDLE, OR, NF, and FE status bits are inhibited.
RWU — Receiver Wake-Up
1 = Places receiver in sleep mode and enables wake-up function
0 = Wake-up function disabled after receiving data word with MSB set (if WAKE = 1)
Wake-up function also disabled after receiving 10 (M = 0) or 11 (M = 1) consecutive ones (if WAKE = 0)
SBK — Send Break
1 = Transmitter continually sends blocks of zeros (sets of 10 or 11) until cleared. Upon completion of break code, transmitter sends one high bit for recognition of valid start bit.
0 = Transmitter sends 10 (M = 0) or 11 (M = 1) zeros then reverts to an idle state or continues sending data. If transmitter is empty and idle, setting and clearing the SBK bit may queue up to two character times of break because the first break transfers immediately to the shift register, and the second is queued into the parallel transmit buffer.

Serial Communications Status Register (SCSR) $10

The SCSR provides inputs to the SCI interrupt logic circuits. Noise flag and framing error bits are also contained in the SCSR.

7	6	5	4	3	2	1	0
TDRE	TC	RDRF	IDLE	OR	NF	FE	—
RESET:							
1	1	0	0	0	0	0	—

TDRE — Transmit Data Register (TDR) Empty
1 = TDR contents transferred to the transmit data shift register
0 = TDR still contains data. TDRE is cleared by reading the SCSR (with TDRE = 1), followed by a write to the TDR.
TC — Transmit Complete
1 = Indicates end of data frame, preamble, or break condition has occurred
0 = TC bit cleared by reading the SCSR (with TC = 1), followed by a write to the TDR
RDRF — Receive Data Register (RDR) Full
1 = Receive data shift register contents transferred to the RDR

0 = Receive data shift register transfer did not occur. RDRF is cleared by reading the SCSR (with RDRF = 1) followed by a read of the RDR

IDLE — Idle Line Detect

1 = Indicates receiver has detected an idle line

0 = IDLE is cleared by reading the SCSR (with IDLE = 1), followed by a read of the RDR. Once IDLE is cleared, IDLE cannot be set until RDI line becomes active and idle again.

OR — Overrun Error

1 = Indicates receive data shift register data is sent to a full RDR (RDRF = 1). Data causing the overrun is lost, and RDR data is not disturbed.

0 = OR is cleared by reading the SCSR (with OR = 1), followed by a read of the RDR.

NF — Noise Flag

1 = Indicates noise is present on the receive bits, including the start and stop bits. NF is not set until RDRF = 1.

0 = NF is cleared by reading the SCSR (with NF = 1), followed by a read of the RDR.

FE — Framing Error

1 = Indicates stop bit not detected in received data character. FE is set the same time RDRF is set. If received byte causes both framing and overrun errors, processor will only recognize the overrun error. Further data transfer into the RDR is inhibited until FE is cleared.

0 = NF is cleared by reading the SCSR (with FE = 1), followed by a read of the RDR.

Bit 0 — Not used

Can read either one or zero

Baud Rate Register $0D

The baud rate register is used to select the SCI transmitter and receiver baud rate. SCP0 and SCP1 prescaler bits are used in conjunction with the SCR0 through SCR2 baud rate bits to provide multiple baud rate combinations for a given crystal frequency. Bits 3, 6, and 7 always read zero.

7	6	5	4	3	2	1	0
—	—	SCP1	SCP0	—	SCR2	SCR1	SCR0
RESET:							
—	—	0	0	—	U	U	U

SCP0 — SCI Prescaler Bit 0

SCP1 — SCI Prescaler Bit 1

Two prescaler bits are used to increase the range of standard baud rates controlled by the SCR0–SCR2 bits. Prescaler internal processor clock division versus bit levels are listed in Table 3.

SCR0 — SCI Baud Rate Bit 0

SCR1 — SCI Baud Rate Bit 1

SCR2 — SCI Baud Rate Bit 2

Three baud rate bits are used to select the baud rates of the SCI transmitter and SCI receiver. Baud rates versus bit levels are listed in Table 4.

Tables 3 and 4 tabulate the divide chain used to obtain the baud rate clock (transmit clock). The actual divider chain is controlled by the combined SCP0-SCP1 and SCR0-SCR2 bits in the baud rate register. All divided frequencies shown in Table 3 represent the final baud rate resulting from the internal processor clock division shown in the divided-by column only (prescaler division only). Table 4 lists the prescaler output divided by the action of the SCI select bits (SCR0-SCR2). For example, assume that a 9600-Hz baud rate is required with a 2.4576-MHz external crystal. In this case, the prescaler bits (SCP0-SCP1) could be configured as a divide-by-one or a divide-by-four. If a divide-by-four prescaler is used, then the SCR0-SCR2 bits must be configured as a divide-by-two. Using the same crystal, the 9600 baud rate can be obtained with a prescaler divide-by-one and the SCR0-SCR2 bits configured for a divide-by-eight.

SERIAL PERIPHERAL INTERFACE

The serial peripheral interface (SPI) is an interface built into the MCU which allows several MCUs or MCUs plus peripherals to be interconnected within the same black box. In the SPI format, the clock is not included in the data stream and must be furnished as a separate signal. An SPI system may consist of one master MCU and several slaves (Figure 14) or MCUs that can be either masters or slaves.

Features:

- Full-duplex, three-wire synchronous transfers
- Master or slave operation

Table 3. Prescaler Highest Baud Rate Frequency Output

SCP Bit		Clock* Divided By	Crystal Frequency MHz				
1	0		4.194304	4.0	2.4576	2.0	1.8432
0	0	1	131.072 kHz	125.000 kHz	76.80 kHz	62.50 kHz	57.60 kHz
0	1	3	43.691 kHz	41.666 kHz	25.60 kHz	20.833 kHz	19.20 kHz
1	0	4	32.768 kHz	31.250 kHz	19.20 kHz	15.625 kHz	14.40 kHz
1	1	13	10.082 kHz	9600 Hz	5.907 kHz	4800 Hz	4430 Hz

*Refers to the internal processor clock.

NOTE: The divided frequencies shown in Table 3 represent baud rates which are the highest transmit baud rate (Tx) that can be obtained by a specific crystal frequency and only using the prescaler division. Lower baud rates may be obtained by providing a further division using the SCI rate select bits as shown below for some representative prescaler outputs.

Table 4. Transmit Baud Rate Output for a Given Prescaler Output

SCR Bits			Divided By	Representative Highest Prescaler Baud Rate Output				
2	1	0		131.072 kHz	32.768 kHz	76.80 kHz	19.20 kHz	9600 Hz
0	0	0	1	131.072 kHz	32.768 kHz	76.80 kHz	19.20 kHz	9600 Hz
0	0	1	2	65.536 kHz	16.384 kHz	38.40 kHz	9600 Hz	4800 Hz
0	1	0	4	32.768 kHz	8.192 kHz	19.20 kHz	4800 Hz	2400 Hz
0	1	1	8	16.384 kHz	4.096 kHz	9600 Hz	2400 Hz	1200 Hz
1	0	0	16	8.192 kHz	2.048 kHz	4800 Hz	1200 Hz	600 Hz
1	0	1	32	4.096 kHz	1.024 kHz	2400 Hz	600 Hz	300 Hz
1	1	0	64	2.048 kHz	512 Hz	1200 Hz	300 Hz	150 Hz
1	1	1	128	1.024 kHz	256 Hz	600 Hz	150 Hz	75 Hz

NOTE: Table 4 illustrates how the SCI select bits can be used to provide lower transmitter baud rates by further dividing the prescaler output frequency. The five examples are only representative samples. In all cases, the baud rates shown are transmit baud rates (transmit clock), and the receive clock is 16 times higher in frequency than the actual baud rate.

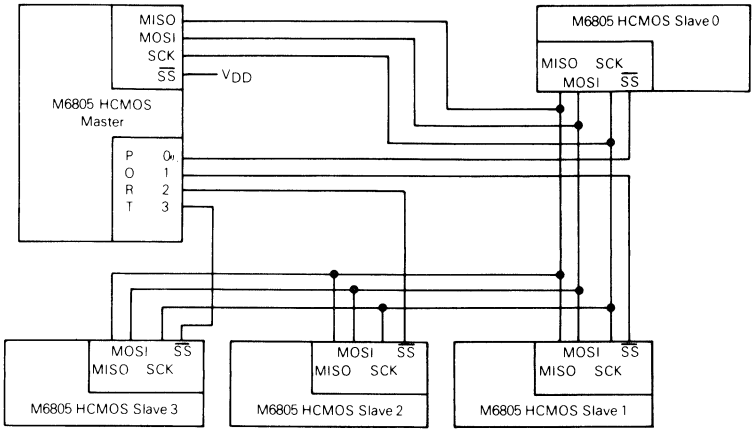


Figure 14. Master-Slave System Configuration

- 1.05 MHz (maximum) master bit frequency
- 2.1 MHz (maximum) slave bit frequency
- Four programmable master bit rates
- Programmable clock polarity and phase
- End-of-transmission interrupt flag
- Write collision flag protection
- Master-master mode fault protection capability

SIGNAL DESCRIPTION

The four basic signals (MOSI, MISO, SCK, and $\overline{SS}$) are described in the following paragraphs. Each signal function is described for both master and slave mode.

Master Out, Slave In

The master out, slave in (MOSI) line is configured as an output in a master device and as an input in a slave device. The MOSI line is one of two lines that transfer serial data in one direction with the most significant bit sent first.

Master In, Slave Out

The master in, slave out (MISO) line is configured as an input in a master device and as an output in a slave device. The MISO is one of two lines that transfer serial data in one direction with the most significant bit sent first. The MISO line of a slave device is placed in a high-impedance state if slave is not selected ($\overline{SS}=1$).

Serial Clock

The serial clock (SCK) is used to synchronize both data in and out of a device via the MOSI and MISO lines. The master and slave devices can exchange a byte of information during a sequence of eight clock cycles. Since SCK is generated by the master device, this line becomes an input on a slave device.

As shown in Figure 15, four possible timing relationships may be chosen by using control bits CPOL and CPHA in the serial peripheral control register (SPCR). Both master and slave devices must operate with the same timing.

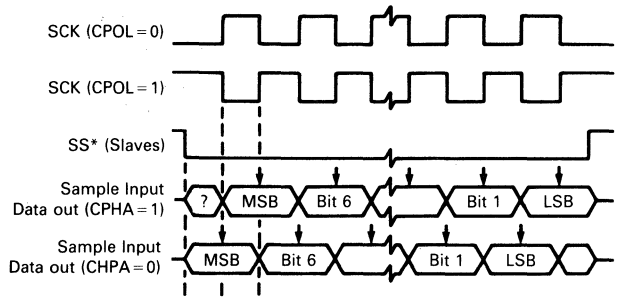


Figure 15. Data Clock Timing Diagram

Two bits (SPR0 and SPR1) in the SPCR of the master device select the clock rate. In a slave device, SPR0 and SPR1 have no effect on SPI operation.

Slave Select

The slave select ($\overline{SS}$) input line selects a slave device. The $\overline{SS}$ line must be low prior to data transactions and must stay low for the duration of the transaction. The $\overline{SS}$ line on the master must be tied high; if the $\overline{SS}$ line goes low, a mode fault error flag (MODF) is set in the serial peripheral status register (SPSR).

When CPHA = 0, the shift clock is the OR of $\overline{SS}$ with SCK. In this clock phase mode, $\overline{SS}$ must go high between successive characters in an SPI message. When CPHA = 1, $\overline{SS}$ must go high between successive characters in an SPI message. When CPHA = 1, $\overline{SS}$ may be left low for several SPI characters. In cases where there is only one SPI slave MCU, the slave MCU $\overline{SS}$ line could be tied to V_{SS} as long as CPHA = 1 clock modes are used.

FUNCTIONAL DESCRIPTION

A block diagram of the SPI is shown in Figure 16. In a master configuration, the CPU sends a signal to the master start logic, which originates an SPI clock (SCK) based on the internal processor clock. As a master device, data is parallel loaded into the 8-bit shift register from the internal bus during a write cycle and then serially shifted via the MOSI pin to the slave devices. During a read cycle, data is applied serially from a slave device via the MISO pin to the 8-bit shift register. Data is then parallel transferred to the read buffer and made available to the internal data bus during a CPU read cycle.

In a slave configuration, the slave start logic receives a logic low at the $\overline{SS}$ pin and a clock input at the SCK pin. This synchronizes the slave with the master. Data from the master is received serially at the slave MOSI pin and shifted into the 8-bit shift register for a parallel transfer to the read buffer. During a write cycle, data is parallel loaded into the 8-bit shift register from the internal data

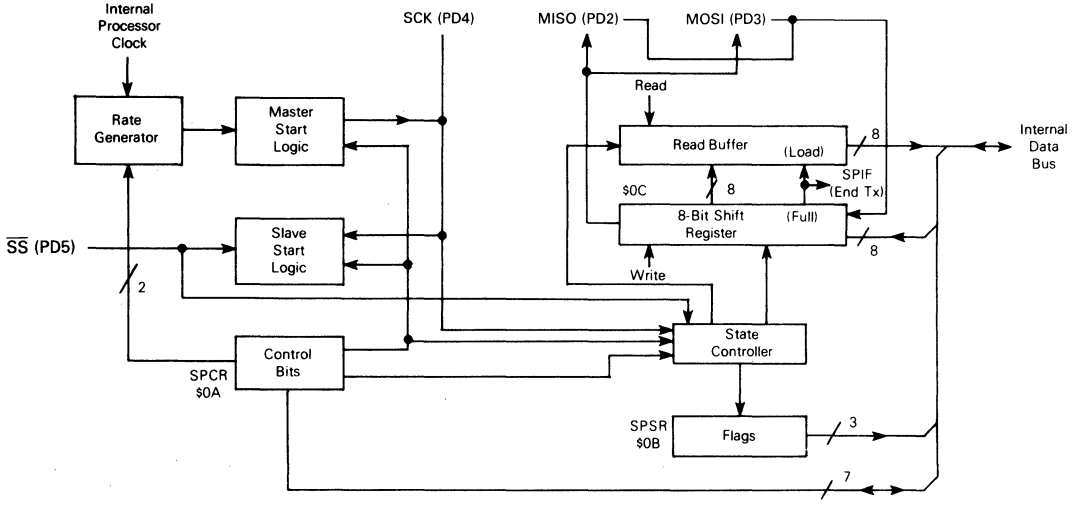


Figure 16. SPI Block Diagram

bus, awaiting the clocks from the master to shift out serially to the MISO pin and then to the master device.

Figure 17 illustrates the MOSI, MISO, SCK, and $\overline{SS}$ master-slave interconnections.

REGISTERS

There are three registers in the SPI that provide control, status, and data storage functions. These registers, the serial peripheral control register (SPCR), serial peripheral status register (SPSR), and serial peripheral data I/O register (SPDR), are described in the following paragraphs.

Serial Peripheral Control Register $0A

The SPCR provides control of individual SPI functions such as interrupt and system enabling/disabling, master/slave mode select, and clock polarity/phase/rate select.

7	6	5	4	3	2	1	0
SPIE	SPE	—	MSTR	CPOL	CPHA	SPR1	SPR0
RESET:							
0	0	—	0	U	U	U	U

SPIE — Serial Peripheral Interrupt Enable
1 = SPI interrupt enabled
0 = SPI interrupt disabled

SPE — Serial Peripheral System Enable
1 = SPI system on
0 = SPI system off

MSTR — Master Mode Select
1 = Master mode
0 = Slave mode

CPOL — Clock Polarity
Clock polarity bit controls the clock value and is used in conjunction with the clock phase (CPHA) bit.
1 = SCK line idles high
0 = SCK line idles in low state

CPHA — Clock Phase
Clock phase bit along with CPOL controls the clock-data relationship between the master and slave devices. CPOL selects one of two clocking protocols.
1 = $\overline{SS}$ is an output enable control.
0 = Shift clock is the OR of SCK with $\overline{SS}$.
When $\overline{SS}$ is low, first edge of SCK invokes first data sample.

SPR0, SPR1 — SPI Clock Rate Bits
Two clock rate bits are used to select one of four clock rates to be used as SCK in the master mode. In the slave mode, the two clock rate bits have no effect. Clock rate selection is shown in the following table.

Bit 5 – Not used
Can read either one or zero

SPI Clock Rate Selection

SPR1	SPR0	Internal Processor Clock Divided By
0	0	2
0	1	4
1	0	16
1	1	32

Serial Peripheral Status Register $0B

The SPSR contains three status bits.

7	6	5	4	3	2	1	0
SPIF	WCOL	—	MODF	—	—	—	—
RESET:							
0	0	—	0	—	—	—	—

SPIF — Serial Peripheral Data Transfer Flag
1 = Indicates data transfer completed between processor and external device.
(If SPIF = 1 and SPIE = 1, SPI interrupt is enabled.)
0 = Clearing is accomplished by reading SPSR (with SPIF = 1) followed by SPDR access.

WCOL — Write Collision
1 = Indicates an attempt is made to write to SPDR while data transfer is in process.
0 = Clearing is accomplished by reading SPSR (with WCOL = 1), followed by SPDR access.

MODF — Mode Fault Flag
1 = Indicates multi-master system control conflict.
0 = Clearing is accomplished by reading SPSR (with MODF = 1), followed by a write to the SPCR.

Bits 0–3, and 5 — Not used
Can read either zero or one

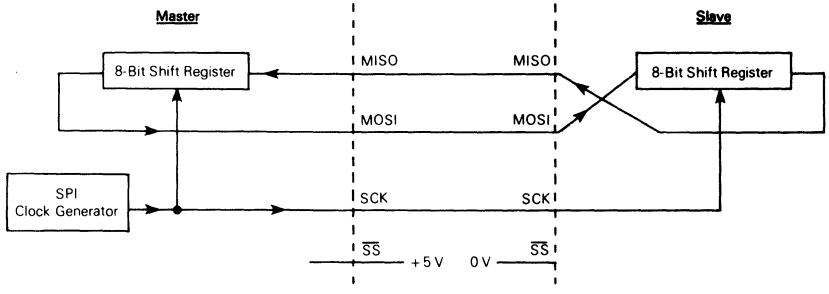


Figure 17. SPI Master-Slave Interconnections

Serial Peripheral Data I/O Register $0C

The SPDR is a read/write register used to receive and transmit SPI data.

7	6	5	4	3	2	1	0
SPD7	SPD6	SPD5	SPD4	SPD3	SPD2	SPD1	SPD0
RESET: U	U	U	U	U	U	U	U

A write to the SPDR places data directly into the shift register for transmission. Only a write to this register will initiate transmission/reception of another byte and will only occur in the master device. On completion of byte transmission, the SPIF status bit is set in both master and slave devices.

A read to the SPDR causes the buffer to be read. The first SPIF status bit must be cleared by the time a second data transfer from the shift register to the read buffer begins, or an overrun condition will exist. In overrun cases, the byte causing the overrun is lost.

3

INSTRUCTION SET

The MCU has a set of 62 basic instructions. They can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

This MCU uses all the instructions available in the M146805 CMOS Family plus one more: the unsigned multiply (MUL) instruction. This instruction allows unsigned multiplication of the contents of the accumulator (A) and the index register (X). The high-order product is then stored in the index register, and the low-order product is stored in the accumulator. A detailed definition of the MUL instruction is shown below.

Operation	X:A X*A			
Description	Multiplies the eight bits in the index register by the eight bits in the accumulator to obtain a 16-bit unsigned number in the concatenated accumulator and index register			
Condition Codes	H: Cleared I: Not affected N: Not affected Z: Not affected C: Cleared			
Source	MUL			
Form(s)	Addressing Mode Inherent	Cycles 11	Bytes 1	Opcode $42

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following instruction list.

Function	Mnemonic
Load A from Memory	LDA
Load X from Memory	LDX
Store A in Memory	STA
Store X in Memory	STX
Add Memory to A	ADD
Add Memory and Carry to A	ADC
Subtract Memory	SUB
Subtract Memory from A with Borrow	SBC
AND Memory to A	AND
OR Memory with A	ORA
Exclusive OR Memory with A	EOR
Arithmetic Compare A with Memory	CMP
Arithmetic Compare X with Memory	CPX
Bit Test Memory with A (Logical Compare)	BIT
Jump Unconditional	JMP
Jump to Subroutine	JSR

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to the following list of instructions.

Function	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (Twos Complement)	NEG
Rotate Left Thru Carry	ROL
Rotate Right Thru Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST
Multiply	MUL

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two-byte instructions. Refer to the following list for branch instructions.

Function	Mnemonic
Branch Always	BRA
Branch Never	BRN

— Continued —

Function	Mnemonic
Branch if Higher	BHI
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
Branch if Higher or Same	BHS
Branch if Carry Set	BCS
Branch if Lower	BLO
Branch if Not Equal	BNE
Branch if Equal	BEQ
Branch if Half Carry Clear	BHCC
Branch if Half Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Bit is Clear	BMC
Branch if Interrupt Mask Bit is Set	BMS
Branch if Interrupt Line is Low	BIL
Branch if Interrupt Line is High	BIH
Branch to Subroutine	BSR

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any writable bit which resides in the first 256 bytes of the memory space where all port registers, port DDRs, timer, timer control, ROM, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear and bit test, and branch functions are all implemented with a single instruction. For test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. Refer to the following list for bit manipulation instructions.

Function	Mnemonic
Branch if Bit n is Set	BRSET n (n=0 . . . 7)
Branch if Bit n is Clear	BRCLR n (n=0 . . . 7)
Set Bit n	BSET n (n=0 . . . 7)
Clear Bit n	BCLR n (n=0 . . . 7)

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

Function	Mnemonic
Transfer A to X	TAX
Transfer X to A	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask Bit	SEI

— Continued —

Function	Mnemonic
Clear Interrupt Mask Bit	CLI
Software Interrupt	SWI
Return from Subroutine	RTS
Return from Interrupt	RTI
Reset Stack Pointer	RSP
No-Operation	NOP
Stop	STOP
Wait	WAIT

OPCODE MAP SUMMARY

Table 5 is an opcode map for the instructions used on the MCU.

3

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single byte instructions; the longest instructions (three bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. One- or two-byte direct addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (e.g., a constant used to initialize a loop counter).

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single three-byte instruction. When using the Motorola assembler, the user need not specify whether

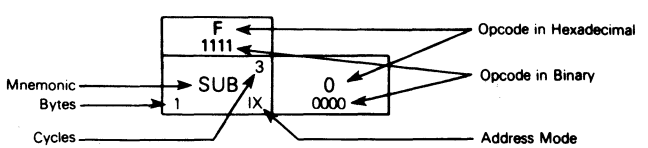
Table 5. Opcode Map

Hi / Low	Bit Manipulation BTB 0 0000	Bit Manipulation BSC 1 0001	Branch REL 2 0010	Read/Modify/Write DIR 3 0011	Read/Modify/Write INH 4 0100	Read/Modify/Write INH 5 0101	Read/Modify/Write IX1 6 0110	Read/Modify/Write IX 7 0111	Control INH 8 1000	Control INH 9 1001	Register/Memory IMM A 1010	Register/Memory DIR B 1011	Register/Memory EXT C 1100	Register/Memory IX2 D 1101	Register/Memory IX1 E 1110	Register/Memory IX F 1111	Hi / Low
0 0000	BRSET0 5 3 BTB	BSET0 5 2 BSC	BRA 3 2 REL	NEG 5 2 DIR	NEGA 3 1 INH	NEGX 3 1 INH	NEG 6 2 IX1	NEG 5 1 IX	RTI 9 1 INH		SUB 2 2 IMM	SUB 3 2 DIR	SUB 4 3 EXT	SUB 5 3 IX2	SUB 4 2 IX1	SUB 3 1 IX	0 0000
1 0001	BRCLR0 5 3 BTB	BCLR0 5 2 BSC	BRN 3 2 REL						RTS 6 1 INH		CMP 2 2 IMM	CMP 3 2 DIR	CMP 4 3 EXT	CMP 5 3 IX2	CMP 4 2 IX1	CMP 3 1 IX	1 0001
2 0010	BRSET1 5 3 BTB	BSET1 5 2 BSC	BHI 3 2 REL		MUL 11 1 INH						SBC 2 2 IMM	SBC 3 2 DIR	SBC 4 3 EXT	SBC 5 3 IX2	SBC 4 2 IX1	SBC 3 1 IX	2 0010
3 0011	BRCLR1 5 3 BTB	BCLR1 5 2 BSC	BLS 3 2 REL	COM 5 2 DIR	COMA 3 1 INH	COMX 3 1 INH	COM 6 2 IX1	COM 5 1 IX	SWI 10 1 INH		CPX 2 2 IMM	CPX 3 2 DIR	CPX 4 3 EXT	CPX 5 3 IX2	CPX 4 2 IX1	CPX 3 1 IX	3 0011
4 0100	BRSET2 5 3 BTB	BSET2 5 2 BSC	BCC 3 2 REL	LSR 5 2 DTR	LSRA 3 1 INH	LSRX 3 1 INH	LSR 6 2 IX1	LSR 5 1 IX			AND 2 2 IMM	AND 3 2 DIR	AND 4 3 EXT	AND 5 3 IX2	AND 4 2 IX1	AND 3 1 IX	4 0100
5 0101	BRCLR2 5 3 BTB	BCLR2 5 2 BSC	BCS 3 2 REL								BIT 2 2 IMM	BIT 3 2 DIR	BIT 4 3 EXT	BIT 5 3 IX2	BIT 4 2 IX1	BIT 3 1 IX	5 0101
6 0110	BRSET3 5 3 BTB	BSET3 5 2 BSC	BNE 3 2 REL	ROR 5 2 DIR	RORA 3 1 INH	RORX 3 1 INH	ROR 6 2 IX1	ROR 5 1 IX			LDA 2 2 IMM	LDA 3 2 DIR	LDA 4 3 EXT	LDA 5 3 IX2	LDA 4 2 IX1	LDA 3 1 IX	6 0110
7 0111	BRCLR3 5 3 BTB	BCLR3 5 2 BSC	BEQ 3 2 REL	ASR 5 2 DIR	ASRA 3 1 INH	ASRX 3 1 INH	ASR 6 2 IX1	ASR 5 1 IX		TAX 2 1 INH		STA 4 2 DIR	STA 5 3 EXT	STA 6 3 IX2	STA 5 2 IX1	STA 4 1 IX	7 0111
8 1000	BRSET4 5 3 BTB	BSET4 5 2 BSC	BHCC 3 2 REL	LSL 5 2 DIR	LSLA 3 1 INH	LSLX 3 1 INH	LSL 6 2 IX1	LSL 5 1 IX		CLC 2 1 INH	EOR 2 2 IMM	EOR 3 2 DIR	EOR 4 3 EXT	EOR 5 3 IX2	EOR 4 2 IX1	EOR 3 1 IX	8 1000
9 1001	BRCLR4 5 3 BTB	BCLR4 5 2 BSC	BHCS 3 2 REL	ROL 5 2 DIR	ROLA 3 1 INH	ROLX 3 1 INH	ROL 6 2 IX1	ROL 5 1 IX		SEC 2 1 INH	ADC 2 2 IMM	ADC 3 2 DIR	ADC 4 3 EXT	ADC 5 3 IX2	ADC 4 2 IX1	ADC 3 1 IX	9 1001
A 1010	BRSET5 5 3 BTB	BSET5 5 2 BSC	BPL 3 2 REL	DEC 5 2 DIR	DECA 3 1 INH	DECX 3 1 INH	DEC 6 2 IX1	DEC 5 1 IX		CLI 2 1 INH	ORA 2 2 IMM	ORA 3 2 DIR	ORA 4 3 EXT	ORA 5 3 IX2	ORA 4 2 IX1	ORA 3 1 IX	A 1010
B 1011	BRCLR5 5 3 BTB	BCLR5 5 2 BSC	BMI 3 2 REL							SEI 2 1 INH	ADD 2 2 IMM	ADD 3 2 DIR	ADD 4 3 EXT	ADD 5 3 IX2	ADD 4 2 IX1	ADD 3 1 IX	B 1011
C 1100	BRSET6 5 3 BTB	BSET6 5 2 BSC	BMC 3 2 REL	INC 5 2 DIR	INCA 3 1 INH	INCX 3 1 INH	INC 6 2 IX1	INC 5 1 IX		RSP 2 1 INH		JMP 2 2 DIR	JMP 3 3 EXT	JMP 4 3 IX2	JMP 3 2 IX1	JMP 2 1 IX	C 1100
D 1101	BRCLR6 5 3 BTB	BCLR6 5 2 BSC	BMS 3 2 REL	TST 4 2 DIR	TSTA 3 1 INH	TSTX 3 1 INH	TST 5 2 IX1	TST 4 1 IX		NOP 2 1 INH	BSR 6 2 REL	JSR 5 2 DIR	JSR 6 3 EXT	JSR 7 3 IX2	JSR 6 2 IX1	JSR 5 1 IX	D 1101
E 1110	BRSET7 5 3 BTB	BSET7 5 2 BSC	BIL 3 2 REL						STOP 2 1 INH		LDX 2 2 IMM	LDX 3 2 DIR	LDX 4 3 EXT	LDX 5 3 IX2	LDX 4 2 IX1	LDX 3 1 IX	E 1110
F 1111	BRCLR7 5 3 BTB	BCLR7 5 2 BSC	BIH 3 2 REL	CLR 5 2 DIR	CLRA 3 1 INH	CLRX 3 1 INH	CLR 6 2 IX1	CLR 5 1 IX	WAIT 2 1 INH	TXA 2 1 INH		STX 4 2 DIR	STX 5 3 EXT	STX 6 3 IX2	STX 5 2 IX1	STX 4 1 IX	F 1111

Abbreviations for Address Modes

INH	Inherent
A	Accumulator
X	Index Register
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



an instruction uses direct or extended addressing. The assembler automatically selects the shortest form of the instruction.

RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −126 to +129 from the opcode address. The programmer need not calculate the offset when using the Motorola assembler, since it calculates the proper offset and checks to see that it is within the span of the branch.

INDEXED, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. This addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode. The addressing mode is useful for selecting the Kth element in an n element table. With this two-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE is the last location at which the instruction may begin).

INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This address mode can be used in a manner similar to indexed, 8-bit offset except that this three-byte instruction allows tables to be anywhere in memory. As with direct and extended addressing, the Motorola assembler determines the shortest form of indexed addressing.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode, and the byte following the opcode specifies the direct addressing of the byte in which the specified bit is to be set or cleared. Any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single two-byte instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit that is to be tested and its condition (set or clear), is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code register.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator as well as the control instruction with no other arguments are included in this mode. These instructions are one byte long.

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS (Voltages referenced to V_{SS})

Rating	Symbol	Value	Unit
Supply Voltage	V_{DD}	−0.3 to +7.0	V
Input Voltage	V_{in}	V_{SS} −0.3 to V_{DD} +0.3	V
Self-Check Mode ($\overline{IRQ}$ Pin Only)	V_{in}	V_{SS} −0.3 to 2×V_{DD}+0.3	V
Current Drain Per Pin Excluding V_{DD} and V_{SS}	I	25	mA
Operating Temperature Range MC68HC05C8P, FN MC68HC05C8CP, CFN MC68HC05C8VP, VFN MC68HC05C8MP, MFN	T_A	T_L to T_H 0 to +70 −40 to +85 −40 to +105 −40 to +125	°C
Storage Temperature Range	T_{stg}	−65 to +150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. For proper operation, it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{DD}$. Reliability of operation is enhanced if unused inputs are connected to an appropriate logic voltage level (e.g., either V_{SS} or V_{DD}).

3

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic Plastic Leaded Chip Carrier (PLCC)	θ_{JA}	 60 70	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{I/O}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
$P_{I/O}$ = Power Dissipation on Input and Output Pins — User Determined

For most applications $P_{I/O} < P_{INT}$ and can be neglected.

The following is an approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected):

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

V_{DD} = 4.5 V

Pins	R1	R2	C
PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4	3.26 kΩ	2.38 kΩ	50 pF
PD0, PD5, PD7	1.9 kΩ	2.26 kΩ	200 pF

V_{DD} = 3.0 V

Pins	R1	R2	C
PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4	10.91 kΩ 6.32 kΩ	50 pF	
PD0, PD5, PD7	6 kΩ	6 kΩ	200 pF

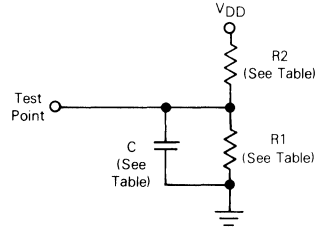


Figure 18. Equivalent Test Load

DC ELECTRICAL CHARACTERISTICS

(V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage, $I_{Load} \leq 10.0\ \mu A$	V_{OL} V_{OH}	— $V_{DD}-0.1$	— —	0.1 —	V
Output High Voltage (I_{Load} = 0.8 mA) PA0-PA7, PB0-PB7, PC0-PC7, TCMP (see Figure 19) (I_{Load} = 1.6 mA) PD1-PD4 (see Figure 20)	V_{OH}	 $V_{DD}-0.8$ $V_{DD}-0.8$	 — —	 — —	V
Output Low Voltage (see Figure 21) (I_{Load} = 1.6 mA) PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4, TCMP	V_{OL}	—	—	0.4	V
Input High Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IH}	$0.7 \times V_{DD}$	—	V_{DD}	V
Input Low Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IL}	V_{SS}	—	$0.2 \times V_{DD}$	V
Data Retention Mode (0° to 70°C)	V_{RM}	2.0	—	—	V
Supply Current (see Notes) Run (see Figures 22 and 23) Wait (see Figures 22 and 23) Stop (see Figure 23) 25°C 0° to 70°C (Standard) −40° to +85°C −40° to +125°C	I_{DD}	 — — — — — —	 3.5 1.6 2.0 — — —	 7.0 4.0 50 140 180 250	 mA mA µA µA µA µA
I/O Ports Hi-Z Leakage Current PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4	I_{IL}	—	—	±10	µA
Input Current $\overline{RESET}$, $\overline{IRQ}$, TCAP, OSC1, PD0, PD5, PD7	I_{in}	—	—	±1	µA
Capacitance Ports (as Input or Output) $\overline{RESET}$, $\overline{IRQ}$, TCAP, PD0-PD5, PD7	 C_{out} C_{in}	 — —	 — —	 12 8	pF

NOTES:
1. All values shown reflect average measurements.
2. Typical values at midpoint of voltage range, 25°C only.
3. Wait I_{DD}: Only timer system active (SPE = TE = RE = 0). If SPI, SCI active (SPE = TE = RE = 1) add 10% current draw.
4. Run (Operating) I_{DD}, Wait I_{DD}: Measured using external square wave clock source (f_{osc} = 4.2 MHz), all inputs 0.2 V from rail; no dc loads, less than 50 pF on all outputs, C_L = 20 pF on OSC2.
5. Wait, Stop I_{DD}: All ports configured as inputs, V_{IL} = 0.2 V, $V_{IH} = V_{DD} - 0.2$ V.
6. Stop I_{DD} measured with OSC1 = V_{SS}.
7. Standard temperature range is 0° to 70°C. Extended temperature (−40° to +85°C, −40° to +125°C) versions and a 25°C only version are available.
8. Wait I_{DD} is affected linearly by the OSC2 capacitance.

DC ELECTRICAL CHARACTERISTICS

(V_{DD} = 3.3 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage, $I_{Load} \leq 10.0\ \mu A$	V_{OL} V_{OH}	— $V_{DD}-0.1$	— —	0.1 —	V
Output High Voltage (I_{Load} = 0.2 mA) PA0-PA7, PB0-PB7, PC0-PC7, TCMP (see Figure 19) (I_{Load} = 0.4 mA) PD1-PD4 (see Figure 20)	V_{OH}	 $V_{DD}-0.3$ $V_{DD}-0.3$	 — —	 — —	V
Output Low Voltage (see Figure 21) (I_{Load} = 0.4 mA) PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4, TCMP	V_{OL}	—	—	0.3	V
Input High Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, RESET, OSC1	V_{IH}	$0.7 \times V_{DD}$	—	V_{DD}	V
Input Low Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, RESET, OSC1	V_{IL}	V_{SS}	—	$0.2 \times V_{DD}$	V
Data Retention Mode (0° to 70°C)	V_{RM}	2.0	—	—	V
Supply Current (see Notes) Run (see Figures 22 and 24) Wait (see Figures 22 and 24) Stop (see Figure 24) 25°C 0° to 70°C (Standard) −40° to +85°C −40° to +125°C	I_{DD}	 — — — — — —	 1.0 0.5 1.0 — — —	 2.5 1.4 30 80 120 175	 mA mA µA µA µA µA
I/O Ports Hi-Z Leakage Current PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4	I_{IL}	—	—	±10	µA
Input Current RESET, IRQ, TCAP, OSC1, PD0, PD5, PD7	I_{in}	—	—	±1	µA
Capacitance Ports (as Input or Output) RESET, IRQ, TCAP, PD0-PD5, PD7	 C_{out} C_{in}	 — —	 — —	 12 8	pF

NOTES:
1. All values shown reflect average measurements.
2. Typical values at midpoint of voltage range, 25°C only.
3. Wait I_{DD}: Only timer system active (SPE = TE = RE = 0). If SPI, SCI active (SPE = TE = RE = 1) add 10% current draw.
4. Run (Operating) I_{DD}, Wait I_{DD}: Measured using external square wave clock source (f_{osc} = 4.2 MHz), all inputs 0.2 V from rail; no dc loads, less than 50 pF on all outputs, C_L = 20 pF on OSC2.
5. Wait, Stop I_{DD}: All ports configured as inputs, V_{IL} = 0.2 V, $V_{IH} = V_{DD} - 0.2$ V.
6. Stop I_{DD} measured with OSC1 = V_{SS}.
7. Standard temperature range is 0° to 70°C. Extended temperature (−40° to +85°C, −40° to +125°C) versions and a 25°C only version are available.
8. Wait I_{DD} is affected linearly by the OSC2 capacitance.

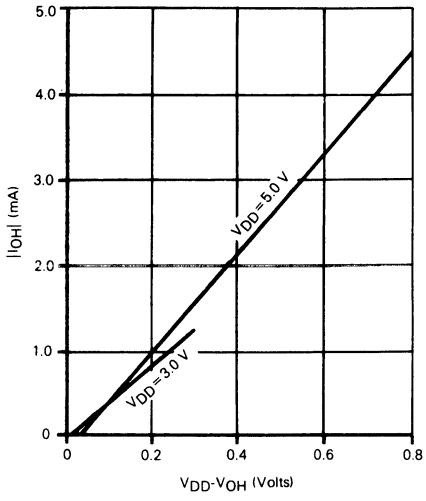


Figure 19. Typical V_{OH} vs I_{OH} for Ports A, B, C, and TCMP

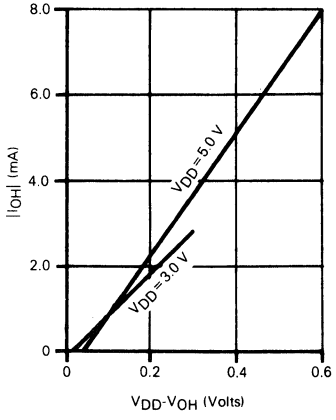


Figure 20. Typical V_{OH} vs I_{OH} for PD1-PD4

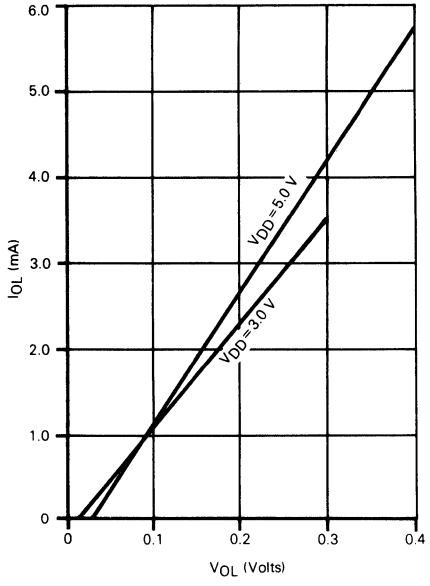


Figure 21. Typical V_{OL} vs I_{OL} for All Ports

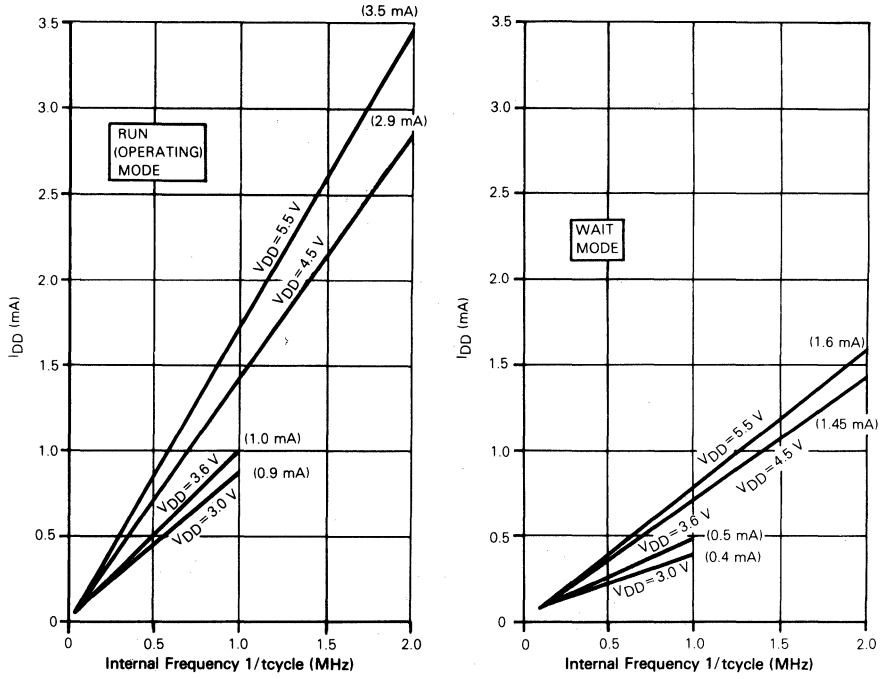


Figure 22. Typical Current vs Internal Frequency for Run and Wait Modes

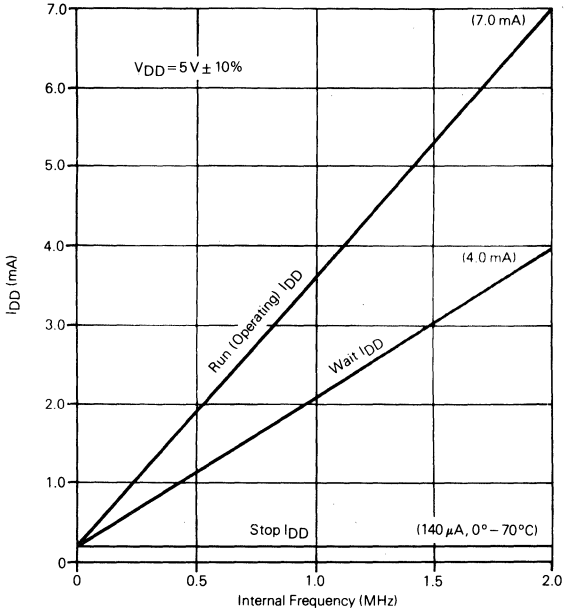


Figure 23. Maximum I_{DD} vs Frequency for V_{DD} = 5.0 Vdc

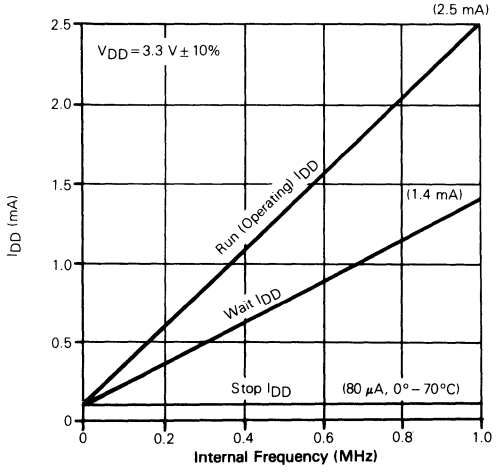


Figure 24. Maximum I_{DD} vs Frequency for V_{DD} = 3.3 Vdc

3

CONTROL TIMING

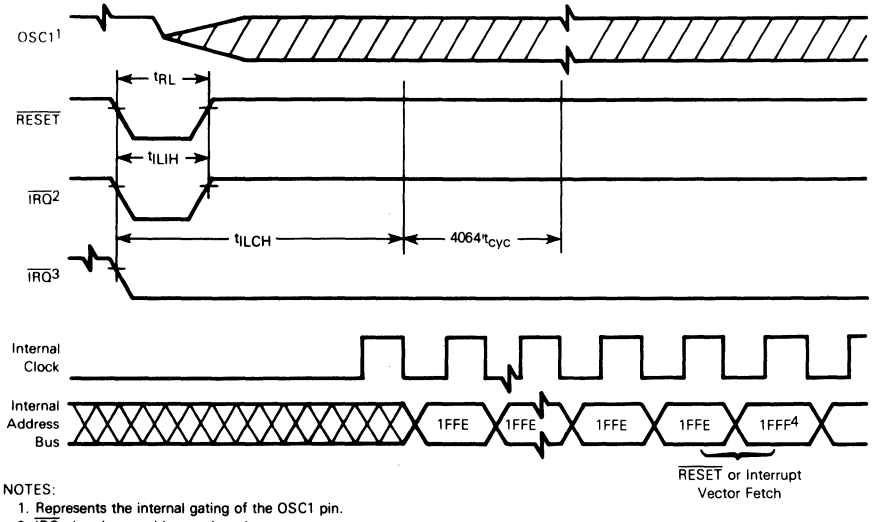
(V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation Crystal Option External Clock Option	f_{osc}	 — dc	 4.2 4.2	MHz
Internal Operating Frequency Crystal ($f_{osc} \div 2$) External Clock ($f_{osc} \div 2$)	f_{op}	 — dc	 2.1 2.1	MHz
Cycle Time (see Figure 28)	t_{cyc}	480	—	ns
Crystal Oscillator Startup Time (see Figure 28)	t_{OXOV}	—	100	ms
Stop Recovery Startup Time (Crystal Oscillator) (see Figure 25)	t_{ILCH}	—	100	ms
$\overline{\text{RESET}}$ Pulse Width (see Figure 28)	t_{RL}	1.5	—	t_{cyc}
Timer Resolution** Input Capture Pulse Width (see Figure 26) Input Capture Pulse Period (see Figure 26)	 t_{RESL} t_{TH}, t_{TL} t_{TLTL}	 4.0 125 ***	 — — —	 t_{cyc} ns t_{cyc}
Interrupt Pulse Width Low (Edge-Triggered) (see Figure 8)	t_{ILIH}	125	—	ns
Interrupt Pulse Period (see Figure 8)	t_{ILIL}	*	—	t_{cyc}
OSC1 Pulse Width	t_{OH}, t_{OL}	90	—	ns

*The minimum period t_{ILIL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus 21 t_{cyc}.

**Since a 2-bit prescaler in the timer must count four internal cycles (t_{cyc}), this is the limiting minimum factor in determining the timer resolution.

***The minimum period t_{TLTL} should not be less than the number of cycle times it takes to execute the capture interrupt service routine plus 24 t_{cyc}.



NOTES:
1. Represents the internal gating of the OSC1 pin.
2. $\overline{\text{IRQ}}$ pin edge-sensitive mask option.
3. $\overline{\text{IRQ}}$ pin level and edge-sensitive mask option.
4. $\overline{\text{RESET}}$ vector address shown for timing example.

Figure 25. Stop Recovery Timing Diagram

CONTROL TIMING

(V_{DD} = 3.3 Vdc ± 0.3 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation Crystal Option External Clock Option	f_{osc}	 — dc	 2.0 2.0	MHz
Internal Operating Frequency Crystal ($f_{osc} \div 2$) External Clock ($f_{osc} \div 2$)	f_{op}	 — dc	 1.0 1.0	MHz
Cycle Time (see Figure 28)	t_{cyc}	1000	—	ns
Crystal Oscillator Startup Time (see Figure 28)	t_{OXOV}	—	100	ms
Stop Recovery Startup Time (Crystal Oscillator) (see Figure 25)	t_{ILCH}	—	100	ms
$\overline{\text{RESET}}$ Pulse Width — Excluding Power-Up (see Figure 28)	t_{RL}	1.5	—	t_{cyc}
Timer Resolution** Input Capture Pulse Width (see Figure 26) Input Capture Pulse Period (see Figure 26)	 t_{RESL} t_{TH}, t_{TL} t_{TLTL}	 4.0 250 ***	 — — —	 t_{cyc} ns t_{cyc}
Interrupt Pulse Width Low (Edge-Triggered) (see Figure 8)	t_{ILIH}	250	—	ns
Interrupt Pulse Period (see Figure 8)	t_{ILIL}	*	—	t_{cyc}
OSC1 Pulse Width	t_{OH}, t_{OL}	200	—	ns

*The minimum period t_{ILIL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus 21 t_{cyc}.

**Since a 2-bit prescaler in the timer must count four internal cycles (t_{cyc}), this is the limiting minimum factor in determining the timer resolution.

***The minimum period t_{TLTL} should not be less than the number of cycle times it takes to execute the capture interrupt service routine plus 24 t_{cyc}.

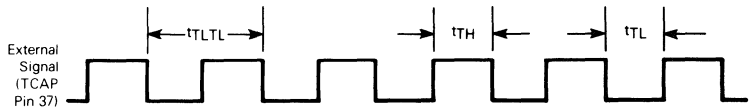


Figure 26. Timer Relationships

SERIAL PERIPHERAL INTERFACE (SPI) TIMING

(V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H) (see Figure 27)

Num.	Characteristic	Symbol	Min	Max	Unit
	Operating Frequency Master Slave	 $f_{op(m)}$ $f_{op(s)}$	 dc dc	 0.5 2.1	 f_{op} MHz
1	Cycle Time Master Slave	 $t_{cyc(m)}$ $t_{cyc(s)}$	 2.0 480	 — —	 t_{cyc} ns
2	Enable Lead Time Master Slave	 $t_{lead(m)}$ $t_{lead(s)}$	 * 240	 — —	 ns ns
3	Enable Lag Time Master Slave	 $t_{lag(m)}$ $t_{lag(s)}$	 * 240	 — —	 ns ns
4	Clock (SCK) High Time Master Slave	 $t_{w(SCKH)m}$ $t_{w(SCKH)s}$	 340 190	 — —	 ns ns
5	Clock (SCK) Low Time Master Slave	 $t_{w(SCKL)m}$ $t_{w(SCKL)s}$	 340 190	 — —	 ns ns
6	Data Setup Time (Inputs) Master Slave	 $t_{su(m)}$ $t_{su(s)}$	 100 100	 — —	 ns ns
7	Data Hold Time (Inputs) Master Slave	 $t_{h(m)}$ $t_{h(s)}$	 100 100	 — —	 ns ns
8	Access Time (Time to Data Active from High-Impedance State) Slave	 t_a	 0	 120	 ns
9	Disable Time (Hold Time to High-Impedance State) Slave	 t_{dis}	 —	 240	 ns
10	Data Valid Master (Before Capture Edge) Slave (After Enable Edge)**	 $t_{v(m)}$ $t_{v(s)}$	 0.25 —	 — 240	 $t_{cyc(m)}$ ns
11	Data Hold Time (Outputs) Master (After Capture Edge) Slave (After Enable Edge)	 $t_{ho(m)}$ $t_{ho(s)}$	 0.25 0	 — —	 $t_{cyc(m)}$ ns
12	Rise Time (20% V_{DD} to 70% V_{DD}, C_L = 200 pF) SPI Outputs (SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and SS)	 t_{rm} t_{rs}	 — —	 100 2.0	 ns µs
13	Fall Time (70% V_{DD} to 20% V_{DD}, C_L = 200 pF) SPI Outputs (SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and SS)	 t_{fm} t_{fs}	 — —	 100 2.0	 ns µs

*Signal production depends on software.
**Assumes 200 pF load on all SPI pins.

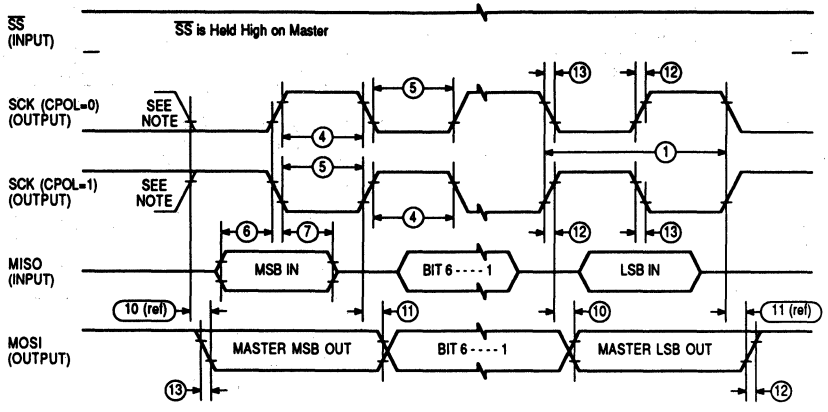
SERIAL PERIPHERAL INTERFACE (SPI) TIMING

(V_{DD} = 3.3 Vdc ± 0.3 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H) (see Figure 27)

Num.	Characteristic	Symbol	Min	Max	Unit
	Operating Frequency Master Slave	 $f_{op(m)}$ $f_{op(s)}$	 dc dc	 0.5 1.0	 f_{op} MHz
1	Cycle Time Master Slave	 $t_{cyc(m)}$ $t_{cyc(s)}$	 2.0 1.0	 — —	 t_{cyc} µs
2	Enable Lead Time Master Slave	 $t_{lead(m)}$ $t_{lead(s)}$	 * 500	 — —	 ns ns
3	Enable Lag Time Master Slave	 $t_{lag(m)}$ $t_{lag(s)}$	 * 500	 — —	 ns ns
4	Clock (SCK) High Time Master Slave	 $t_{w(SCKH)m}$ $t_{w(SCKH)s}$	 720 400	 — —	 µs ns
5	Clock (SCK) Low Time Master Slave	 $t_{w(SCKL)m}$ $t_{w(SCKL)s}$	 720 400	 — —	 µs ns
6	Data Setup Time (Inputs) Master Slave	 $t_{su(m)}$ $t_{su(s)}$	 200 200	 — —	 ns ns
7	Data Hold Time (Inputs) Master Slave	 $t_{h(m)}$ $t_{h(s)}$	 200 200	 — —	 ns ns
8	Access Time (Time to Data Active from High-Impedance State) Slave	 t_a	 0	 250	 ns
9	Disable Time (Hold Time to High-Impedance State) Slave	 t_{dis}	 —	 500	 ns
10	Data Valid Master (Before Capture Edge) Slave (After Enable Edge)**	 $t_{v(m)}$ $t_{v(s)}$	 0.25 —	 — 500	 $t_{cyc(m)}$ ns
11	Data Hold Time (Outputs) Master (After Capture Edge) Slave (After Enable Edge)	 $t_{ho(m)}$ $t_{ho(s)}$	 0.25 0	 — —	 $t_{cyc(m)}$ ns
12	Rise Time (20% V_{DD} to 70% V_{DD}, C_L = 200 pF) SPI Outputs (SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and SS)	 t_{rm} t_{rs}	 — —	 200 2.0	 ns µs
13	Fall Time (70% V_{DD} to 20% V_{DD}, C_L = 200 pF) SPI Outputs (SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and SS)	 t_{fm} t_{fs}	 — —	 200 2.0	 ns µs

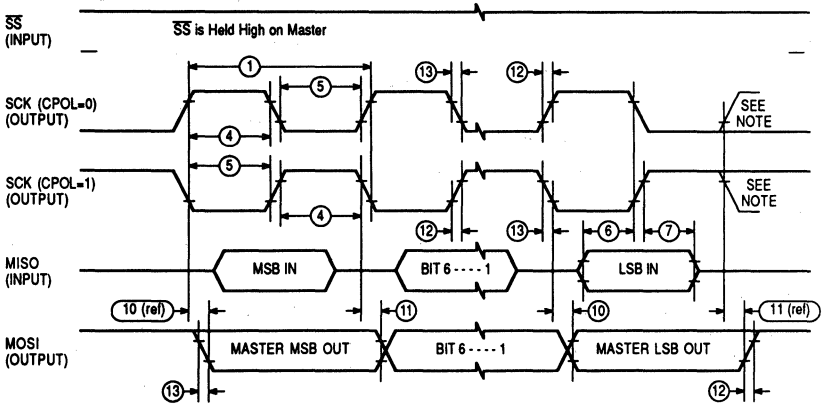
*Signal production depends on software.
**Assumes 200 pF load on all SPI pins.

3



NOTE: This first clock edge is generated internally but is not seen at the SCK pin.

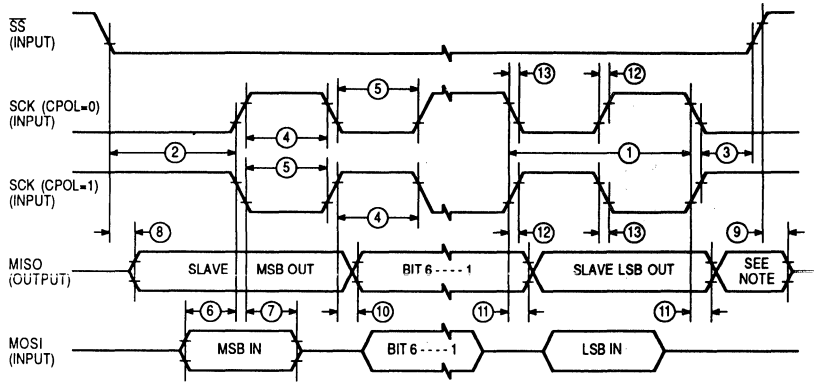
a) SPI MASTER TIMING (CPHA = 0)



NOTE: This last clock edge is generated internally but is not seen at the SCK pin.

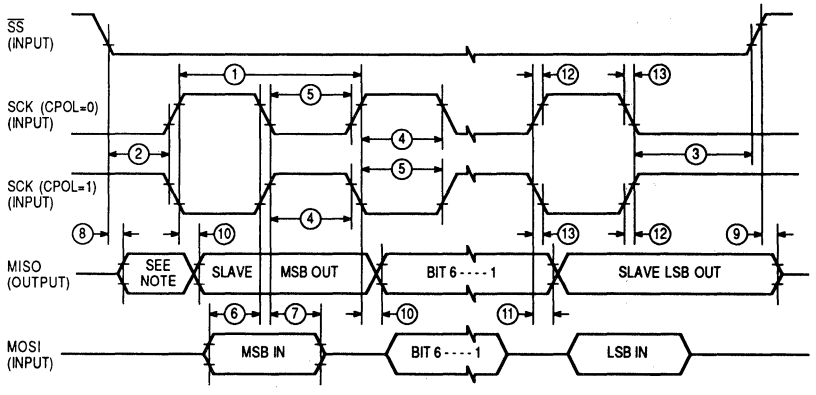
b) SPI MASTER TIMING (CPHA = 1)

Figure 27. SPI Timing Diagrams (Sheet 1 of 2)



NOTE: Not defined but normally MSB of character just received.

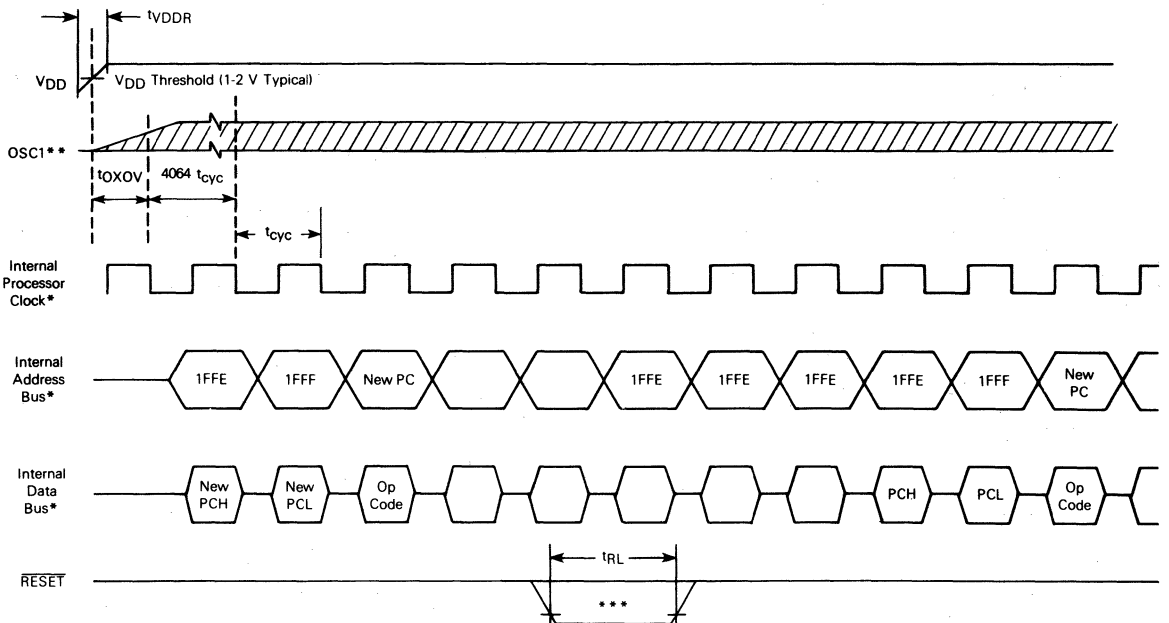
c) SPI SLAVE TIMING (CPHA = 0)



NOTE: Not defined but normally LSB of character previously transmitted.

d) SPI SLAVE TIMING (CPHA = 1)

Figure 27. SPI Timing Diagrams (Sheet 2 of 2)



*Internal timing signal and bus information not available externally.
**OSC1 line is not meant to represent frequency. It is only used to represent time.
***The next rising edge of the internal processor clock following the rising edge of $\overline{\text{RESET}}$ initiates the reset sequence.

Figure 28. Power-On Reset and $\overline{\text{RESET}}$

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola in the following media:

MDOS™, disk file
MS™-DOS/PC-DOS disk file (360K)
EPROM(s) 2764, MCM68764, MCM68766, or EEPROM MC68HC805C4

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, a sales person, or a Motorola representative.

FLEXIBLE DISKS

A flexible disk (MS-DOS/PC-DOS disk file), programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. The diskette should be clearly labeled with the customer's name, data, project or product name, and the name of the file containing the pattern.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to expedite the process in case of any difficulty with the pattern file.

MS-DOS/PC-DOS Disk File

MS-DOS is Microsoft's Disk Operating System. PC-DOS is the IBM® Personal Computer (PC) Disk Operating System. Disk media submitted must be a standard density (360K) double-sided 5 1/4 inch compatible floppy diskette. The diskette must contain object file code in Motorola's S-record format. The S-record format is a character-based object file format generated by M6805 cross assemblers and linkers on IBM PC style machines.

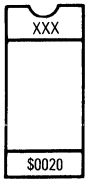
EPROMs

A 2764, 68764, or 68766 type EPROM, programmed with the customer's program (positive logic sense for address and data), may be submitted for pattern generation. Since all program and data space information will fit on one 68766 EPROM device, the EPROM must be programmed as described in the following paragraph.

Start the page zero, user ROM at EPROM address $0020 through $004F. Start the user ROM at EPROM address $0100 through $1EFF with vectors from $1FF4 to $1FFF. All unused bytes, including the user's space, must be set to zero.

To use a 2764 or 6874 EPROM or the EEPROM in an MC68HC805C4, two are required. Start the page zero user ROM data at EPROM or EEPROM address $0020 through $004F in the first device. Start the user ROM data at address $0100 through $10FF in the first device. The remainder of the user ROM data should go from $0100 through $10FF in the second device, with vectors from $0004 through $000F. The EPROM devices or EEPROM MCU devices should be clearly marked to indicate which device corresponds to which address space.

For shipment to Motorola, EPROMs should be placed in a conductive IC carrier and packed securely. Styrofoam is not acceptable for shipment.



xxx = Customer ID

Verification Media

All original pattern media (EPROMs or floppy disks) are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked, and the verification form should be completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for the creation of the customer mask. To aid in the verification process, Motorola will program *customer supplied* blank EPROM(s) or DOS disks from the data file used to create the custom mask.

ROM VERIFICATION UNITS (RVUs)

Ten MCUs containing the customer's ROM pattern will be sent for program verification. These units will have been made using the custom mask, but are for the purpose of ROM verification only. For expediency, the MCUs are unmarked, packaged in ceramic, and tested with five volts at room temperature. These RVUs are free with the minimum order quantity, but are not production parts. RVUs are not backed or guaranteed by Motorola Quality Assurance.

ORDERING INFORMATION

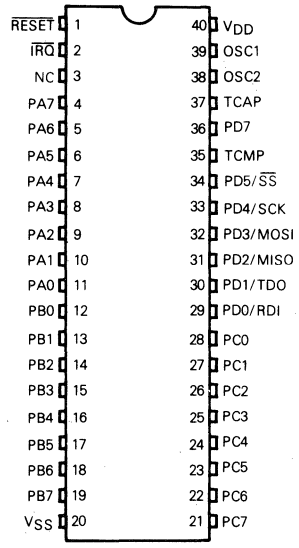
The following table provides ordering information pertaining to the package type, temperature, and MC order numbers for the MC68HC05C8 device.

Package Type	Temperature	MC Order Number
Plastic (P Suffix)	0°C to +70°C	MC68HC05C8P
	−40°C to +85°C	MC68HC05C8CP
	−40° to +105°C	MC68HC05C8VP
	−40°C to +125°C	MC68HC05C8MP
PLCC (FN Suffix)	0°C to +70°C	MC68HC05C8FN
	−40°C to +85°C	MC68HC05C8CFN
	−40°C to +105°C	MC68HC05C8VFN
	−40°C to +125°C	MC68HC05C8MFN

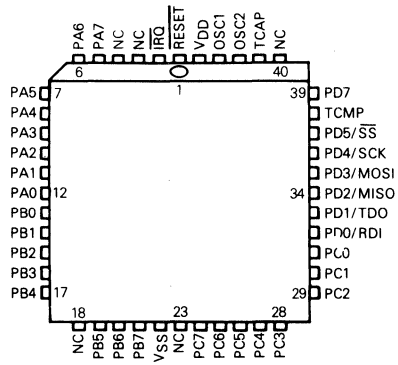
MDOS is a trademark of Motorola Inc.
MS is a trademark of Microsoft, Inc.
IBM is a registered trademark of International Business Machines Corporation.

PIN ASSIGNMENTS

40-PIN DUAL-IN-LINE PACKAGE



44-LEAD PLCC PACKAGE



NOTE: Bulk substrate tied to V_{SS}.

3

Continued in Volume II

MOTOROLA

Literature Distribution Centers:

USA: Motorola Literature Distribution; P.O. Box 20912; Phoenix, Arizona 85036.
EUROPE: Motorola Ltd.; European Literature Center; 88 Tanners Drive, Blakelands Milton Keynes, MK145BP, England.
ASIA PACIFIC: Motorola Semiconductors H.K. Ltd.; P.O. Box 80300; Cheung Sha Wan Post Office; Kowloon Hong Kong.

TB46503B021