

HC05

MC68HC705C8

TECHNICAL
DATA



Freescale Semiconductor, Inc.

Freescale Semiconductor, Inc.

**For More Information On This Product,
Go to: www.freescale.com**

MC68HC705C8

HCMOS MICROCONTROLLER UNIT

Motorola reserves the right to make changes without further notice to any products herein to improve reliability, function or design. Motorola does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights nor the rights of others. Motorola products are not authorized for use as components in life support devices or systems intended for surgical implant into the body or intended to support or sustain life. Buyer agrees to notify Motorola of any such intended end use whereupon Motorola shall determine availability and suitability of its product or products for the use intended. Motorola and  are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Employment Opportunity/Affirmative Action Employer.

©MOTOROLA INC., 1990

Freescale Semiconductor, Inc.

Freescale Semiconductor, Inc.

**For More Information On This Product,
Go to: www.freescale.com**

TABLE OF CONTENTS

Paragraph Number	Title	Page Number
---------------------	-------	----------------

Section 1 Introduction

Section 2 Signal Description, Input/Output Programming, Memory and CPU Registers

2.1	Signal Description.....	2-1
2.1.1	V _{DD} and V _{SS}	2-1
2.1.2	I _{RO}	2-1
2.1.3	OSC1 and OSC2	2-1
2.1.3.1	Crystal.....	2-2
2.1.3.2	Ceramic Resonator.....	2-3
2.1.3.3	External Clock	2-3
2.1.4	Input Capture (TCAP)	2-3
2.1.5	Output Compare (TCMP)	2-3
2.1.6	RESET	2-3
2.1.7	Input/Output Ports PA7–PA0, PB7–PB0, PC7–PC0)	2-3
2.1.8	Fixed Input Port (PD7, PD5–PD0)	2-4
2.1.9	V _{PP}	2-4
2.2	Input/Output Programming	2-4
2.2.1	I/O Port Programming	2-4
2.2.2	Fixed Input Port Programming	2-5
2.2.3	Serial Port (SCI and SPI) Programming	2-6
2.3	Memory.....	2-6
2.3.1	Control Registers	2-8
2.3.1.1	Option Register	2-8
2.3.1.2	Program Register.....	2-9
2.3.2	EPROM Erasing	2-9
2.3.3	OTPROM/EPROM (PROM) Programming with the MC68HC05PGMR	2-9
2.3.3.1	Standalone Programming.....	2-10
2.3.3.2	Before Programming.....	2-14
2.3.3.3	Preprogramming Steps	2-14

TABLE OF CONTENTS (Continued)

Paragraph Number	Title	Page Number
2.3.4	Program and Verify PROM.....	2-15
2.3.5	Verify PROM Contents.....	2-16
2.3.6	Secure PROM.....	2-16
2.3.6.1	Secure PROM and Verify	2-17
2.6.3.2	Secure PROM and Dump.....	2-17
2.3.7	Load Program into RAM and Execute	2-18
2.3.8	Execute Program in Ram	2-19
2.3.9	Dump PROM Contents	2-19
2.4	CPU Registers	2-20
2.4.1	Accumulator (A)	2-20
2.4.2	Index Register (X).....	2-20
2.4.3	Program Counter (PC)	2-20
2.4.4	Stack Pointer (SP).....	2-20
2.4.5	Condition Code Register (CCR).....	2-21
2.4.5.1	Half Carry (H)	2-21
2.4.5.2	Interrupt (I).....	2-21
2.4.5.3	Negative (N).....	2-21
2.4.5.4	Zero (Z)	2-21
2.4.5.5	Carry/Borrow (C).....	2-22

Section 3

Resets, Interrupts, Low Power, and Data Retention Modes

3.1	Resets	3-1
3.1.1	Poweron Reset (POR)	3-1
3.1.2	External Reset	3-1
3.1.3	Computer Operating Properly (COP) Watchdog Timer Reset.....	3-2
3.1.3.1	COP Reset Register	3-2
3.1.3.2	COP Control Register	3-2
3.1.4	Clock Monitor Reset.....	3-3
3.2	Interrupts.....	3-4
3.2.1	External Interrupt.....	3-5
3.2.2	Software Interrupt (SWI).....	3-8
3.2.3	Timer Interrupt.....	3-8
3.2.4	SCI Interrupts	3-8
3.2.5	SPI Interrupts	3-8

TABLE OF CONTENTS (Continued)

Paragraph Number	Title	Page Number
3.3	Low-Power Modes	3-9
3.3.1	STOP Mode.....	3-9
3.3.1.1	SCI During STOP Mode	3-10
3.3.1.2	SPI During STOP Mode	3-10
3.3.2	WAIT Mode.....	3-10
3.3.3	Data Retention Mode	3-11

Section 4 Timer

4.1	Counter	4-1
4.2	Output Compare Register (OCR) \$0016–\$0017	4-3
4.3	Input Capture Register (ICR) \$0014–\$0015	4-4
4.4	Timer Control Register	4-5
4.5	Timer Status Register.....	4-6
4.6	Timer During WAIT Mode.....	4-7
4.7	Timer During STOP Mode.....	4-7

Section 5 Serial Communications Interface

5.1	Data Format	5-2
5.2	Wake-Up Feature	5-2
5.3	Receive Data in.....	5-2
5.4	Start Bit Detection Following a Framing Error	5-3
5.5	Transmit Data Out	5-3
5.6	Functional Description.....	5-3
5.7	Registers.....	5-6
5.7.1	Serial Communications Data Register	5-7
5.7.2	Serial Communications Control Register 1.....	5-7
5.7.3	Serial Communications Control Register 2.....	5-8
5.7.4	Serial Communications Status Register.....	5-9
5.7.5	Baud Rate Register	5-10

Section 6 Serial Peripheral Interface

6.1	Signal Description.....	6-2
6.1.1	Master Out, Slave In	6-2
6.1.2	Master In, Slave Out	6-2

TABLE OF CONTENTS (Continued)

Paragraph Number	Title	Page Number
6.1.3	Serial Clock.....	6-2
6.1.4	Slave Select	6-2
6.2	Functional Description.....	6-3
6.3	Registers.....	6-5
6.3.1	Serial Peripheral Control Register	6-6
6.3.2	Serial Peripheral Status Register	6-7
6.3.3	Serial Peripheral Data I/O Register.....	6-8

Section 7

Instruction Set and Addressing Modes

7.1	Instruction Set.....	7-1
7.1.1	Register/Memory Instructions.....	7-6
7.1.2	Read-Modify-Write Instructions	7-6
7.1.3	Branch Instructions	7-7
7.1.4	Bit Manipulation Instructions	7-8
7.1.5	Control Instructions.....	7-8
7.1.6	Opcode Map Summary	7-8
7.2	Addressing Modes.....	7-10
7.2.1	Immediate.....	7-10
7.2.2	Direct	7-10
7.2.3	Extended	7-10
7.2.4	Relative	7-11
7.2.5	Indexed, No Offset	7-11
7.2.6	Indexed, 8-Bit Offset.....	7-11
7.2.7	Indexed, 16-Bit Offset.....	7-11
7.2.8	Bit Set/Clear	7-12
7.2.9	Bit Test and Branch.....	7-12
7.2.10	Inherent.....	7-12

Section 8

Electrical Specifications

8.1	Maximum Ratings.....	8-1
8.2	Thermal Characteristics	8-1
8.3	Power Considerations	8-2
8.4	DC Electrical Characteristics ($V_{DD} = 5.0$ Vdc)	8-3
8.5	DC Electrical Characteristics ($V_{DD} = 3.3$ Vdc)	8-4
8.6	Control Timing	8-8

TABLE OF CONTENTS (Concluded)

Paragraph Number	Title	Page Number
8.7	Control Timing	8-10
8.8	Serial Peripheral Interface (SPI) Timing ($V_{DD} = 5.0$ Vdc)	8-11
8.9	Serial Peripheral Interface (SPI) Timing ($V_{DD} = 3.3$ Vdc)	8-12

**Section 9
Mechanical Data**

9.1	Ordering Information	9-1
9.2	Pin Assignments.....	9-2
9.2.1	40-Pin Dual-in-Line Package	9-2
9.2.2	44-Lead PLCC Package	9-2
9.3	Package Dimensions	9-3

LIST OF ILLUSTRATIONS

Figure Number	Title	Page Number
1-1	Block Diagram.....	1-2
2-1	Oscillator Connections.....	2-2
2-2	Typical Port I/O Circuit.....	2-5
2-3	Memory Map	2-7
2-4	OTPROM/EPROM Programming	2-11
2-5	PROM Programming Circuit.....	2-12
3-1	Interrupt Stacking Order	3-5
3-2	Reset and Interrupt Processing Flowchart.....	3-6
3-3	External Interrupt.....	3-7
3-4	STOP Function Flowchart	3-9
3-5	WAIT Function Flowchart.....	3-11
4-1	Timer Block Diagram.....	4-2
5-1	Data Format	5-2
5-2	SCI Block Transmitter Block Diagram.....	5-4
5-3	SCI Receiver Block Diagram.....	5-5
6-1	Master-Slave System Configuration.....	6-1
6-2	Data Clock Timing Diagram	6-3
6-3	SPI Block Diagram	6-4
6-4	SPI Master-Slave Interconnections.....	6-5
8-1	Equivalent Test Load.....	8-2
8-2	Typical Voltage Compared to Current	8-5
8-3	Typical Current vs Internal Frequency for RUN and WAIT Modes	8-6
8-4	Total Current Drain vs Frequency	8-7
8-5	Stop Recovery Timing Diagram.....	8-9
8-6	Timer Relationships	8-10
8-7	SPI Timing Diagrams	8-13
8-8	Power-On Reset and RESET	8-15

LIST OF TABLES

Number	Title	Page Number
2-1	I/O Pin Functions	2-5
3-1	COP Timeout Period.....	3-3
5-1	Prescaler Highest Baud Rate Frequency Output.....	5-11
5-2	Transmit Baud Rate Output for a Given Prescaler Output.....	5-11
7-1	Instructions, Addressing Modes, and Execution Times.....	7-2
7-2	Opcode Map	7-9

SECTION 1 INTRODUCTION

The MC68HC705C8 (HCMOS) microcontroller unit (MCU) is a member of the M68HC05 Family of microcontrollers and is available in either erasable programmable read-only memory (EPROM), or one-time programmable read-only memory (OTPROM). 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.

Figure 1-1 depicts the hardware features. Additional features available on the MCU are as follows:

- On-Chip Oscillator with Crystal/Ceramic Resonator
- Memory-Mapped I/O
- Selectable Memory Configurations
- Computer Operating Properly (COP) Watchdog Timer
- Clock Monitor
- 24 Bidirectional I/O Lines and 7 Input-Only Lines
- Serial Communications Interface (SCI) System
- Serial Peripheral Interface (SPI) System
- Bootstrap Capability
- Power-Saving STOP, WAIT, and Data Retention Modes
- Single 3.0- to 5.5-Volt Supply (2-Volt Data Retention Mode)
- Fully Static Operation
- Software-Programmable External Interrupt Sensitivity

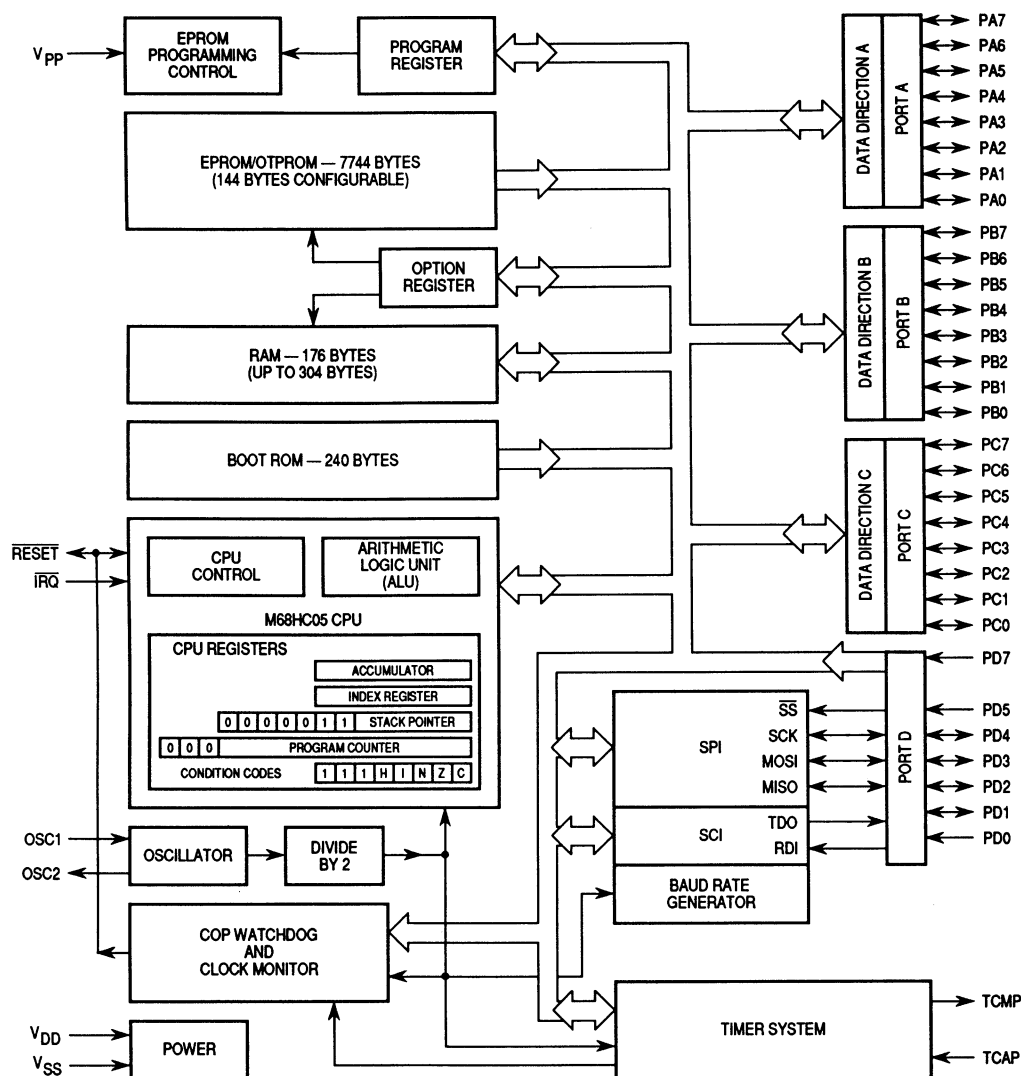


Figure 1-1. Block Diagram

SECTION 2

SIGNAL DESCRIPTION, INPUT/OUTPUT PROGRAMMING, MEMORY AND CPU REGISTERS

This section provides a description of the signals, input/output programming, memory, and CPU registers.

2.1 SIGNAL DESCRIPTION

The following paragraphs provide a description of the MC68HC705C8 signals. Reference is made, where applicable, to other sections that contain more detail about the function being performed.

2.1.1 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.

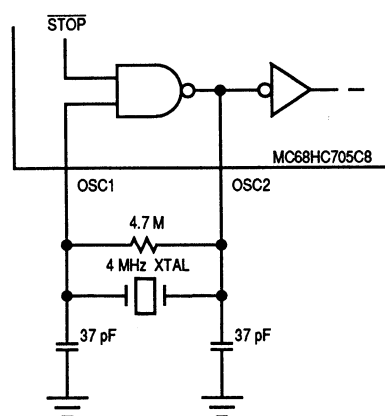
2.1.2 $\overline{IRQ}$

This pin provides the capability for applying an asynchronous external interrupt to the MCU. It has a programmable option that provides two different interrupt triggering sensitivity choices. Refer to **3.2 INTERRUPTS** for more detail.

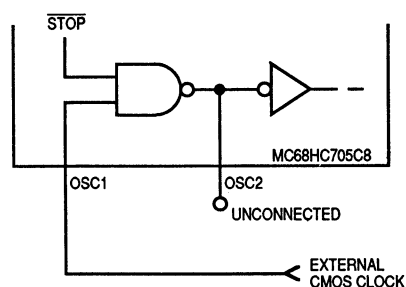
2.1.3 OSC1 and OSC2

These two pins provide connections for an on-chip clock oscillator circuit. A crystal, a ceramic resonator, or an external signal connects to these pins to provide a system clock. Each alternative is discussed in subsequent paragraphs. In all cases, particular care should be taken in the circuit board layout around the oscillator pins. The oscillator frequency (f_{OSC}) connected to these pins is two times the internal bus operating frequency (f_{OP}).

2.1.3.1 CRYSTAL. The circuit in Figure 2-1(a) shows a typical crystal oscillator circuit for a generic 4 MHz, AT-cut, parallel resonant crystal which is designed for a 20 pF load capacitance. Crystal load capacitance for this circuit assumes additional stray capacitance of 3 pF on each side of the crystal so that load capacitance is equal to the series combination of two 40 pF capacitors (or the desired 20 pF). These values do not apply for other frequencies and the crystal manufacturer's recommendations should be followed in all cases. Crystal parameters determine the external component values required to provide maximum stability and reliable starting.



(a) Crystal/Ceramic Resonator Oscillator Connections



(b) External Clock Source Connections

Figure 2-1. Oscillator Connections

2.1.3.2 CERAMIC RESONATOR. A ceramic resonator can be used in place of the crystal in cost-sensitive applications. The circuit in Figure 2-1(a) can be used for a ceramic resonator, but the manufacturer of the resonator should be consulted for specific information on external component values, since the ceramic resonator characteristics determine the external component values required.

2.1.3.3 EXTERNAL CLOCK. An external clock from another CMOS-compatible device can be connected to the OSC1 input, with the OSC2 input not connected, as shown in Figure 2-1(b). The OSC2 pin is normally left open, but a 10K–100K load resistor to V_{DD} can be used to reduce EMI noise emission.

NOTE

The bus frequency (f_{op}) is one-half the external or crystal frequency (f_{osc}), while the processor clock cycle (t_{cyc}) is two times the f_{osc} period.

2.1.4 Input Capture (TCAP)

This pin controls the input capture feature for the on-chip programmable timer.

2.1.5 Output Compare (TCMP)

This pin provides an output for the output compare feature of the on-chip timer.

2.1.6 RESET

This pin is used as an input to reset the MCU and provide an orderly start-up procedure by pulling $\overline{RESET}$ low. As an output, the $\overline{RESET}$ pin indicates that an internal MCU failure has been detected.

2.1.7 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 **2.2 INPUT/OUTPUT PROGRAMMING** for additional information.

2.1.8 Fixed Input Port (PD7, PD5–PD0)

These seven lines comprise port D, a fixed input port. All special functions that are enabled (SPI, SCI) affect this port. Refer to **2.2 INPUT/OUTPUT PROGRAMMING** for additional information.

2.1.9 V_{pp}

This pin is used to program the OTPROM or EPROM. V_{pp} should be connected to V_{DD} for normal operation.

CAUTION

Connecting the V_{pp} pin (programming voltage) to V_{SS} (ground), could result in damage to the MCU.

2.2 INPUT/OUTPUT PROGRAMMING

Input/output port programming, fixed input port programming, and serial port programming are discussed in the following paragraphs.

2.2.1 I/O 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 or 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 the port bits are programmed as outputs, the latched output data is readable as input data regardless of the logic level 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 2-2 for typical port circuitry and to Table 2-1 for a list of the I/O pin functions.

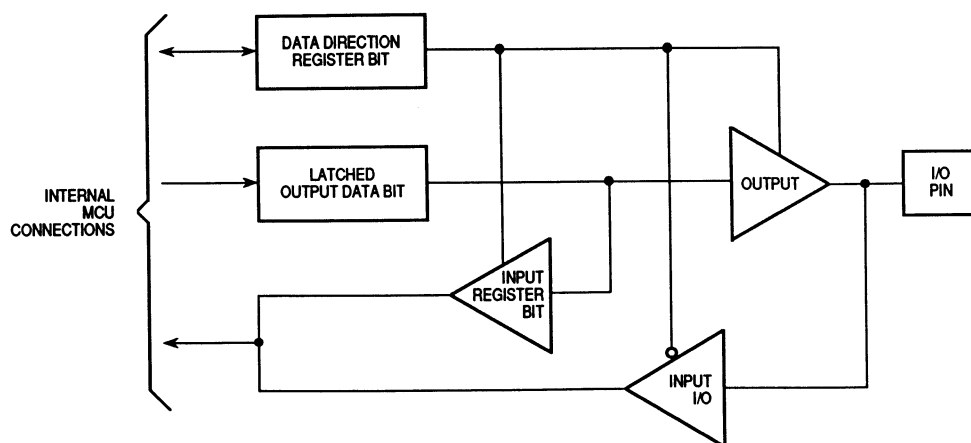


Figure 2-2. Typical Port I/O Circuit

Table 2-1. 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.

2.2.2 Fixed Input Port Programming

Port D is a fixed input port of seven input-only lines (PD7, PD5–PD0), that monitors the external pins whenever the SCI or SPI is disabled. When SCI or SPI is enabled, associated port D bits read as zero. When these functions are disabled, the associated bits read the state of the pin at the time of the read operation. After reset, all seven bits become valid inputs because all special function drivers are disabled.

NOTE

Any unused inputs and I/O ports should be tied to an appropriate logic level, either VDD or VSS).

2.2.3 Serial Port (SCI and SPI) Programming

The SCI and SPI use the port D pins for their functions. The SCI requires two pins (PD1–PD0) for its transmit data output (TDO) and receive data input (RDI), respectively. The SPI function requires four of the pins (PD5–PD2) for its slave select ($\overline{SS}$), serial clock (SCK), serial data output/input (MOSI), and serial data input/output (MISO), respectively.

2.3 MEMORY

The MCU is capable of addressing 8192 bytes of memory and I/O registers. The locations consist of user programmable read-only memory (PROM, either OTPROM or EPROM), user random-access memory (RAM), bootstrap read-only memory (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 **3.2 INTERRUPTS** for additional information.

NOTE

Using the stack area for data storage or temporary work locations requires care to avoid overwriting due to stacking from an interrupt or subroutine call.

In the MC68HC705C8, one of four selectable memory configurations is selected by the state of the RAM1 and RAM0 bits of the options register (OPTION), located at \$1FDF. Reset or poweron reset clears these bits, automatically selecting the first memory configuration shown in the following table:

RAM0	RAM1	RAM Bytes	PROM Bytes
0	0	176	7744
1	0	208	7696
0	1	272	7648
1	1	304	7600

Figure 2-3 illustrates the four memory configurations.

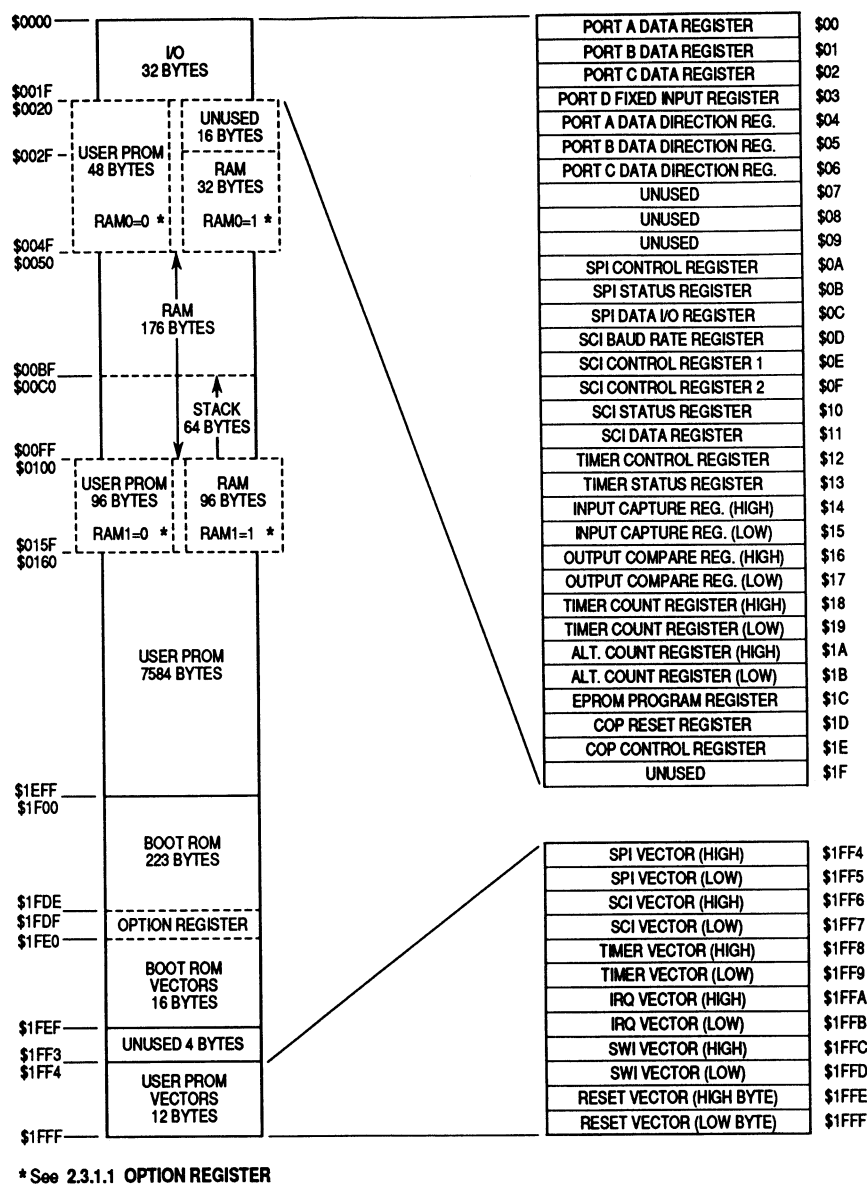


Figure 2-3. Memory Map

2.3.1 Control Registers

The following paragraphs describe the registers that control memory configuration, PROM security, programming, latching, and $\overline{\text{IRQ}}$ edge- or level-sensitivity.

2.3.1.1 OPTION REGISTER. The option register is used to select the $\overline{\text{IRQ}}$ sensitivity, enable the PROM security, and select the memory configuration.

	7	6	5	4	3	2	1	0	
\$1FDF	RAM0	RAM1	0	0	SEC	—	IRQ	0	OPTION
RESET:	0	0	0	0	*	—	1	0	

*The SEC bit is implemented as an EPROM cell.

RAM0 — Random-Access Memory Control Bit 0

- 1 = Maps 32 bytes of RAM into page zero starting at address \$0030. Addresses from \$0020 to \$002F are reserved. This bit can be read or written at any time, allowing memory configuration to be changed during program execution.
- 0 = Provides 48 bytes of PROM at location \$0020–\$005F.

RAM1 — Random-Access Memory Control Bit 1

- 1 = Maps 96 bytes of RAM into page one starting at address \$0100. This bit can be read or written at any time, allowing memory configuration to be changed during program execution.
- 0 = Provides 96 bytes of PROM at location \$0100.

SEC — Security

- 1 = Bootloader disabled, MCU operates only in single-chip mode.
 - 0 = Security off, bootloader enabled.
- This bit is implemented as an EPROM cell and is not affected by reset (U).

IRQ — Interrupt Request Pin Sensitivity

- 1 = $\overline{\text{IRQ}}$ pin is both negative edge- and level-sensitive.
 - 0 = $\overline{\text{IRQ}}$ pin is negative edge-sensitive only.
- IRQ is set only by reset, but can be cleared by software. This bit can only be written once.

Bits 5, 4, 0 — Not used; these bits always read zero.

Bit 2

This bit is not affected by reset (—), and can read either one or zero.

2.3.1.2 PROGRAM REGISTER. The program register (\$1C) is used to perform PROM programming.

	7	6	5	4	3	2	1	0	
\$001C	0	0	0	0	0	LAT	0	PGM	PROG
RESET:	0	0	0	0	0	0	0	0	

LAT — Latch Enable

1=Enables PROM data and address bus latches for programming on the next byte write cycle.

0=Latch disabled. PROM data and address buses are unlatched for normal CPU operations.

This bit is both readable and writable.

PGM — Program

1=Enables Vpp power to the PROM for programming.

0=Vpp is disabled.

If LAT is cleared, PGM cannot be set.

Bits 1, 3–7 — Not used; these bits always read zero.

2.3.2 EPROM Erasing

In the MC68HC705C8, the erased state of an OTPROM or EPROM byte is \$00. EPROM devices can be erased by exposure to a high intensity ultraviolet (UV) light with a wavelength of 2537 Å. The recommended erasure dosage (UV intensity on a given surface area × exposure time) is 15 Ws/cm². UV lamps should be used without short-wave filters, and the EPROM device should be positioned about 1 inch from the UV source.

OTPROM devices are shipped in an erased state. Once programmed, they cannot be erased. Electrical erasing procedures cannot be performed on either OTPROM or EPROM devices.

2.3.3 OTPROM/EPROM (PROM) Programming with the MC68HC05PGMR

Programming of the internal PROM of the MC68HC705C8 MCU is most readily accomplished using the MC68HC05PGMR Programmer Board, available from Motorola. This printed circuit board (PCB) permits the user to program the

MCU using an 8K EPROM device already programmed with user code (standalone programming), or to download developed user code from a host computer using the board's terminal I/O port. In addition, the MC68HC05PGMR can be used to perform limited evaluation of the programmed MCU. Only standalone programming is discussed in the following paragraphs. For more information concerning the MC68HC05PGMR and its usages, contact your local Motorola representative for a copy of MC68HC05PGMR2/D1, *MC68HC05PGMR Programmer Board User's Manual #2*. Refer to Figure 2-4 for an EPROM programming flowchart and to Figure 2-5 for a schematic of the MC68HC05PGMR PCB.

2.3.3.1 STANDALONE PROGRAMMING. The programming of the PROM MCU in standalone mode is a matter of installing the MCU in the PCB, along with an 8K EPROM device programmed with user code, then subjecting it to a series of routines. The routines necessary to program, verify, then secure the PROM device are:

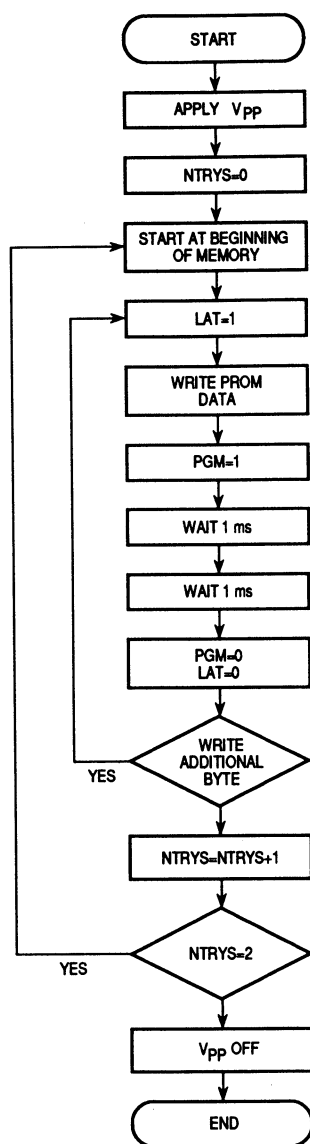
- Program and Verify PROM
- Verify PROM Contents (Only)
- Secure PROM and Verify
- Secure PROM and Dump (Through SCI)

Other board routines available to the user are:

- Load Program into RAM and Execute
- Execute Program in RAM
- Dump PROM Contents (Binary Upload)

The user first configures the MCU for the bootstrap mode of operations by installing a fabricated jumper across pins 1 and 2 of the board's mode select header, J1. Next, the board's mode switches (S3, S4, S5, and S6) are set to determine the routine to be executed after the next reset, as shown below:

Routine	S3	S4	S5	S6
Program and Verify PROM	Off	Off	Off	Off
Verify PROM Contents (Only)	Off	Off	On	Off
Secure PROM Contents and Verify	On	Off	On	Off
Secure PROM Contents and Dump	On	On	On	Off
Load Program into RAM and Execute	Off	On	Off	Off
Execute Program in RAM	Off	Off	Off	On
Dump PROM Contents	Off	On	On	Off



NOTE: If programming in user mode the $\overline{\text{IRQ}}$ pin must be held to 9 V.

Figure 2-4. OTPROM/EPROM Programming

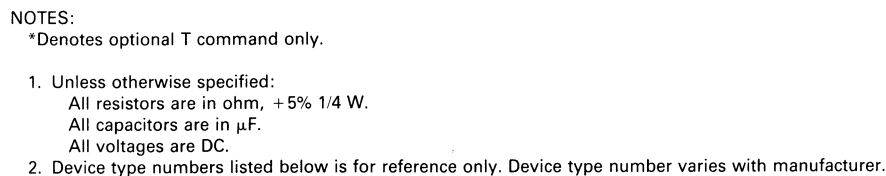
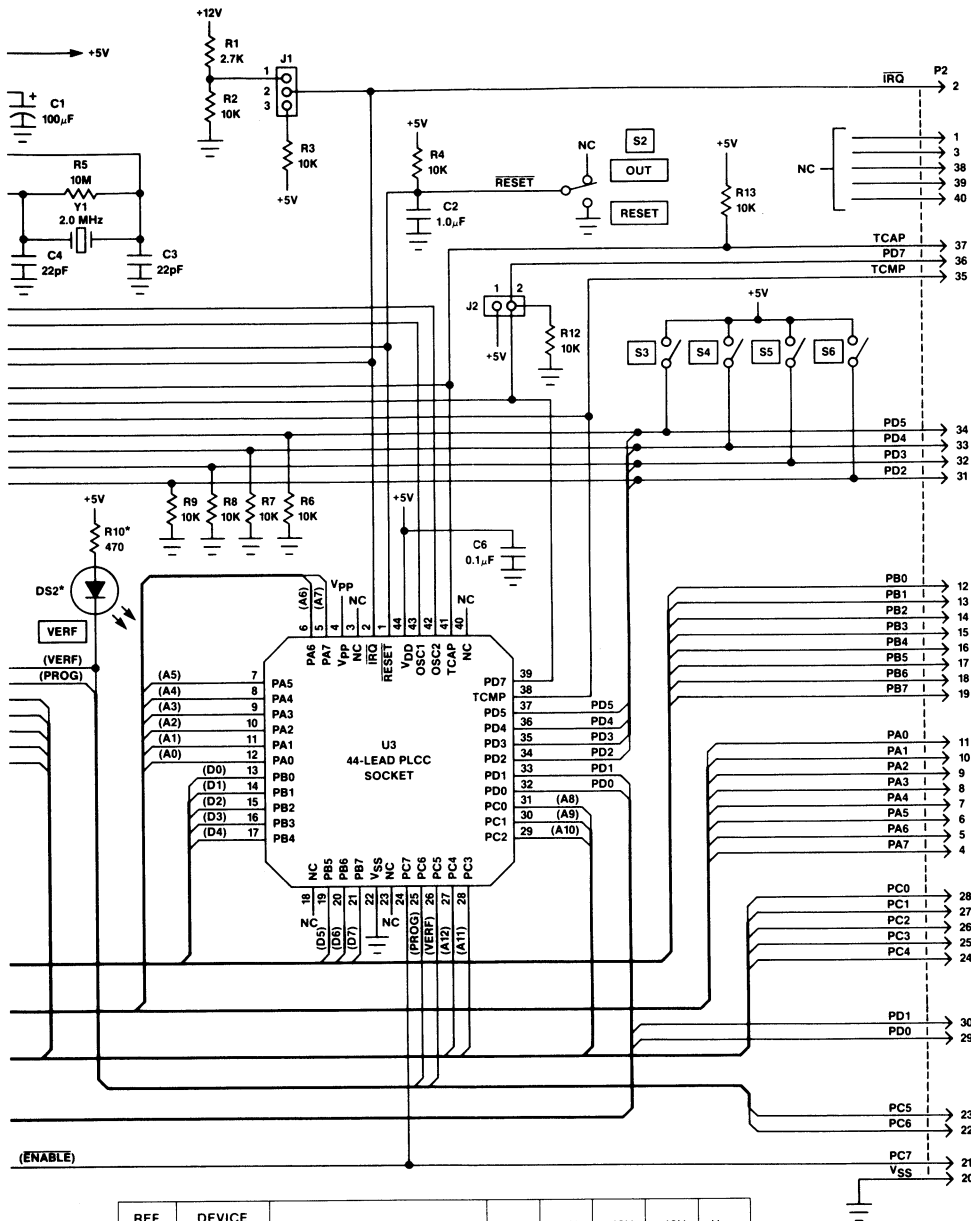


Figure 2-5. PROM Programming Circuit



REF DES	DEVICE TYPE	NOTES	GND	+5V	+12V	-12V	Vpp
U1	2764	8K x 8 EPROM	14, 20	1, 26, 27, 28			
U2	MCU	40-PIN DIP SOCKET	20	40			3
U3	MCU	44-LEAD PLCC SOCKET	22	44			4
U4	MC145406	DRIVER/RECEIVER	9	16	1	8	
VR1	NMA0512S	DC-DC CONVERTER	2, 5	1	6	4	

2.3.3.2 BEFORE PROGRAMMING. Before attempting to program the PROM of any of the variants of the MC68HC705C8 using an MC68HC05PGMR PCB in stand-alone mode, the user should ensure that:

- A jumper is installed on pins 1 and 2 of mode select header J1.
- An 8K 2764-type EPROM has been programmed with the necessary user code.
- The erasure window (if any) of the device to be programmed is covered.
- V_{DD} of +5 Vdc is available on the board.
- V_{pp} of +15.5 Vdc $\pm$ 0.25 Vdc, measured at connector P1 (pin 5) or V_{pp} of +14.75 Vdc $\pm$ 0.25 Vdc, measured at socket U2 or U3 (pin 3) whichever socket is to be used, is available on the board.

CAUTION

If the V_{pp} level at the MCU exceeds +16 Vdc, then the MC68HC705C8 MCU device will suffer permanent damage.

2.3.3.3 PREPROGRAMMING STEPS. Once the above conditions are met, the user should take the following steps before beginning programming:

1. Remove the V_{pp} power source.
2. Set switch 1 in the OFF position (removes V_{DD}).
3. Place the programmed 8K 2764-type EPROM in socket U1.
4. Insert the erased PROM MCU device to be programmed in the proper socket:
 - MC68HC705C8S or MC68HC705C8P in socket U2 (40-pin DIP), or;
 - MC68HC705C8FN in socket U3 (44-pin PLCC), with the device notch at the upper right corner of the socket.
5. Set switch S2 in the RESET position.

NOTE

No PROM MCU should be inserted in or removed from its board socket (U2 or U3) while V_{pp} (P1, slot 5) or V_{DD} (switch 1) is active on the board.

2.3.4 Program and Verify PROM

The program and verify PROM routine copies the contents of the external 8K EPROM into the MCU PROM, with direct correspondence between the addresses. Memory addresses in the MCU that are not implemented in PROM are skipped. Unprogrammed addresses in the 8K EPROM being copied should contain \$00 bytes to speed up the programming process.

To run the program and verify PROM routine on the PROM MCU, take the following steps:

1. Set switch 1 in the ON position (restores V_{DD}).
2. Restore the V_{pp} power source.
3. Set switches S3, S4, S5, and S6 in the OFF position (selects proper routine).
4. Set switch 2 in the OUT position (routine is activated).

The red light emitting diode (LED) is illuminated, showing that the programming part of the routine is running. The LED goes out when programming is finished. The verification part of the routine now begins. When the green LED is illuminated, verification is successfully completed and the routine is finished.

5. Set switch 2 in the RESET position.

At this point, if no other MCU is to be programmed or secured, remove V_{pp} power from the board. If another routine is to be performed on the MCU being programmed, the user can then set switches S3, S4, S5, and S6 to the positions necessary to select the next routine, and begin it by setting switch 2 to the OUT position. If no other routine is to be performed, remove V_{DD} from the board and remove the MCU from the programming socket.

2.3.5 Verify PROM Contents

The verify PROM contents routine is normally run automatically after the PROM is programmed. Direct entry to this routine causes the PROM contents of the MCU to be compared to the contents of the external memory locations of the 8K EPROM at the same addresses.

To invoke the verify PROM contents routine of the MCU, take the following steps:

1. Set switch 1 in the ON position (restores V_{DD}).
2. Connect V_{pp} to V_{DD} .
3. Set switches S3, S4, and S6 in the OFF position.
4. Set S5 in the ON position.
5. Set switch 2 in the OUT position (routine is activated).

The red LED is not illuminated during this routine, since no programming takes place. If verification fails, the routine halts with the failing address in the external memory bus. When the green LED is illuminated, verification is successfully completed and the routine is finished.

6. Set switch 2 in the RESET position.

At this point, if another routine is to be performed on the MCU being programmed, the user can set switches S3, S4, S5, and S6 to the positions necessary to select the next routine, and move switches S2 to the OUT position to start it. If no other routine is to be performed, remove V_{DD} from the board and remove the MCU from the programming socket.

2.3.6 Secure PROM

The secure PROM routines are used after the PROM is successfully programmed and verified. Only the SEC bit of the OPTION register (\$1FDF) is programmed, but V_{pp} is necessary. Once this bit is programmed, PROM is secure, and can be neither verified nor dumped.

2.3.6.1 SECURE PROM AND VERIFY. This routine is used after the PROM is successfully programmed to verify the contents of the MCU PROM against the contents of the 8K EPROM, then secure the PROM. To accomplish this routine, take the following steps:

1. Set switch 1 in the ON position (restores V_{DD}).
2. Restore V_{pp} power to the programming board.
3. Set switches S4 and S6 in the OFF position.
4. Set switches S3 and S5 in the ON position.
5. Set switch 2 in the OUT position (routine is activated).

Execution time for this routine is about one second.

6. Set switch 2 in the RESET position when the routine is completed.

No LED is illuminated during this routine. Further, the end of the routine does not mean that the SEC bit was verified. To ensure that security is properly enabled, attempt to perform another verify routine. If the green LED does not light, the PROM has been properly secured.

2.3.6.2 SECURE PROM AND DUMP. This routine is used after the PROM is successfully programmed to dump the contents of the MCU PROM through the SCI (binary upload), then secure the PROM. To accomplish this routine, take the following steps:

1. Set switch 1 in the ON position (restores V_{DD}).
2. Restore V_{pp} power to the programming board.
3. Set switch S6 in the OFF position.
4. Set switches S3, S4, and S5 in the ON position.
5. Set switch 2 in the OUT position (routine is activated).

Execution time for this routine is about one second.

6. Set switch 2 in the RESET position when the routine is completed.

No LED is illuminated during this routine. Further, the end of the routine does not mean that the SEC bit was verified. To ensure that security is properly enabled, attempt to perform another verify routine. If the green LED does not light, the PROM has been properly secured.

2.3.7 Load Program into RAM and Execute

In the load program in RAM and execute routine, user programs are loaded via the SCI port, and then executed. Data is loaded sequentially, starting at address \$0050. After the last byte is loaded, control is transferred to the RAM program starting at \$0051. The first byte loaded is the count of the total number of bytes in the program, plus the count byte. The program starts at location \$0051 in RAM. During initialization, the SCI is configured for eight data bits and one stop bit. The baud rate is 4800 with a 2 MHz crystal.

To load a program into RAM and execute it, take the following steps:

1. Set switch 1 in the ON position (restores V_{DD}).
2. Connect V_{pp} to V_{DD} .
3. Set switches S3, S5, and S6 in the OFF position.
4. Set switch S4 in the ON position.
5. Set switch 2 in the OUT position (routine is activated).

The downloaded program starts executing as soon as the last byte is received by the SCI.

Execution of the routine can be held off by setting the byte count in the count byte (the first byte loaded) to a value greater than the number of bytes to be loaded. After loading the last byte, the firmware waits for more data. Program execution does not begin. At this point, placing switch 2 in the RESET position resets the MCU with the RAM data intact. Any other routine can be entered, including the one to execute the program in RAM, simply by setting switches S3–S6 as necessary to select the desired routine, then setting switch 2 in the OUT position.

2.3.8 Execute Program In RAM

This routine allows the MCU to transfer control to a program previously loaded in RAM. This program is executed once bootstrap mode is entered, if switch S6 is in the ON position and switch 2 is in the OUT position, without any firmware initialization. The program must start at location \$0051 to be compatible with the load program in RAM routine.

To run the execute program in RAM routine, take the following steps:

1. Set switch 1 in the ON position (restores V_{DD}).
2. Connect V_{pp} to V_{DD} .
3. Set switch S6 in the OFF position.
4. Switches S3, S4, and S5 can be in either position.
5. Set switch 2 in the OUT position (routine is activated).

2.3.9 Dump PROM Contents

In the dump PROM contents routine, the PROM contents are dumped sequentially to the SCI output, provided the PROM has not been secured. The first location sent is \$0020, and the last location sent is \$1FFF. Unused locations are skipped so that no gaps exist in the data stream. The external memory address lines indicate the current location being sent. Data is sent with eight data bits and one stop bit at 4800 baud (with a 2 MHz crystal).

To run the dump PROM contents routine, take the following steps:

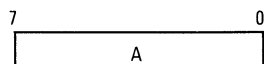
1. Set switch 1 in the ON position (restores V_{DD}).
2. Connect V_{pp} to V_{DD} .
3. Set switches S3 and S6 in the OFF position.
4. Set switches S4 and S5 in the ON position.
5. Set switch 2 in the OUT position (routine is activated).
6. Once PROM dumping is complete, set switch 2 in the RESET position.

2.4 CPU REGISTERS

The MC68HC705C8 MCU contains the CPU registers described in the following paragraphs.

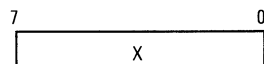
2.4.1 Accumulator (A)

The accumulator is a general-purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.



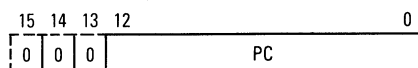
2.4.2 Index Register (X)

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 a 0-, 8- or 16-bit immediate value to create an effective address. The index register may also be used as a temporary storage area.



2.4.3 Program Counter (PC)

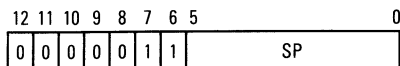
The program counter is a 13-bit register that contains the address of the next byte to be fetched. Since addresses are often expressed as 16-bit values, the PC register may be thought of as having three imaginary upper bits, which are always zero.



2.4.4 Stack Pointer (SP)

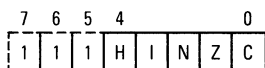
The stack pointer is a 13-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set to location \$00FF. The stack pointer is then decremented as data is pushed onto the stack and incremented 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 bits of the register to produce an address within the range of \$00FF to \$00C0. Subroutines and interrupts may use up to 64 (decimal) locations. If the total of 64 locations is exceeded, the SP wraps around and stores new information over the previously stored information. A subroutine call occupies two locations on the stack. An interrupt uses five locations.



2.4.5 Condition Code Register (CCR)

The CCR is a 5-bit register in which the H, N, Z, and C bits are used to indicate the results of the instruction just executed, and the I bit is used to enable interrupts. These bits can be individually tested by a program, and specific actions can be taken as a result of their state. The CCR should be thought of as having three additional upper bits that are always ones. The function of each of the lower five bits is explained in the following paragraphs.



2.4.5.1 HALF CARRY (H). This bit is set during add (ADD) and add with carry (ADC) operations to indicate that a carry occurred between bits 3 and 4. Operations on binary coded decimal (BCD) values require this status indicator.

2.4.5.2 INTERRUPT (I). When this bit is set, the timer, SCI, SPI, and external interrupts are masked (disabled). If an interrupt occurs while this bit is set, the interrupt is latched and is processed as soon as the I bit is cleared.

2.4.5.3 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).

2.4.5.4 ZERO (Z). When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

2.4.5.5 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.

SECTION 3

RESETS, INTERRUPTS, LOW POWER, AND DATA RETENTION MODES

The following paragraphs describe the resets, interrupts, and the low power and data retention modes.

3.1 RESETS

The MCU can be reset in the following four ways:

- An internal, power-on condition,
- An external, active-low input on the $\overline{\text{RESET}}$ pin,
- An internal computer operating properly (COP) watchdog timer reset condition, or
- An internal clock monitor reset condition.

The following paragraphs describe each of these conditions.

3.1.1 Poweron Reset (POR)

An internal reset is generated on powerup to allow the internal clock generator to stabilize. The power-on reset is strictly for power turnon 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 $\overline{\text{RESET}}$ pin is low at the end of 4064 t_{CYC} , the MCU remains in the reset condition until $\overline{\text{RESET}}$ goes high.

3.1.2 External Reset

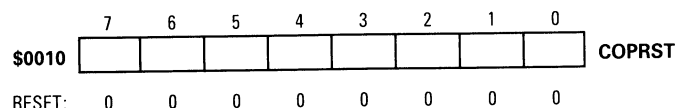
The MCU is reset when a logic zero is applied to the $\overline{\text{RESET}}$ pin for a period of eight processor cycles (t_{CYC}). After 6 t_{CYC} , $\overline{\text{RESET}}$ input is sampled. If the pin is still low, an external reset has occurred. If the $\overline{\text{RESET}}$ input is high, then the reset was initiated internally by either the computer operating properly (COP) watchdog timer, or by the clock monitor. This method of differentiating between external and internal reset conditions assumes that the $\overline{\text{RESET}}$ pin will rise to a logic one less than 2 t_{CYC} after its release, and that an externally generated reset should stay active for at least 8 t_{CYC} .

3.1.3 Computer Operating Properly (COP) Watchdog Timer Reset

The MCU includes a COP watchdog timer to help protect against software failures. Once the COP is enabled, a COP reset sequence must be executed on a periodic basis so the COP does not time out. Since the COP timer uses the internal bus clock, a clock monitor is included to guard against clock failure.

The COP reset register (\$1D) and the COP control register (\$1E) shown below are used to control the COP watchdog timer and clock monitor functions.

3.1.3.1 COP RESET REGISTER



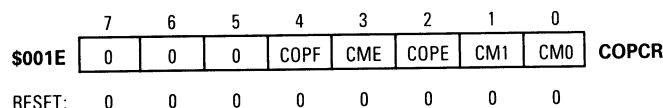
The sequence required to reset the COP timer is as follows:

Write \$55 to the COP reset register

Write \$AA to the COP reset register

Both write operations must occur in the order listed, but any number of instructions may be executed between the two write operations. The elapsed time between software resets must not be greater than the COP timeout period. Reading the COP reset register does not return valid data and does not affect the watchdog timer.

3.1.3.2 COP CONTROL REGISTER



COPF — Computer Operating Properly

1 = COP or clock monitor reset has occurred

0 = No COP or clock monitor reset has occurred

Reading the COP control register clears COPF

CME — Clock Monitor Enable

1 = Clock monitor enabled

0 = Clock monitor disabled

CME is readable at any time, but may be written only once.

COPE — Computer Operating Properly Enable

1 = COP timeout enabled

0 = COP timeout disabled

COPE is readable anytime, but may be written only once.

CM1 — Computer Operating Properly Mode 1

Used in conjunction with CM0 to establish the COP timeout period. CM1 can be read and set anytime, but is cleared only by reset. See Table 3-1.

CM0 — Computer Operating Properly Mode 0

Used in conjunction with CM1 to establish the COP timeout period. CM0 can be read and set anytime, but is cleared only by reset. See Table 3-1.

Bits 7–5 — Not used; these bits always read zero.

Table 3-1. COP Timeout Period

CM1	CM0	$f_{op}/2^{15}$ Divided By	$f_{osc}=4.0\text{ MHz}$ $f_{op}=2.0\text{ MHz}$	$f_{osc}=3.5795\text{ MHz}$ $f_{op}=1.7897\text{ MHz}$	$f_{osc}=2.0\text{ MHz}$ $f_{op}=1.0\text{ MHz}$	$f_{osc}=1.0\text{ MHz}$ $f_{op}=0.5\text{ MHz}$
0	0	1	16.38 ms	18.31 ms	32.77 ms	65.54 ms
0	1	4	65.54 ms	73.24 ms	131.07 ms	262.14 ms
1	0	16	262.14 ms	292.95 ms	524.29 ms	1.048 s
1	1	64	1.048 s	1.172 s	2.097 s	4.194 s

3.1.4 Clock Monitor Reset

When the CME bit in the COP control register is set, the clock monitor detects the absence of the internal bus clock for a certain period of time. The timeout period depends on processing parameters and varies from 5 to 100 μ s, which implies that systems using a bus clock rate of 200 kHz or less should not use the clock monitor function.

If a slow or absent clock is detected, the clock monitor causes a system reset. The reset is issued to the external system for four bus cycles via the bidirectional $\overline{\text{RESET}}$ pin.

Special consideration is required when using the STOP instruction with the clock monitor. Since STOP causes the system clocks to halt, the clock monitor issues a system reset when STOP is executed.

The clock monitor is a useful backup to the COP watchdog system. Because the watchdog timer requires a clock to function, it cannot indicate a system clock failure. The clock monitor would detect such a condition and force the MCU to a reset state. Clocks are not required for the MCU to reach a reset condition. They are, however, required to bring the MCU through the reset sequence, and back to the run condition.

3.2 INTERRUPTS

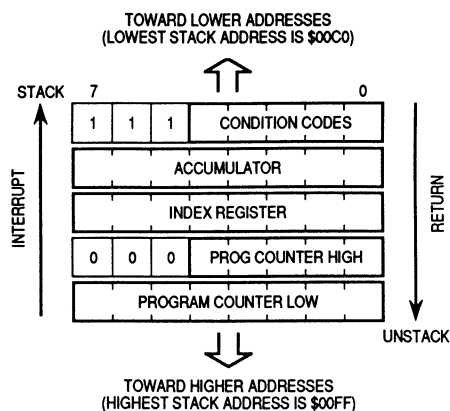
Unlike resets, which are asynchronous, stopping the flow of the program in mid-instruction and forcing it to restart, interrupts are synchronous. They do not cause the current instruction, the instruction already fetched and being operated on, to be halted. Instead, interrupts are considered pending until the current instruction is complete.

There are five sources of interrupt in the MC68HC705C8. An interrupt can be generated externally by the application of a logic low signal on the $\overline{\text{IRQ}}$ pin, or as a software instruction (SWI) that is part of the program being executed. The remaining sources of interrupt are the SPI, SCI, and timer, internal interrupt sources. These three and the $\overline{\text{IRQ}}$ are considered hardware interrupts and can be masked (disabled) by setting the I bit of the CCR.

Once the current instruction is complete, the processor checks all pending hardware interrupts. If there are unmasked pending hardware interrupts, and if the corresponding enable bit is set, the processor proceeds with interrupt processing. Otherwise, the next instruction is fetched and executed.

Interrupts cause the processor to save the register contents on the stack, and to set the interrupt mask, the I bit of the CCR, to prevent additional interrupts. The return-from-interrupt (RTI) instruction that ends each interrupt service routine causes the register contents to be recovered from the stack and normal processing to resume. The stacking order is shown in Figure 3-1.

If both an external interrupt and an internal 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 3-2 for the reset and interrupt instruction processing sequence.



NOTE: When an interrupt occurs, CPU registers are saved on the stack in the order PCL, PCH, X, A, CCR. On a return from interrupt registers are recovered from the stack in reverse order.

Figure 3-1. Interrupt Stacking Order

3.2.1 External Interrupt

If the interrupt mask bit (I bit) of the CCR is set, all interrupts except SWI 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 interrupt request 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 software option. See **2.3.1.1 OPTION REGISTER** for more information. Figure 3-3 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 two single pulses 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 normally not occur until an RTI occurs. This time (t_{LIL}) is obtained by adding 21 instruction cycles to

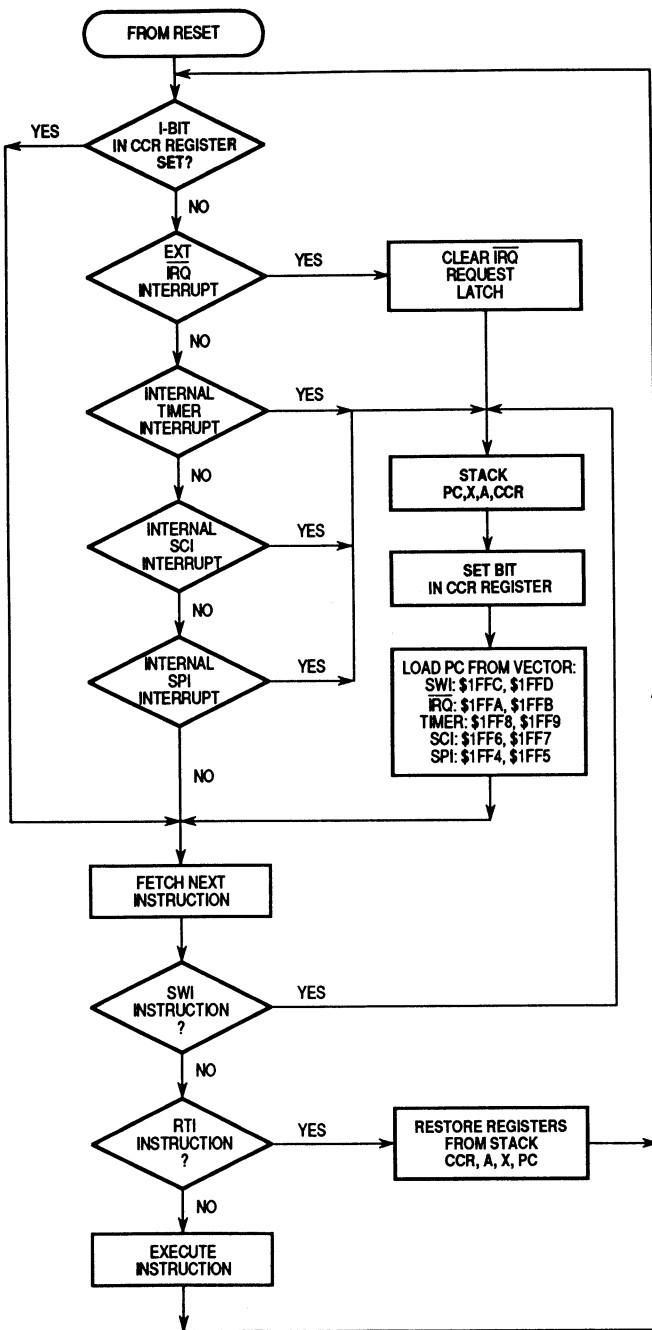
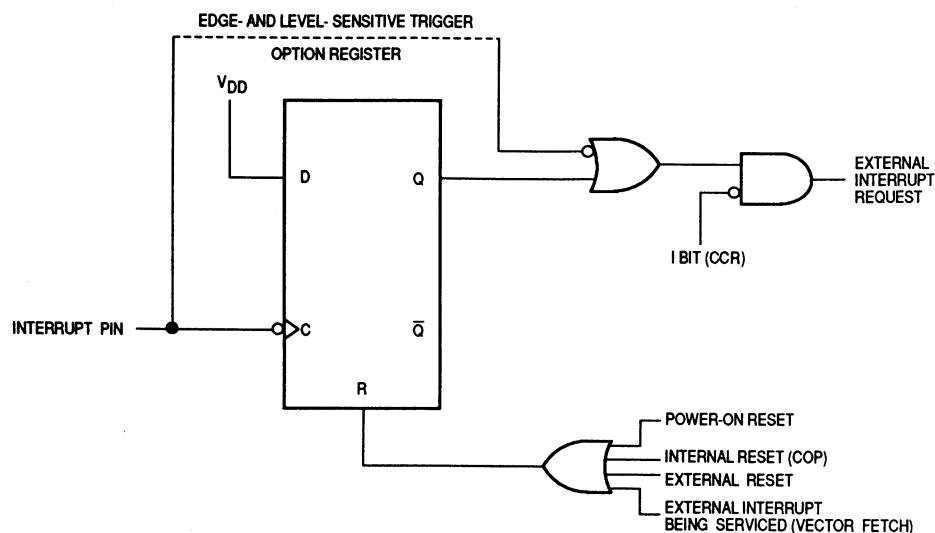
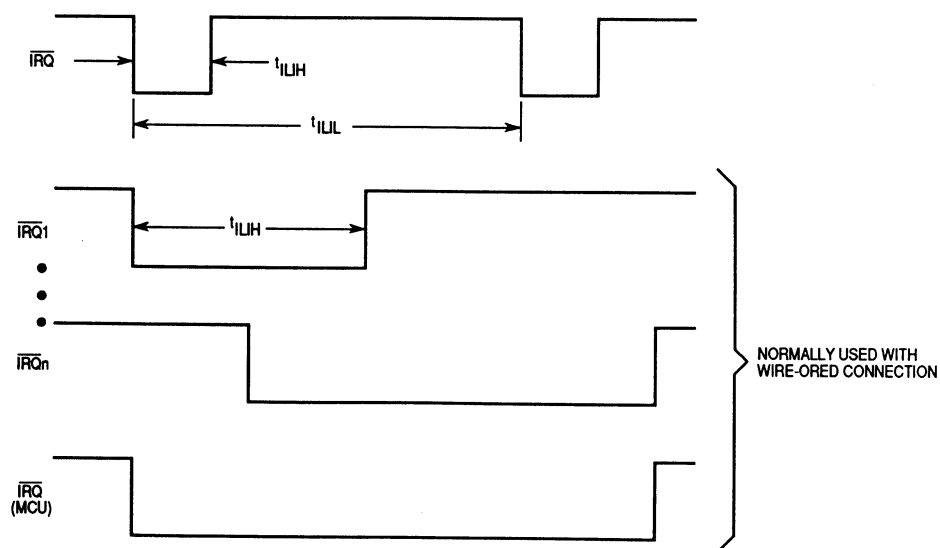


Figure 3-2. Reset and Interrupt Processing Flowchart



(a) Interrupt Internal Function Diagram



(b) Interrupt Mode Diagram

Figure 3-3. External Interrupt

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 the $\overline{\text{IRQ}}$ line. 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, a new external interrupt pulse could be latched and serviced as soon as the I bit is cleared.

3.2.2 Software Interrupt (SWI)

The SWI is an executable instruction that is executed regardless of the state of the I bit in the CCR. SWI is higher priority than other interrupts in the sense that once the SWI opcode is fetched, no other interrupt source can interrupt the SWI. In another sense, SWI is the lowest priority interrupt source because any other enabled interrupt that is pending before the SWI opcode is fetched will be serviced before the SWI. 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.

3.2.3 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 **SECTION 4 TIMER** for more information.

3.2.4 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 status bits in the SCI status register.

3.2.5 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.

3.3 LOW-POWER MODES

The following paragraphs provide a description of the STOP and WAIT modes.

3.3.1 STOP Mode

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 3-4).

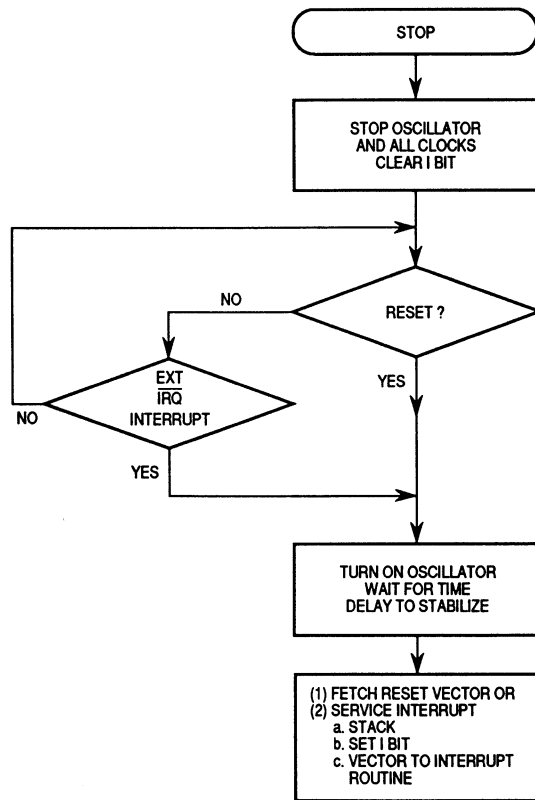


Figure 3-4. STOP Function Flowchart

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.

3.3.1.1 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 $\overline{\text{IRQ}}$ 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.

3.3.1.2 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 $\overline{\text{IRQ}}$ 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 $\overline{\text{IRQ}}$ 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.

3.3.2 WAIT Mode

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 3-5). An interrupt from the timer, SCI, or SPI can cause the MCU to exit the WAIT mode.

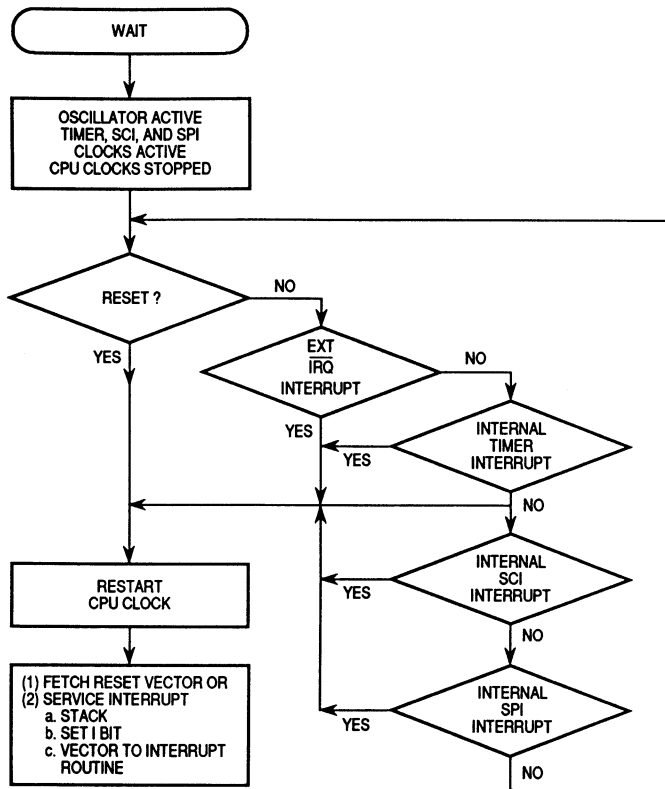


Figure 3-5. WAIT Function Flowchart

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.

3.3.3 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 RESET during data retention mode.

SECTION 4

TIMER

The timer consist of a 16-bit free-running 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 4-1 for a timer block diagram.

Because the timer has a 16-bit architecture, each function is represented by two registers. These registers contain a high and a low byte. 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

In some cases, the I bit in the CCR should be set while manipulating both the high and low byte register of a specific timer function. This is done to ensure that an interrupt does not occur between reads of associated bytes.

4.1 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 μ s if the internal bus clock is 2.0 MHz. 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

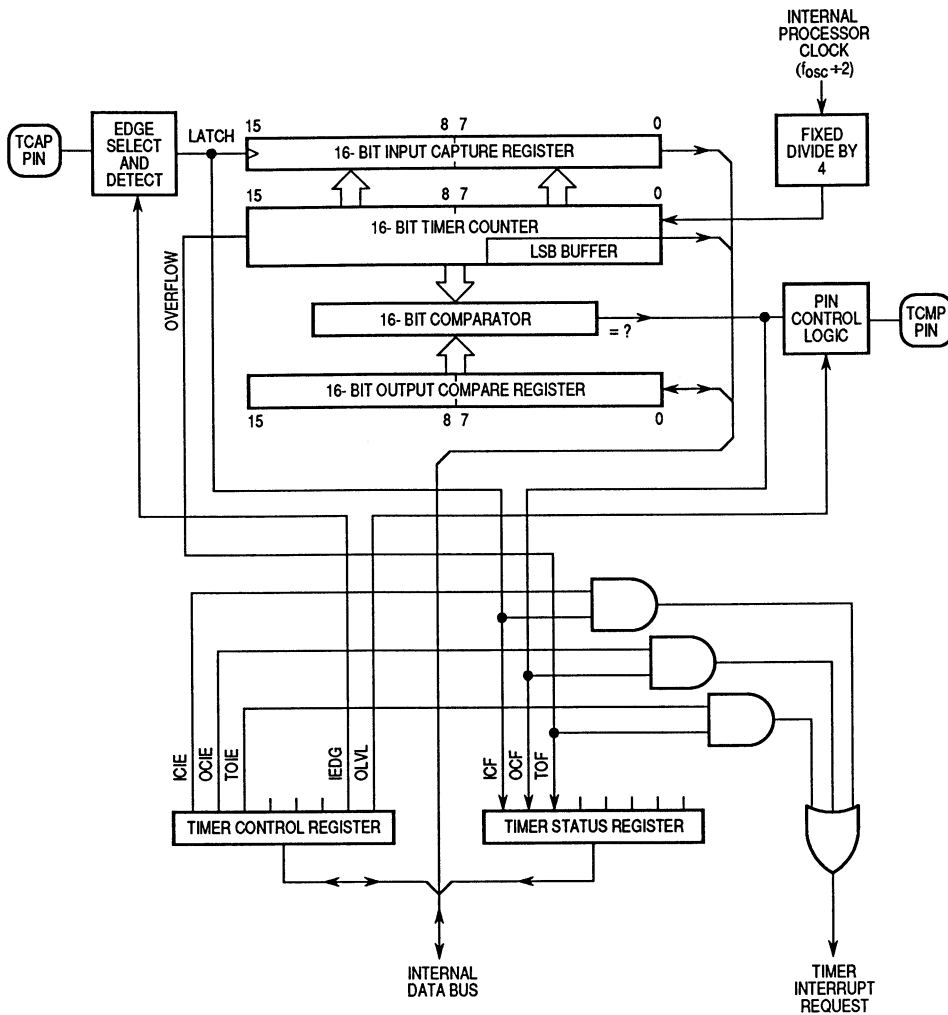


Figure 4-1. Timer Block Diagram

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 unintentional clearing of the TOF.

The free-running counter is preset 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).

4.2 OUTPUT COMPARE REGISTER (OCR) \$0016–\$0017

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 (OLVL) 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.

4.3 INPUT CAPTURE REGISTER (ICR) \$0014–\$0015

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 MSB (\$14), 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.

4.4 TIMER CONTROL REGISTER

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	
\$0012	ICIE	OCIE	TOIE	0	0	0	IEDG	OLVL	TCR
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; these bits always read zero.

4.5 TIMER STATUS REGISTER

The timer status register (TSR) is a read-only register containing three status flag bits.

	7	6	5	4	3	2	1	0	
\$0013	ICF	OCF	TOF	0	0	0	0	0	TSR
RESET:	U	U	U	0	0	0	0	0	

ICF — Input Capture Flag

- 1=Flag set when selected polarity edge is sensed by the input capture edge detector.
- 0=Flag cleared by reading the input capture low register (\$15), after reading TSR while ICF was set.

OCF — Output Compare Flag

- 1=Flag set when output compare register contents matches the free-running counter contents.
- 0=Flag cleared by writing to the output compare low register (\$17), after reading TSR while OCF was set.

TOF — Timer Overflow Flag

- 1=Flag set when free-running counter transition from \$FFFF to \$0000 occurs.
- 0=Flag cleared by reading the free-running counter low register (\$19), after reading TSR while TOF was set.

Bits 4–0 — Not used, these bits 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.

4.6 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.

4.7 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.

SECTION 5

SERIAL COMMUNICATIONS INTERFACE

A full-duplex asynchronous SCI is provided with a standard nonreturn to zero (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 (8- or 9-bit words)
- Separate transmitter and receiver enable bits
- SCI may be interrupt driven
- Four separate interrupt conditions

SCI Receiver features:

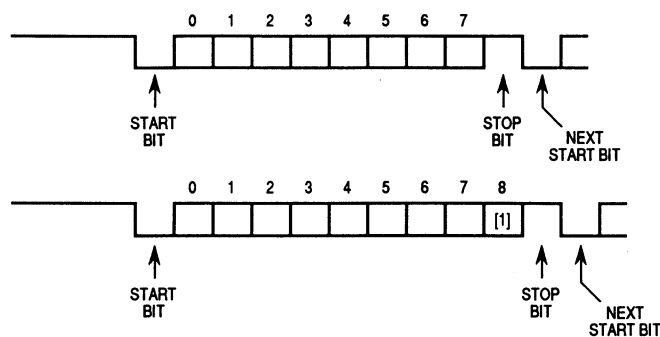
- Receiver wake-up function (idle line or address bit methods)
- Idle line detect
- Framing error detect
- Noise detect
- Overrun detect
- Receiver data register full flag

SCI transmitter features:

- Transmit data register empty flag
- Transmit complete flag
- Break send

5.1 DATA FORMAT

The SCI system communicates via a receive data in (RDI) pin and a transmit data out (TDO) pin. The NRZ data format is as shown in Figure 5-1.



[1] - Control bit "M" selects optional ninth data bit.

Figure 5-1. Data Format

5.2 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 10 (or 11) 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.

5.3 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 Figures 5-2 and 5-3); 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.

5.4 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.

5.5 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.

5.6 FUNCTIONAL DESCRIPTION

A block diagram of the SCI is shown in Figures 5-2 and 5-3. 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

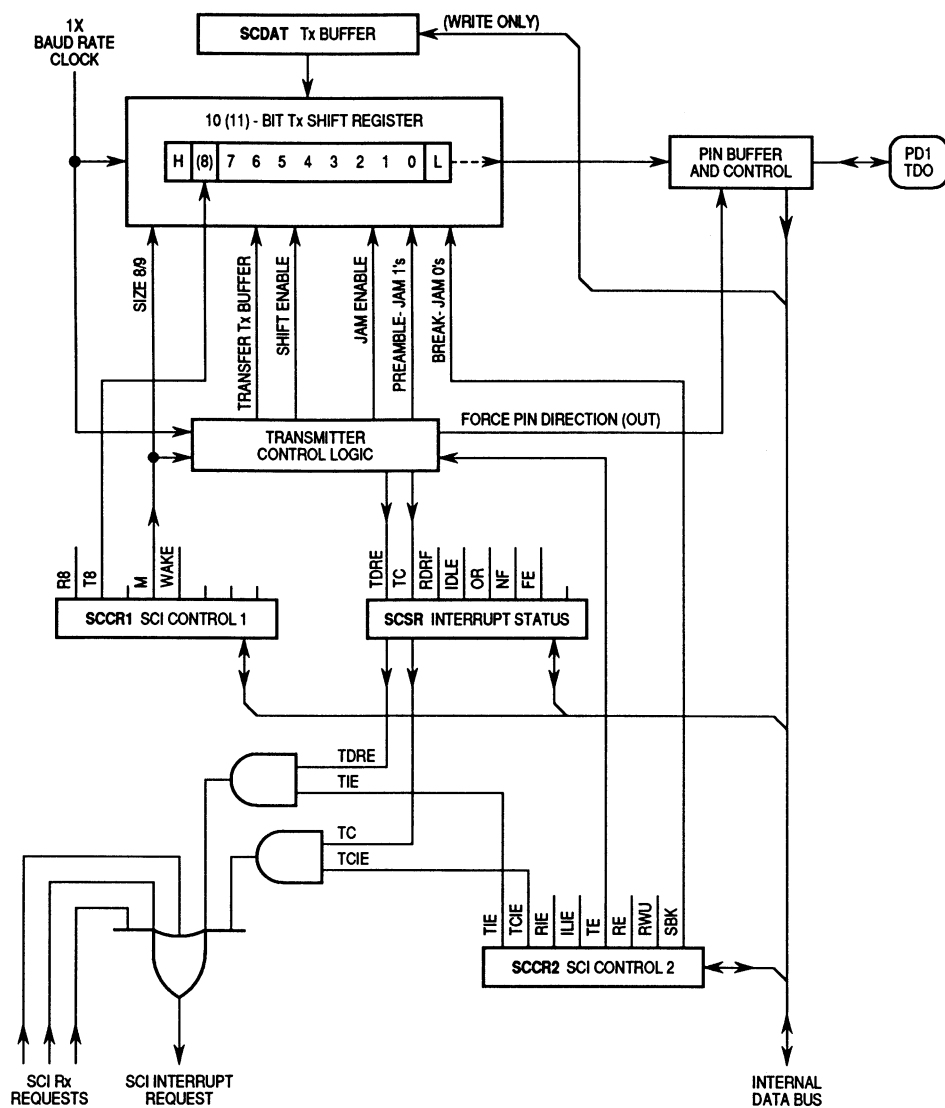


Figure 5-2. SCI Block Transmitter Block Diagram

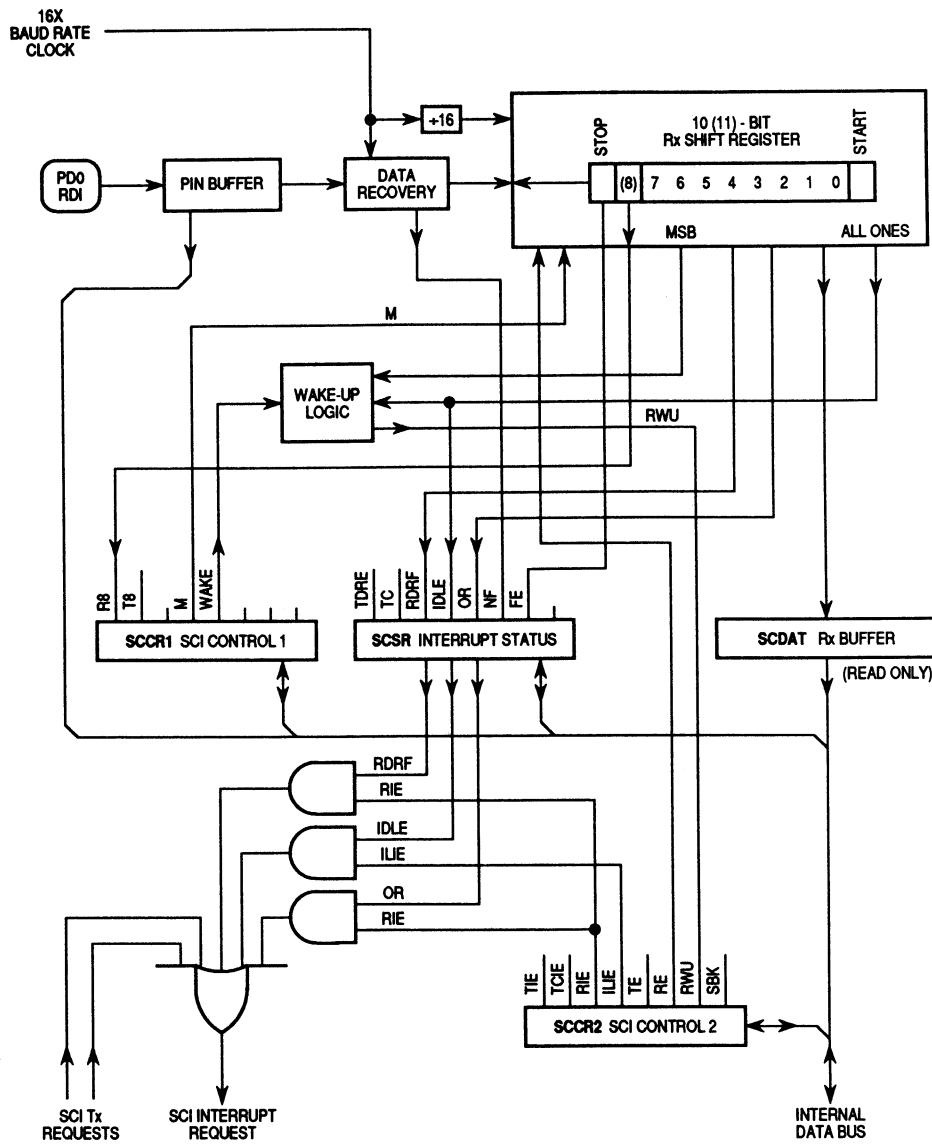


Figure 5-3. SCI Receiver Block Diagram

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 indicates the receive data line is idle) is set. This allows a receiver that is not in the wake-up mode to detect the end of a message, to detect the preamble of a new message, or to resynchronize with the transmitter. After IDLE becomes set, a valid character must be received before another idle line condition can be recognized.

5.7 REGISTERS

There are five registers used in the SCI. The internal configuration of these registers is discussed in the following paragraphs.

5.7.1 Serial Communications Data Register

The SCDAT is a read/write register used to receive and transmit SCI data.

	7	6	5	4	3	2	1	0	
\$0011	SCD7	SCD6	SCD5	SCD4	SCD3	SCD2	SCD1	SCD0	SCDAT
RESET:	U	U	U	U	U	U	U	U	

As shown in Figures 5-2 and 5-3, 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.

5.7.2 Serial Communications Control Register 1

The SCCR1 provides control bits that determine word length and select the wake-up method.

	7	6	5	4	3	2	1	0	
\$000E	R8	T8	—	M	WAKE	—	—	—	SCCR1
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. A one in the most significant bit (the eight or ninth bit, depending on the M bit) of a received character causes the receiver to wake up.
- 0 = Idle line condition causes the receiver to wake up.

Bits 2–0, and 5 — Not used, can read either one or zero.

5.7.3 Serial Communications Control Register 2

The SCCR2 provides control of individual SCI functions such as interrupts, transmit/receive enabling, receiver wake-up, and send break code.

	7	6	5	4	3	2	1	0	
\$000F	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK	SCCR2
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 the transmit shifter completes any serial transfer that was in progress when TE went to zero. After the last byte is transmitted, the TD0 line becomes a high-impedance line.

RE — Receive Enable

1 = The RDI line is applied to the receiver shift register input.

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 = If the WAKE bit = 1, the RWU bit is cleared after receiving a data word with the MSB set. If WAKE = 0, the RWU bit is cleared after receiving 10 (M=0) or 11 (M=11) consecutive ones.

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 almost immediately to the shift register, and the second is queued into the parallel transmit buffer.

5.7.4 Serial Communications Status Register

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	
\$0010	TDRE	TC	RDRF	IDLE	OR	NF	FE	—	SCSR
RESET:	1	1	0	0	0	0	0	—	

TDRE — Transmit Data Register (TDR) Empty

- 1 = TDR is empty and can accept new data to be transmitted.
- 0 = TDR still contains data. TDRE is cleared by reading the SCSR, followed by a write to the TDR.

TC — Transmit Complete

- 1 = Indicates that end of data frame, preamble, or break condition has occurred (transmitter is empty, including shifter).
- 0 = Transmitter is currently active. TC bit cleared by reading the SCSR, followed by a write to the TDR.

RDRF — Receive Data Register (RDR) Full

- 1 = Becomes set when the receive data shift register contents are transferred to the RDR.
- 0 = No significant data is in the RDR. RDRF is cleared by a read of 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 become set until the 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, 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.

5.7.5 Baud Rate Register

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	
\$000D	—	—	SCP1	SCP0	—	SCR2	SCR1	SCR0	BAUD
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 5-1.

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 5-1.

Tables 5-1 and 5-2 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 5-1 represent the final baud rate resulting from the internal processor clock division shown in the divided-by column. Table 5-2 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 (SCP1–SCP0) could be configured as a divide-by-one or as a divide-by-four.

If a divide-by-four prescaler is used, then the SCR2–SCR0 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 SCR2–SCR0 bits configured for a divide-by-eight.

Table 5-1. Prescaler Highest Baud Rate Frequency Output

SCP Bit		Clock* (f _{op}) Divided By	Crystal Frequency (f _{osc}) 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 5-1 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 (see Table 5-2) for some representative prescaler outputs.

Table 5-2. 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 5-2 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.

SECTION 6

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 6-1) or MCUs that can be either masters or slaves.

Features include:

- 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

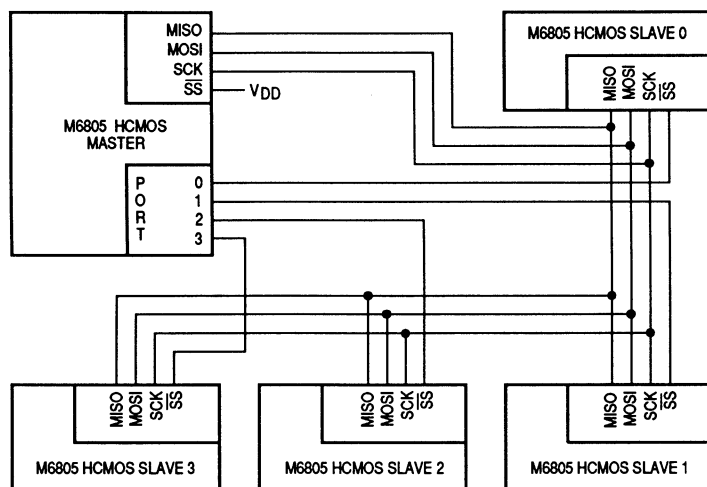


Figure 6-1. Master-Slave System Configuration

6.1 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.

6.1.1 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.

6.1.2 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 ($\overline{SS} = 1$).

6.1.3 Serial Clock

The serial clock (SCK) is used to synchronize data both 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 6-2, 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.

6.1.4 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 of a master goes low, a mode fault error flag (MODF) is set in the serial peripheral status register (SPSR).

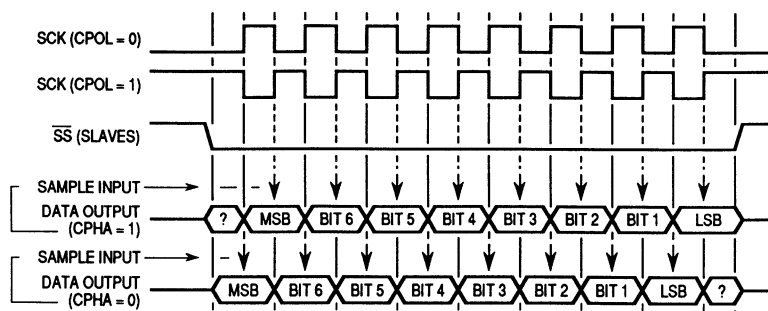


Figure 6-2. Data Clock Timing Diagram

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 $CPHA = 1$, $\overline{SS}$ 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.

6.2 FUNCTIONAL DESCRIPTION

A block diagram of the SPI is shown in Figure 6-3. 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 6-4 illustrates the MOSI, MISO, SCK, and $\overline{SS}$ master-slave interconnections.

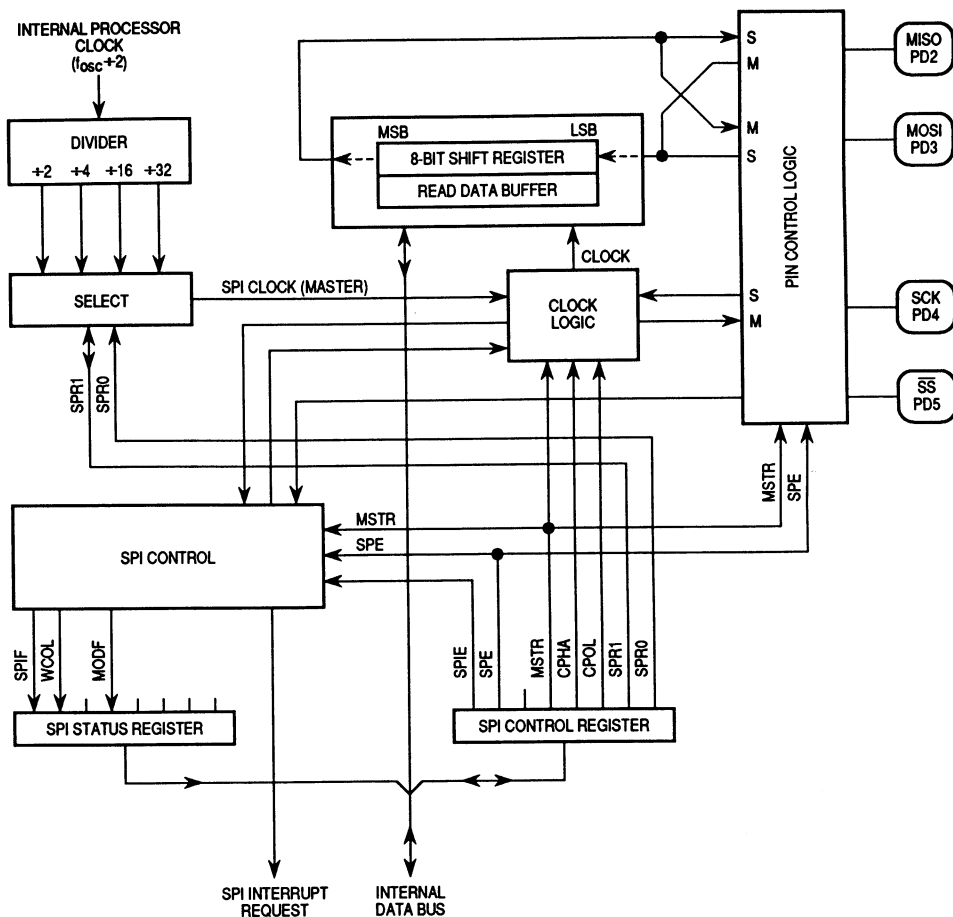


Figure 6-3. SPI Block Diagram

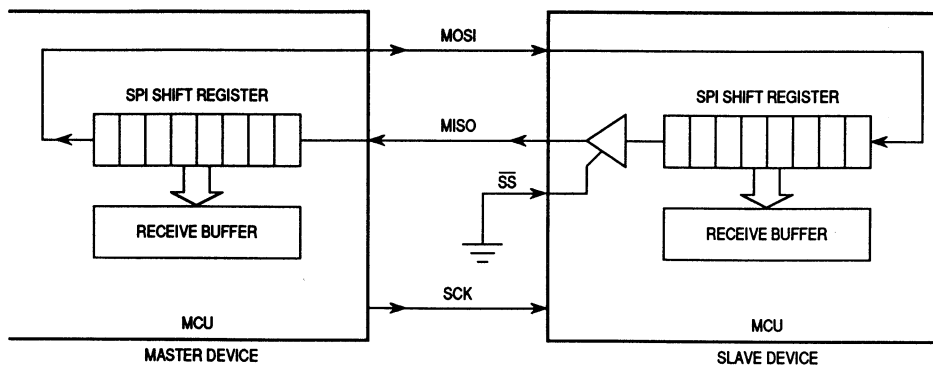


Figure 6-4. SPI Master-Slave Interconnections

6.3 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.

6.3.1 Serial Peripheral Control Register

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	
\$000A	SPIE	SPE	—	MSTR	CPOL	CPHA	SPR1	SPR0	SPCR
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

Bit 5 – Not used, this bit can read either one or zero

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. CPHA selects one of two fundamentally different 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.

SPR1	SPR0	Internal Processor Clock* Divided By
0	0	2
0	1	4
1	0	16
1	1	32

*Internal processor clock equals f_{osc} divided by two.

6.3.2 Serial Peripheral Status Register

The SPSR contains three status bits.

	7	6	5	4	3	2	1	0	
\$000B	SPIF	WCOL	—	MODF	—	—	—	—	SPSR
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 was made to write to SPDR while data transfer was in process.
- 0=Clearing is accomplished by reading SPSR (with WCOL = 1), followed by SPDR access.

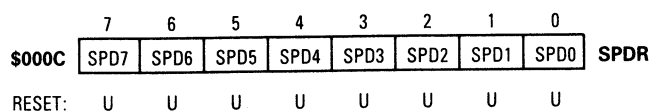
MODF — Mode Fault Flag

- 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 5 and 3-0 — Not used, these bits can read either zero or one.

6.3.3 Serial Peripheral Data I/O Register

The SPDR is a read/write register used to receive and transmit SPI data.



A write to the SPDR places data directly into the shift register for transmission. Only a write to the SPDR register of the master SPI device will initiate the transmission/reception of a byte. On completion of the 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 SPIF status bit must be cleared by the time a new data transfer finishes and data is transferred from the shift register to the read buffer, or an overrun condition will exist. In overrun cases, the new byte is written into the read buffer, whether the previous data was read or not.

SECTION 7

INSTRUCTION SET AND ADDRESS MODES

This section provides a description of the instruction set and address modes.

7.1 INSTRUCTION SET

The MCU has a set of basic instructions that can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type.

Table 7-1 shows all the MC68HC705C8 instructions in all possible addressing modes. For each instruction, the operand construction is shown as well as the number of machine code bytes and the execution time in internal processor clock cycles (t_{CYC}). One internal processor clock cycle equals two oscillator input cycles (f_{OSC}).

This MCU uses all the instructions available in the M146805 CMOS Family plus 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.

Table 7-1. Instructions, Addressing Modes, and Execution Times (Sheet 1 of 4)

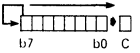
Source Form(s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (hexadecimal)		Bytes	Cycles	Condition Code				
				Opcode	Operand			H	I	N	Z	C
ADC (opr)	Add with Carry	$ACCA \leftarrow ACCA + M + C$	IMM DIR EXT IX2 IX1 IX	A9 B9 C9 D9 E9 F9	ii dd hh ll ee ff ff	2 2 3 3 2 1	2 3 4 5 4 3	+	—	+	+	+
ADD (opr)	Add	$ACCA \leftarrow ACCA + M$	IMM DIR EXT IX2 IX1 IX	AB BB CB DB EB FB	ii dd hh ll ee ff ff	2 2 3 3 2 1	2 3 4 5 4 3	+	—	+	+	+
AND (opr)	Logical AND	$ACCA \leftarrow ACCA \bullet M$	IMM DIR EXT IX2 IX1 IX	A4 B4 C4 D4 E4 F4	ii dd hh ll ee ff ff	2 2 3 3 2 1	2 3 4 5 4 3	—	—	+	+	—
ASL (opr) ASLA ASLX ASL (opr) ASL (opr)	Arithmetic Shift Left		DIR INH(A) INH(X) IX1 IX	38 48 58 68 78	dd ff	2 1 1 2 1	5 3 3 6 5	—	—	+	+	+
ASR (opr) ASRA ASRX ASR (opr) ASR (opr)	Arithmetic Shift Right		DIR INH(A) INH(X) IX1 IX	37 47 57 67 77	dd ff	2 1 1 2 1	5 3 3 6 5	—	—	+	+	+
BCC (rel)	Branch if Carry Clear	? C = 0	REL	24	rr	2	3	—	—	—	—	—
BCLR n, (opr)	Clear Bit n in Memory	$Mn \leftarrow 0$	DIR(b0) DIR(b1) DIR(b2) DIR(b3) DIR(b4) DIR(b5) DIR(b6) DIR(b7)	11 13 15 17 19 1B 1D 1F	dd dd dd dd dd dd dd dd	2 2 2 2 2 2 2 2	5 5 5 5 5 5 5 5	—	—	—	—	—
BCS (rel)	Branch if Carry Set	? C = 1	REL	25	rr	2	3	—	—	—	—	—
BEQ (rel)	Branch if Equal	? Z = 1	REL	27	rr	2	3	—	—	—	—	—
BHCC (rel)	Branch if Half Carry Clear	? H = 0	REL	28	rr	2	3	—	—	—	—	—
BHCS (rel)	Branch if Half Carry Set	? H = 1	REL	29	rr	2	3	—	—	—	—	—
BHI (rel)	Branch if Higher	? (C + Z) = 0	REL	22	rr	2	3	—	—	—	—	—
BHS (rel)	Branch if Higher or Same	? C = 0	REL	24	rr	2	3	—	—	—	—	—
BIH (rel)	Branch if $\overline{IRQ}$ Pin is High	? $\overline{IRQ}$ Pin = 1	REL	2F	rr	2	3	—	—	—	—	—
BIL (rel)	Branch if $\overline{IRQ}$ Pin is Low	? $\overline{IRQ}$ Pin = 0	REL	2E	rr	2	3	—	—	—	—	—
BIT (rel)	Bit Test Memory with A	$ACCA \bullet M$	IMM DIR EXT IX2 IX1 IX	A5 B5 C5 D5 E5 F5	ii dd hh ll ee ff ff	2 2 3 3 2 1	2 3 4 5 4 3	—	—	+	+	—
BLO (rel)	Branch if Lower	? C = 1	REL	25	rr	2	3	—	—	—	—	—
BLS (rel)	Branch if Lower or Same	? (C + Z) = 1	REL	23	rr	2	3	—	—	—	—	—
BMC (rel)	Branch if I Bit is Clear	? I = 0	REL	2C	rr	2	3	—	—	—	—	—

Table 7-1. Instructions, Addressing Modes, and Execution Times (Sheet 2 of 4)

Source Form(s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (hexadecimal)		Bytes	Cycles	Condition Code				
				Opcode	Operand			H	I	N	Z	C
BMI (rel)	Branch if Minus	? N = 1	REL	2B	rr	2	3	—	—	—	—	—
BMS (rel)	Branch if I Bit is Set	? I = 1	REL	2D	rr	2	3	—	—	—	—	—
BNE (rel)	Branch if Not Equal	? Z = 0	REL	26	rr	2	3	—	—	—	—	—
BPL (rel)	Branch if Plus	? N = 0	REL	2A	rr	2	3	—	—	—	—	—
BRA (rel)	Branch Always	? 1 = 1	REL	20	rr	2	3	—	—	—	—	—
BRCLR n, (opr) (rel)	Branch if Bit n of M = 0	? Bit n of M = 0	DIR(b0) DIR(b1) DIR(b2) DIR(b3) DIR(b4) DIR(b5) DIR(b6) DIR(b7)	01 03 05 07 09 0B 0D 0F	dd rr dd rr dd rr dd rr dd rr dd rr dd rr dd rr	3 3 3 3 3 3 3 3	5 5 5 5 5 5 5 5	—	—	—	—	—
BRN (rel)	Branch Never	? 1 = 0	REL	21	rr	2	3	—	—	—	—	—
BRSET n, (opr) (rel)	Branch if Bit n of M = 1	? Bit n of M = 1	DIR(b0) DIR(b1) DIR(b2) DIR(b3) DIR(b4) DIR(b5) DIR(b6) DIR(b7)	00 02 04 06 08 0A 0C 0E	dd rr dd rr dd rr dd rr dd rr dd rr dd rr dd rr	3 3 3 3 3 3 3 3	5 5 5 5 5 5 5 5	—	—	—	—	—
BSET n, (opr)	Set Bit n in Memory	Mn $\neq$ 1	DIR(b0) DIR(b1) DIR(b2) DIR(b3) DIR(b4) DIR(b5) DIR(b6) DIR(b7)	10 12 14 16 18 1A 1C 1E	dd dd dd dd dd dd dd dd	2 2 2 2 2 2 2 2	5 5 5 5 5 5 5 5	—	—	—	—	—
BSR (rel)	Branch to Subroutine	PC $\neq$ PC + 0002 (SP) $\neq$ PCL; SP $\neq$ SP - 0001 (SP) $\neq$ PCH; SP $\neq$ SP - 0001 PC $\neq$ PC + Rel	REL	AD	rr	2	6	—	—	—	—	—
CLC	Clear C Bit	C bit $\neq$ 0	INH	98		1	2	—	—	—	—	0
CLI	Clear I Bit	I bit $\neq$ 0	INH	9A		1	2	—	0	—	—	—
CLR (opr) CLRA CLR X CLR (opr) CLR (opr)	Clear	M $\neq$ 00 A $\neq$ 00 X $\neq$ 00 M $\neq$ 00 M $\neq$ 00	DIR INH(A) INH(X) IX1 IX	3F 4F 5F 6F 7F	dd ff ff ff ff	2 1 1 2 1	5 3 3 6 5	—	—	0	1	—
CMP (opr)	Compare A with Memory	ACCA - M	IMM DIR EXT IX2 IX1 IX	A1 B1 C1 D1 E1 F1	ii dd hh ll ee ff ff ff	2 2 3 3 2 1	2 3 4 5 4 3	—	—	—	—	—
COM (opr) COMA COM X COM (opr) COM (opr)	1's Complement	M $\neq$ $\overline{M}$ = \$FF - M A $\neq$ $\overline{A}$ = \$FF - A X $\neq$ $\overline{X}$ = \$FF - X M $\neq$ $\overline{M}$ = \$FF - M M $\neq$ $\overline{M}$ = \$FF - M	DIR INH(A) INH(X) IX1 IX	33 43 53 63 73	dd ff ff ff ff	2 1 1 2 1	5 3 3 6 5	—	—	—	—	1
CPX (opr)	Compare X with Memory	X - M	IMM DIR EXT IX2 IX1 IX	A3 B3 C3 D3 E3 F3	ii dd hh ll ee ff ff ff	2 2 3 3 2 1	2 3 4 5 4 3	—	—	—	—	—

Table 7-1. Instructions, Addressing Modes, and Execution Times (Sheet 3 of 4)

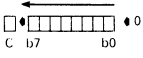
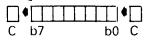
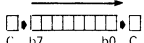
Source Form(s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (hexadecimal)		Bytes	Cycles	Condition Code				
				Opcode	Operand			H	I	N	Z	C
DEC (opr) DECA DECX DEC (opr) DEC (opr)	Decrement DEX (same as DECX)	$M \leftarrow M - 01$ $A \leftarrow A - 01$ $X \leftarrow X - 01$ $M \leftarrow M - 01$ $M \leftarrow M - 01$	DIR INH(A) INH(X) IX1 IX	3A 4A 5A 6A 7A	dd ff	2 1 1 2 1	5 3 3 6 5	—	—	—	—	—
EOR (opr)	Exclusive OR A with Memory	$ACCA \leftarrow ACCA \oplus M$	IMM DIR EXT IX2 IX1 IX	A8 B8 C8 D8 E8 F8	ii dd hh ll ee ff ff	2 2 3 3 2 1	2 3 4 5 4 3	—	—	—	—	—
INC (opr) INCA INCX INC (opr) INC (opr)	Increment INX (same as INCX)	$M \leftarrow M + 01$ $A \leftarrow A + 01$ $X \leftarrow X + 01$ $M \leftarrow M + 01$ $M \leftarrow M + 01$	DIR INH(A) INH(X) IX1 IX	3C 4C 5C 6C 7C	dd ff	2 1 1 2 1	5 3 3 6 5	—	—	—	—	—
JMP (opr)	Jump	$PC \leftarrow$ effective address	DIR EXT IX2 IX1 IX	BC CC DC EC FC	dd hh ll ee ff ff	2 3 3 2 1	2 3 4 3 2	—	—	—	—	—
JSR (opr)	Jump to Subroutine	$PC \leftarrow PC + n$ (n = 1, 2, or 3) (SP) $\leftarrow$ PCL; SP $\leftarrow$ SP - 0001 (SP) $\leftarrow$ PCH; SP $\leftarrow$ SP - 0001 $PC \leftarrow$ effective address	DIR EXT IX2 IX1 IX	BD CD DD ED FD	dd hh ll ee ff ff	2 3 3 2 1	5 6 7 6 5	—	—	—	—	—
LDA (opr)	Load A from Memory	$ACCA \leftarrow M$	IMM DIR EXT IX2 IX1 IX	A6 B6 C6 D6 E6 F6	ii dd hh ll ee ff ff	2 2 3 3 2 1	2 3 4 5 4 3	—	—	—	—	—
LDX (opr)	Load X from Memory	$X \leftarrow M$	IMM DIR EXT IX2 IX1 IX	AE BE CE DE EE FE	ii dd hh ll ee ff ff	2 2 3 3 2 1	2 3 4 5 4 3	—	—	—	—	—
LSL (opr) LSLA LSLX LSL (opr) LSL (opr)	Logical Shift Left		DIR INH(A) INH(X) IX1 IX	38 48 58 68 78	dd ff	2 1 1 2 1	5 3 3 6 5	—	—	—	—	—
LSR (opr) LSRA LSRX LSR (opr) LSR (opr)	Logical Shift Right		DIR INH(A) INH(X) IX1 IX	34 44 54 64 74	dd ff	2 1 1 2 1	5 3 3 6 5	—	—	0	—	—
MUL	Unsigned Multiply	$X:A \leftarrow X \times A$	INH	42		1	11	0	—	—	—	0
NEG (opr) NEGA NEGX NEG (opr) NEG (opr)	Negate (2's Complement)	$M \leftarrow -M$ (i.e. $00 - M$) $A \leftarrow -A$ $X \leftarrow -X$ $M \leftarrow -M$ $M \leftarrow -M$	DIR INH(A) INH(X) IX1 IX	30 40 50 60 70	dd ff	2 1 1 2 1	5 3 3 6 5	—	—	—	—	—
NOP	No Operation		INH	9D		1	2	—	—	—	—	—
ORA (opr)	Inclusive OR	$ACCA \leftarrow ACCA + M$	IMM DIR EXT IX2 IX1 IX	AA BA CA DA EA FA	ii dd hh ll ee ff ff	2 2 3 3 2 1	2 3 4 5 4 3	—	—	—	—	—

Table 7-1. Instructions, Addressing Modes, and Execution Times (Sheet 4 of 4)

Source Form(s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (hexadecimal)		Bytes	Cycles	Condition Code				
				Opcode	Operand			H	I	N	Z	C
ROL (opr) ROLA ROLX ROL (opr) ROL (opr)	Rotate Left through Carry		DIR INH(A) INH(X) IX1 IX	39 49 59 69 79	dd ff 	2 1 1 2 1	5 3 3 6 5	—	—	—	—	—
ROR (opr) RORA RORX ROR (opr) ROR (opr)	Rotate Right through Carry		DIR INH(A) INH(X) IX1 IX	36 46 56 66 76	dd ff 	2 1 1 2 1	5 3 3 6 5	—	—	—	—	—
RSP	Reset Stack Pointer	SP ← \$00FF	INH	9C		1	2	—	—	—	—	—
RTI	Return from Interrupt	SP ← SP + 0001; CC ← (SP) SP ← SP + 0001; ACCA ← (SP) SP ← SP + 0001; X ← (SP) SP ← SP + 0001; PCH ← (SP) SP ← SP + 0001; PCL ← (SP)	INH	80		1	9	(From Stack)				
RTS	Return from Subroutine	SP ← SP + 0001; PCH ← (SP) SP ← SP + 0001; PCL ← (SP)	INH	81		1	6	—	—	—	—	—
SBC (opr)	Subtract with Carry	ACCA ← ACCA – M – C	IMM DIR EXT IX2 IX1 IX	A2 B2 C2 D2 E2 F2	ii dd hh ll ee ff ff	2 2 3 3 2 1	2 3 4 5 4 3	—	—	—	—	—
SEC	Set C Bit	C bit ← 1	INH	99		1	2	—	—	—	—	1
SEI	Set I Bit	I bit ← 1	INH	9B		1	2	—	1	—	—	—
STA (opr)	Store A in Memory	M ← ACCA	DIR EXT IX2 IX1 IX	B7 C7 D7 E7 F7	dd hh ll ee ff ff	2 3 3 2 1	4 5 6 5 4	—	—	—	—	—
STOP	Enable $\overline{IRQ}$, Stop Oscillator		INH	8E		1	2	—	0	—	—	—
STX (opr)	Store X in Memory	M ← X	DIR EXT IX2 IX1 IX	BF CF DF EF FF	dd hh ll ee ff ff	2 3 3 2 1	4 5 6 5 4	—	—	—	—	0
SUB (opr)	Subtract	ACCA ← ACCA – M	IMM DIR EXT IX2 IX1 IX	A0 B0 C0 D0 E0 F0	ii dd hh ll ee ff ff	2 2 3 3 2 1	2 3 4 5 4 3	—	—	—	—	—
SWI	Software Interrupt	PC ← PC + 0001 (SP) ← PCL; SP ← SP – 0001 (SP) ← PCH; SP ← SP – 0001 (SP) ← X; SP ← SP – 0001 (SP) ← ACCA; SP ← SP – 0001 (SP) ← CC; SP ← SP – 0001 I bit ← 1 PCH ← \$xFFC (vector PCL ← \$xFFD fetch)	INH	83		1	10	—	1	—	—	—
TAX	Transfer A to X	X ← ACCA	INH	97		1	2	—	—	—	—	—
TST (opr) TSTA TSTX TST (opr) TST (opr)	Test for Negative or Zero	M – 0	DIR INH(A) INH(X) IX1 IX	3D 4D 5D 6D 7D	dd ff 	2 1 1 2 1	4 3 3 5 4	—	—	—	—	0
TXA	Transfer X to A	ACCA ← X	INH	9F		1	2	—	—	—	—	—
WAIT	Enable Interrupts, Halt CPU		INH	8F		1	2	—	0	—	—	—

7.1.1 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

7.1.2 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

7.1.3 Branch Instructions

This set of instructions branch 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

7.1.4 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, 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)

7.1.5 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

7.1.6 Opcode Map Summary

Table 7-2 is an opcode map for the instructions used on the MCU.

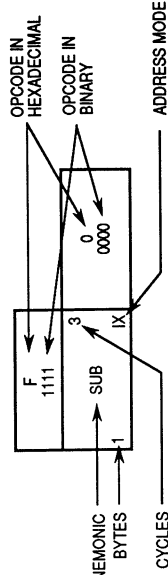
Table 7-2. Opcode Map

HI	Bit Manipulation			Branch			Read/Modify/Write			Control			Register/Memory			IX
	BTB	BTB	BSC	REL	DIR	INH	4	5	6	7	8	9	IMM	DIR	EXT	
LOW	0	0000	BSC	2	0010	3	0011	4	5	6	7	8	9	10	11	LOW
0	BRSET0	BTB 2	BSC 2	BRA	REL 2	3	NEG	DIR 1	5	6	NEG	5	RTI	INH	5	0
1	BRCLR0	BTB 2	BSC 2	BRN	REL 2	3						1	RTS	INH	5	1
2	BRSET1	BTB 2	BSC 2	BHI	REL 2	3										2
3	BRCLR1	BTB 2	BSC 2	BLS	REL 2	3	COM	5	3	6	COM	5				3
4	BRSET2	BTB 2	BSC 2	BCC	REL 2	3	LSR	DIR 1	5	6	LSR	5				4
5	BRCLR2	BTB 2	BSC 2	BCS	REL 2	3										5
6	BRSET3	BTB 2	BSC 2	BNE	REL 2	3	ROR	DIR 1	5	6	ROR	5				6
7	BRCLR3	BTB 2	BSC 2	BEQ	REL 2	3	ASR	DIR 1	5	6	ASR	5				7
8	BRSET4	BTB 2	BSC 2	BHCC	REL 2	3	LSL	DIR 1	5	6	LSL	5				8
9	BRCLR4	BTB 2	BSC 2	BHCS	REL 2	3	ROL	DIR 1	5	6	ROL	5				9
A	BRSET5	BTB 2	BSC 2	BPL	REL 2	3	DEC	DIR 1	5	6	DEC	5				A
B	BRCLR5	BTB 2	BSC 2	BMI	REL 2	3										B
C	BRSET6	BTB 2	BSC 2	BMC	REL 2	3	INC	DIR 1	5	6	INC	5				C
D	BRCLR6	BTB 2	BSC 2	BMS	REL 2	3	TST	DIR 1	5	6	TST	5				D
E	BRSET7	BTB 2	BSC 2	BIL	REL 2	3										E
F	BRCLR7	BTB 2	BSC 2	BIH	REL 2	3	CLR	DIR 1	5	6	CLR	5				F
1111																1111

Abbreviations for Address Modes

INH	Inherent	BSC	Bit Set/Clear
A	Accumulator	BTB	Bit Test and Branch
X	Index Register	IX	Indexed (No Offset)
IMM	Immediate	IX1	Indexed, 1 Byte (8-Bit) Offset
DIR	Direct	IX2	Indexed, 2 Byte (16-Bit) Offset
EXT	Extended		
REL	Relative		

Legend



7.2 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.

7.2.1 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).

7.2.2 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.

7.2.3 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 modes 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.

7.2.4 Relative

The relative addressing mode is only used in branch instructions. In relative addressing, the content of the 8-bit signed byte following the opcode (the offset) 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 -127 to $+128$ locations (with respect to the address of the instruction immediately following the branch instruction). The programmer need not calculate the offset when using the Motorola assembler, since it calculates the proper offset and checks to see if it is within the span of the branch.

7.2.5 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.

7.2.6 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 which can be accessed in this way.

7.2.7 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.

7.2.8 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.

7.2.9 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.

7.2.10 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.

SECTION 8

ELECTRICAL SPECIFICATIONS

This section contains the electrical specifications and associated timing information for the MC68HC705C8.

8.1 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
Programming Voltage	V _{PP}	V _{DD} − 0.3 to 16.0	V
Bootstrap Mode (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 MC68HC705C8P, FN, S MC68HC705C8CP, CFN, CS	T _A	T _L to T _H 0 to +70 −40 to +85	°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} ≤ (V_{in} or V_{out}) ≤ 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}).

8.2 THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance	θ _{JA}		°C/W
Plastic		60	
Cerdip		60	
Plastic Leaded Chip Carrier (PLCC)		70	

8.3 POWER CONSIDERATIONS

The average chip-junction temperature, T_J , in $^{\circ}\text{C}$ can be obtained from:

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

where:

T_A = Ambient Temperature, $^{\circ}\text{C}$

θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, $^{\circ}\text{C}/\text{W}$

$P_D = P_{INT} + P_{I/O}$

$P_{INT} = I_{DD} \times V_{DD}$, 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):

$$P_D = K \div (T_J + 273^{\circ}\text{C}) \quad (2)$$

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

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

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 .

$V_{DD} = 4.5 \text{ V}$

Pins	R1	R2	C
PA7-PA0, PB7-PB0, PC7-PC0, PD4-PD1	3.26 k Ω	2.38 k Ω	50 pF
PD7, PD5, PD0	1.9 k Ω	2.26 k Ω	200 pF

$V_{DD} = 3.0 \text{ V}$

Pins	R1	R2	C
PA7-PA0, PB7-PB0, PC7-PC0, PD4-PD1	10.91 k Ω	6.32 k Ω	50 pF
PD7, PD5, PD0	6 k Ω	6 k Ω	200 pF

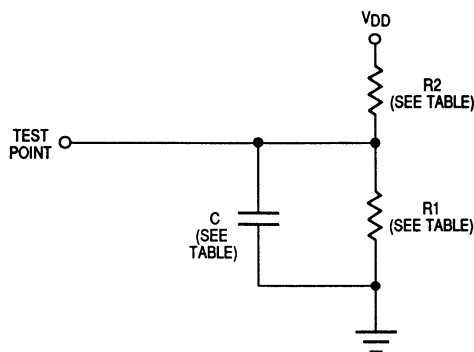


Figure 8-1. Equivalent Test Load

8.4 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 8-2) (I _{Load} = 1.6 mA) PD1–PD4 (see Figure 8-3)	V _{OH}	V _{DD} - 0.8 V _{DD} - 0.8	— —	— —	V
Output Low Voltage (see Figure 8-4) (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, $\overline{\text{IRQ}}$, RESET, OSC1	V _{IH}	0.7 × V _{DD}	—	V _{DD}	V
Input Low Voltage PA0–PA7, PB0–PB7, PC0–PC7, PD0–PD5, PD7, TCAP, $\overline{\text{IRQ}}$, RESET, OSC1	V _{IL}	V _{SS}	—	0.2 × V _{DD}	V
EPROM Programming Voltage	V _{PP}	14.5	14.75	15.0	V
EPROM/OTPROM Programming Current	I _{PP}	—	5	10	mA
User Mode Current	I _{PP}	—	—	± 10	mA
Data Retention Mode (0° to 70°C)	V _{RM}	2.0	—	—	V
Supply Current (see Notes) Run Wait Stop	I _{DD}	— — —	4.7 1.7 2.0	7.0 3.0 50	mA mA μA
I/O Ports Hi-Z Leakage Current PA0–PA7, PB0–PB7, PC0–PC7, PD1–PD4	I _{IL}	—	—	± 10	μA
Input Current $\overline{\text{IRQ}}$, TCAP, OSC1, PD0, PD5, PD7	I _{in}	—	—	± 1	μA
Capacitance Ports (as Input or Output) RESET, $\overline{\text{IRQ}}$, TCAP, PD0–PD5, PD7	C _{out} C _{in}	— —	— —	12 8	pF

NOTES:

1. Typical values at midpoint of voltage range, 25°C only.
2. Wait I_{DD}: Only timer system active (SPE = TE = RE = 0). If SPI, SCI active (SPE = TE = RE = 1) add 10% current draw.
3. 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.
4. Wait, Stop I_{DD}: All ports configured as inputs, V_{IL} = 0.2 V, V_{IH} = V_{DD} - 0.2 V.
5. Stop I_{DD} measured with OSC1 = V_{SS}.
6. Standard temperature range is 0° to 70°C. Extended temperature (-40° to +85°C) is available.
7. Wait I_{DD} is affected linearly by the OSC2 capacitance.

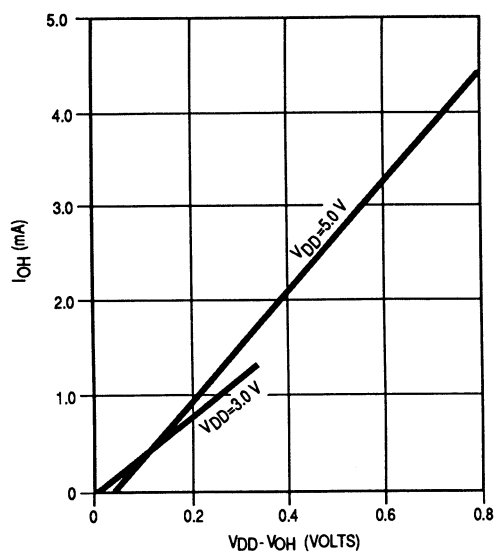
8.5 DC ELECTRICAL CHARACTERISTICS

($V_{DD} = 3.3 \text{ Vdc} \pm 0.3 \text{ Vdc}$, $V_{SS} = 0 \text{ Vdc}$, $T_A = T_L$ to T_H , unless otherwise noted)

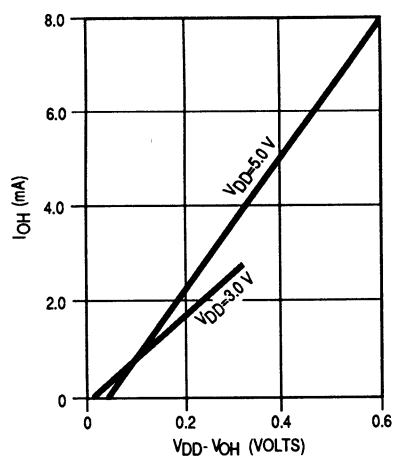
Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage, $I_{Load} \leq 10.0 \mu\text{A}$	V_{OL} V_{OH}	— $V_{DD} - 0.1$	— —	0.1 —	V
Output High Voltage ($I_{Load} = 0.2 \text{ mA}$) PA0–PA7, PB0–PB7, PC0–PC7, TCMP (see Figure 8-2) ($I_{Load} = 0.4 \text{ mA}$) PD1–PD4 (see Figure 8-3)	V_{OH}	$V_{DD} - 0.3$ $V_{DD} - 0.3$	— —	— —	V
Output Low Voltage (see Figure 8-4) ($I_{Load} = 0.4 \text{ mA}$) 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 Wait Stop	I_{DD}	— — —	1.9 0.43 0.84	3.0 1.0 20	mA mA μA
I/O Ports Hi-Z Leakage Current PA0–PA7, PB0–PB7, PC0–PC7, PD1–PD4	I_{IL}	—	—	± 10	μA
Input Current RESET, $\overline{IRQ}$, TCAP, OSC1, PD0, PD5, PD7	I_{in}	—	—	± 1	μA

NOTES:

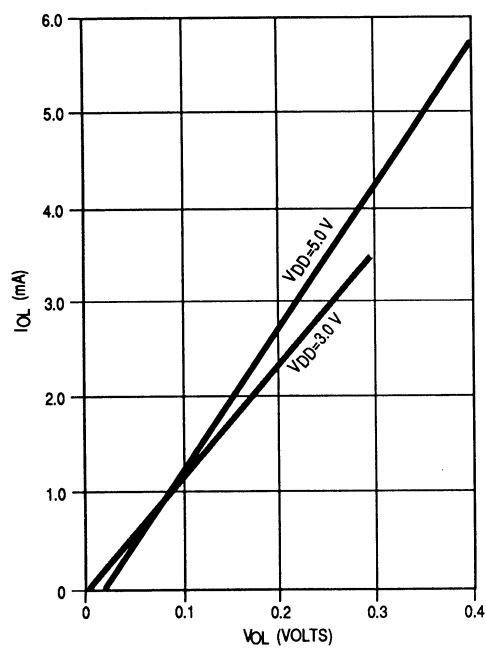
1. Typical values at midpoint of voltage range, 25°C only.
2. Wait I_{DD} : Only timer system active ($SPE = TE = RE = 0$). If SPI, SCI active ($SPE = TE = RE = 1$) add 10% current draw.
3. Run (Operating) I_{DD} , Wait I_{DD} : Measured using external square wave clock source ($f_{OSC} = 4.2 \text{ MHz}$), all inputs 0.2 V from rail; no dc loads, less than 50 pF on all outputs, $C_L = 20 \text{ pF}$ on OSC2.
4. Wait, Stop I_{DD} : All ports configured as inputs, $V_{IL} = 0.2 \text{ V}$, $V_{IH} = V_{DD} - 0.2 \text{ V}$.
5. Stop I_{DD} measured with OSC1 = V_{SS} .
6. Standard temperature range is 0° to 70°C . Extended temperature (-40° to $+85^\circ\text{C}$) is available.
7. Wait I_{DD} is affected linearly by the OSC2 capacitance.



(a) V_{OH} vs I_{OH} for Ports A, B, C, and TCMP

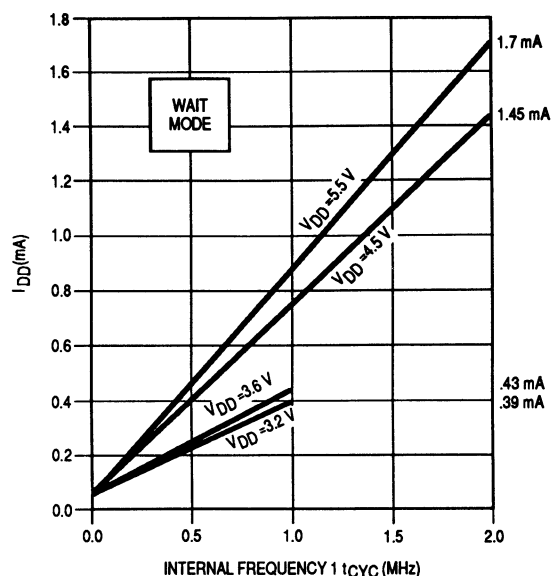


(b) V_{OH} vs I_{OH} for PD1-PD4

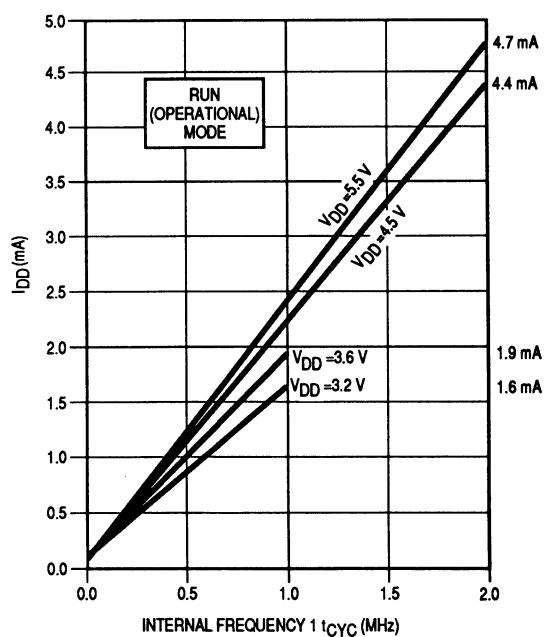


(c) V_{OL} vs I_{OL} for All Ports

Figure 8-2. Typical Voltage Compared to Current



(a) WAIT Mode



(b) RUN Mode

Figure 8-3. Typical Current vs Internal Frequency for RUN and WAIT Modes

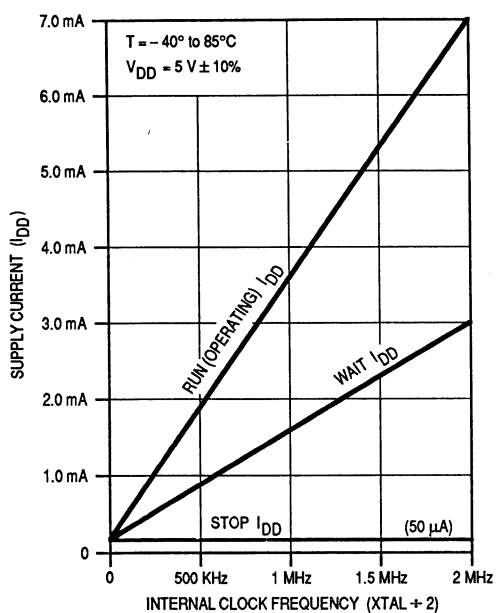
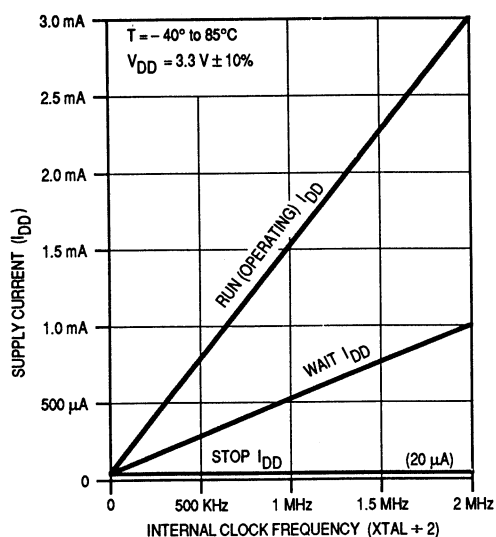


Figure 8-4. Total Current Drain vs Frequency

8.6 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 8-8)	t _{cyc}	480	—	ns
Crystal Oscillator Startup Time (see Figure 8-8)	t _{OXOV}	—	100	ms
Stop Recovery Startup Time (Crystal Oscillator) (see Figure 8-5)	t _{ILCH}	—	100	ms
RESET Pulse Width (see Figure 8-8)	t _{RL}	8	—	t _{cyc}
Timer Resolution** Input Capture Pulse Width (see Figure 8-6) Input Capture Pulse Period (see Figure 8-6)	t _{RESL} t _{TH} , t _{TL} t _{TLTL}	4.0 125 ***	— — —	t _{cyc} ns t _{cyc}
Interrupt Pulse Width Low (Edge-Triggered) (see Figure 3-3b)	t _{LIH}	125	—	ns
Interrupt Pulse Period (see Figure 3-3b)	t _{LIL}	*	—	t _{cyc}
OSC1 Pulse Width	t _{OH} , t _{OL}	90	—	ns

*The minimum period t_{LIL} 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}.

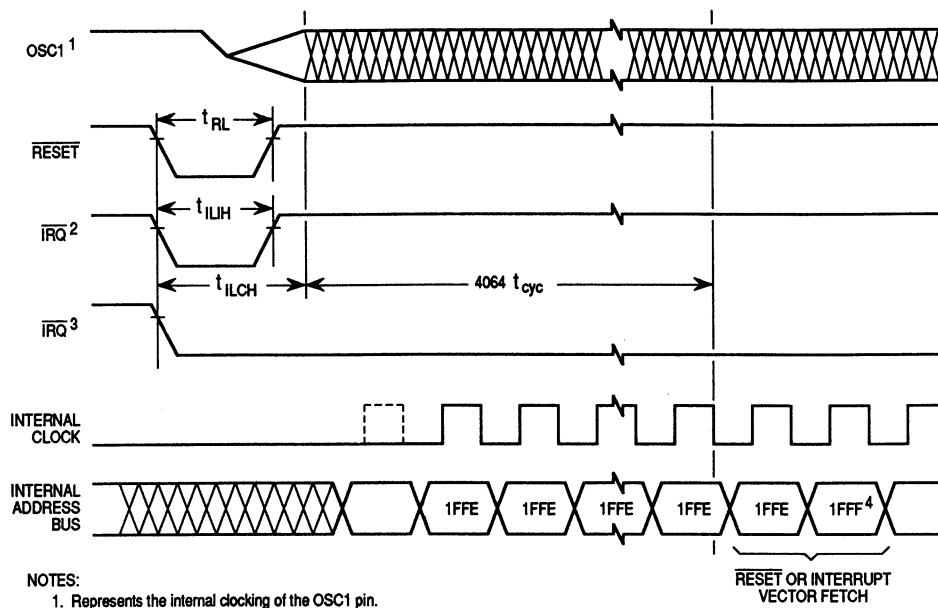


Figure 8-5. Stop Recovery Timing Diagram

8.7 CONTROL TIMING

($V_{DD}=3.3\text{ Vdc} \pm 0.3\text{ Vdc}$, $V_{SS}=0\text{ Vdc}$, $T_A=T_L$ 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 8-8)	t_{cyc}	1000	—	ns
Crystal Oscillator Startup Time (see Figure 8-8)	t_{OXOV}	—	100	ms
Stop Recovery Startup Time (Crystal Oscillator) (see Figure 8-5)	t_{ILCH}	—	100	ms
$\overline{RESET}$ Pulse Width — Excluding Powerup (see Figure 8-8)	t_{RL}	8	—	t_{cyc}
Timer Resolution** Input Capture Pulse Width (see Figure 8-6) Input Capture Pulse Period (see Figure 8-6)	t_{RESL} t_{TH}, t_{TL} t_{TLTL}	4.0 250 ***	— — —	t_{cyc} ns t_{cyc}
Interrupt Pulse Width Low (Edge-Triggered) (see Figure 3-3b)	t_{LILH}	250	—	ns
Interrupt Pulse Period (see Figure 3-3b)	t_{LIL}	*	—	t_{cyc}
OSC1 Pulse Width	t_{OH}, t_{OL}	200	—	ns

*The minimum period t_{LIL} 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} .

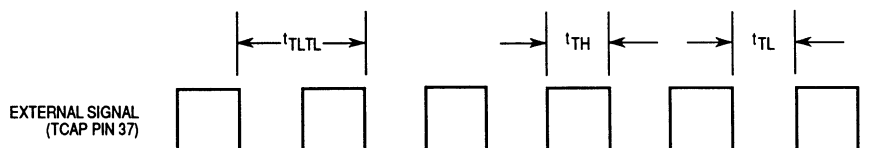


Figure 8-6. Timer Relationships

8.8 SERIAL PERIPHERAL INTERFACE (SPI) TIMING

(V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = T_L to T_H) (see Figure 8-7)

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.

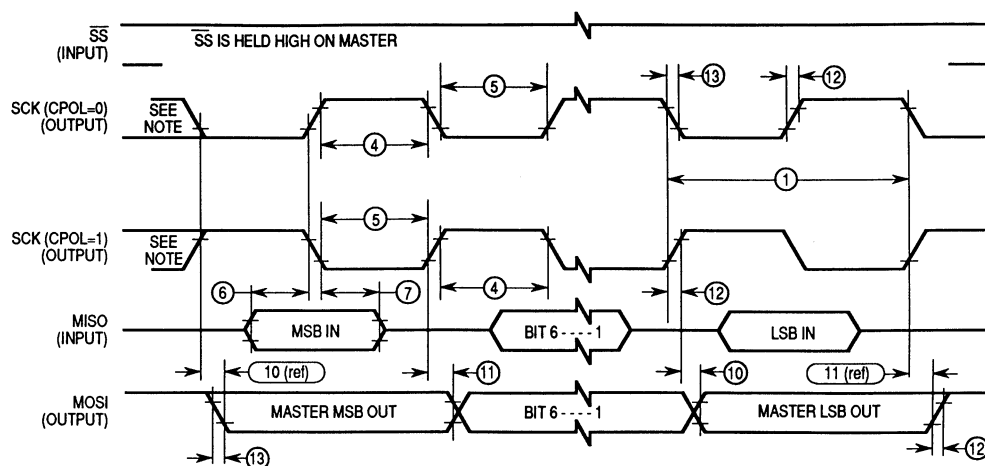
8.9 SERIAL PERIPHERAL INTERFACE (SPI) TIMING

($V_{DD} = 3.3 \text{ Vdc} \pm 10\%$, $V_{SS} = 0 \text{ Vdc}$, $T_A = T_L$ to T_H) (see Figure 8-7)

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} , $C_L = 200 \text{ pF}$) 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} , $C_L = 200 \text{ pF}$) SPI Outputs (SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and SS)	t_{fm} t_{fs}	— —	200 2.0	ns μs

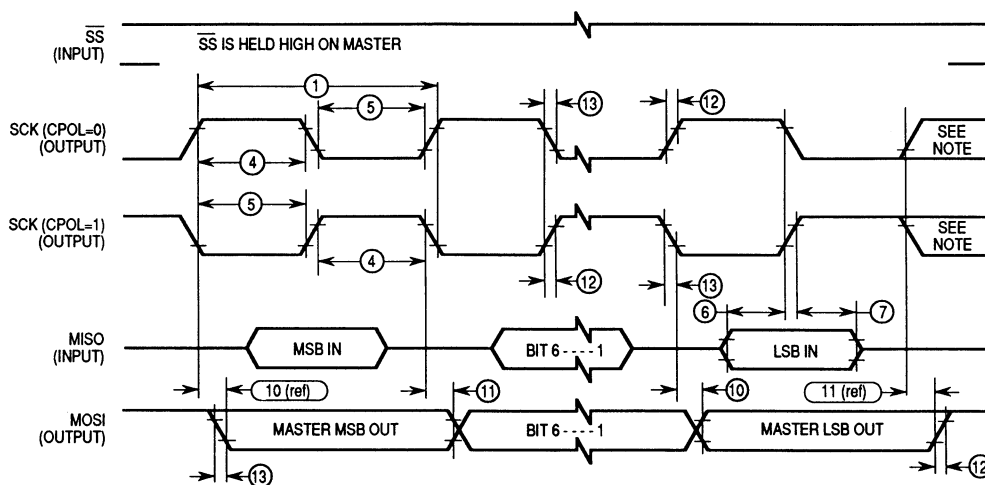
*Signal production depends on software.

**Assumes 200 pF load on all SPI pins.



NOTE: This first clock edge is generated internally but is not seen at the SCK pin.

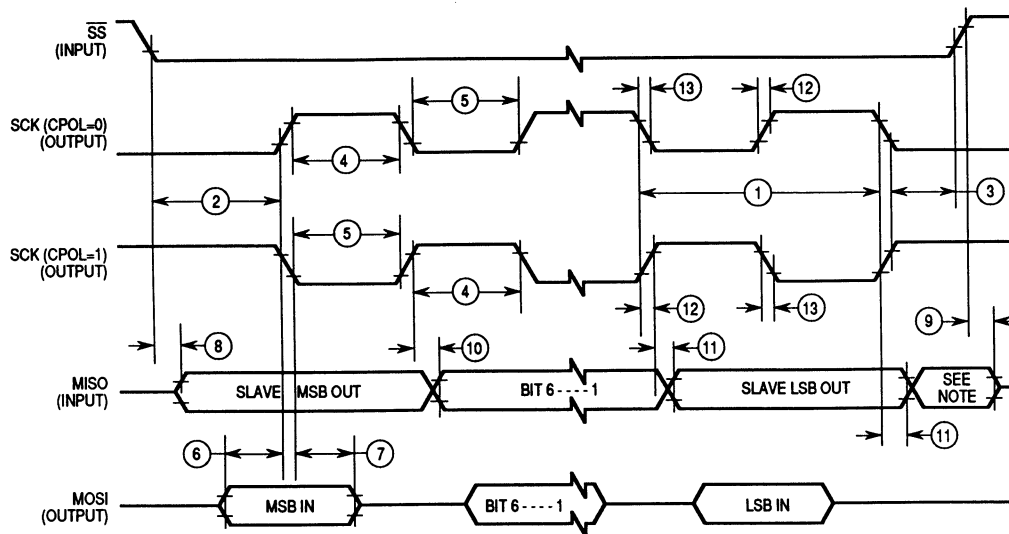
(a) SPI Master Timing (CPHA = 0)



NOTE: This last clock edge is generated internally but is not seen at the SCK pin.

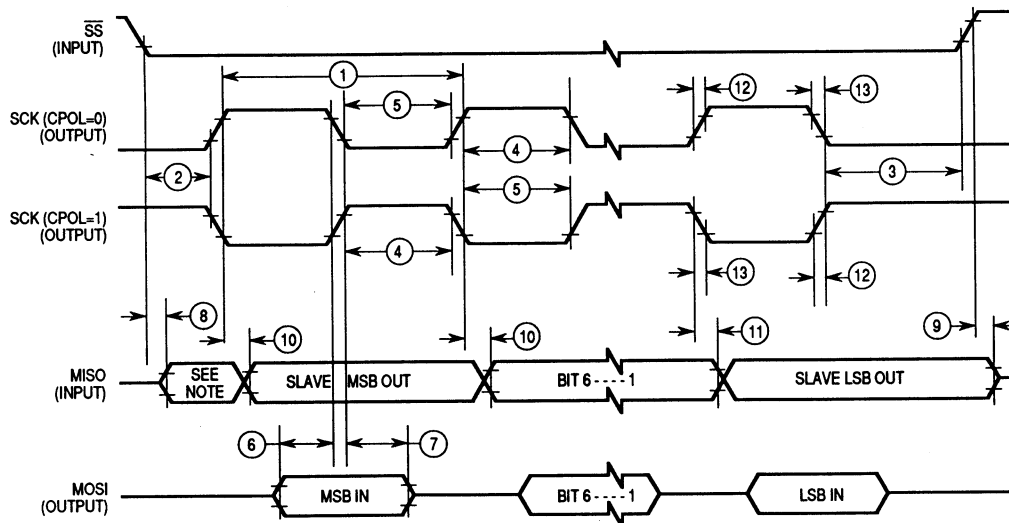
(b) SPI Master Timing (CPHA = 1)

Figure 8-7. 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 8-7. SPI Timing Diagrams (Sheet 2 of 2)



SECTION 9

MECHANICAL DATA

This section provides ordering information, pin assignments, and package dimensions for the MC68HC705C8.

9.1 ORDERING INFORMATION

The following table provides ordering information pertaining to the package type, temperature, and order numbers for the MC68HC705C8 device.

OTPROM MCU Devices

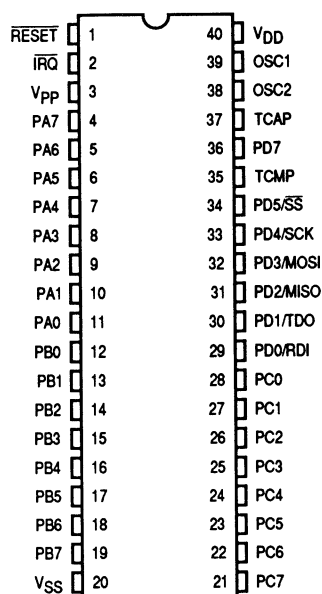
Package Type	Temperature	Order Number
Plastic (P Suffix)	0°C to +70°C	MC68HC705C8P
	–40°C to +85°C	MC68HC705C8CP
PLCC (FN Suffix)	0°C to +70°C	MC68HC705C8FN
	–40°C to +85°C	MC68HC705C8CFN

EPROM MCU Device

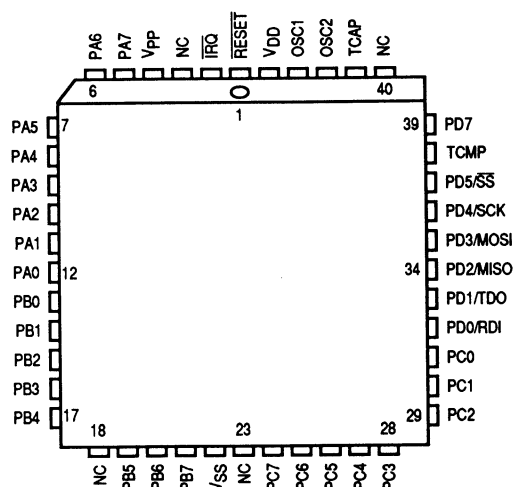
Package Type	Temperature	Part Number
Cerdip (S Suffix)	0°C to +70°C	MC68HC705C8S
	–40°C to +85°C	MC68HC705C8CS

9.2 PIN ASSIGNMENTS

9.2.1 40-Pin Dual-in-Line Package



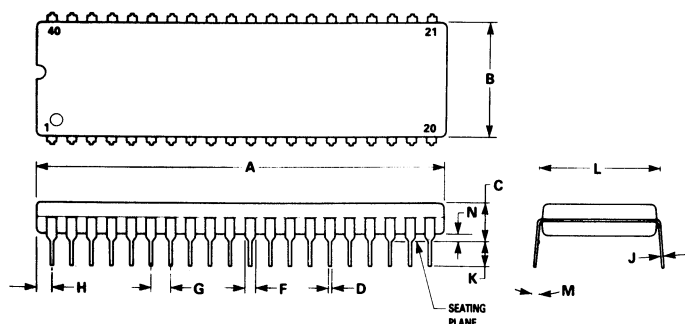
9.2.2 44-Lead PLCC Package



NOTE: Bulk substrate tied to VSS.

9.3 PACKAGE DIMENSIONS

P SUFFIX
PLASTIC PACKAGE
CASE 711-03

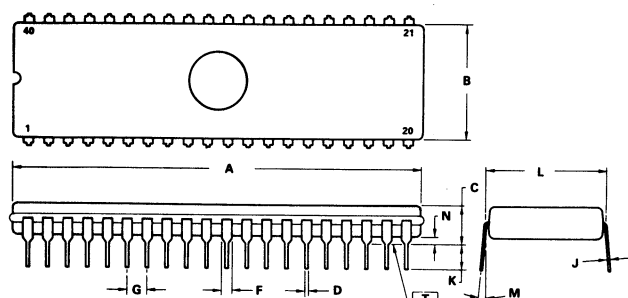


NOTES:

1. POSITIONAL TOLERANCE OF LEADS (D), SHALL BE WITHIN 0.25 mm (0.010) AT MAXIMUM MATERIAL CONDITION, IN RELATION TO SEATING PLANE AND EACH OTHER.
2. DIMENSION L TO CENTER OF LEADS WHEN FORMED PARALLEL.
3. DIMENSION B DOES NOT INCLUDE MOLD FLASH.

DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	51.69	52.45	2.035	2.065
B	13.72	14.22	0.540	0.560
C	3.94	5.08	0.155	0.200
D	0.36	0.56	0.014	0.022
F	1.02	1.52	0.040	0.060
G	2.54 BSC		0.100 BSC	
H	1.65	2.16	0.065	0.085
J	0.20	0.38	0.008	0.015
K	2.92	3.43	0.115	0.135
L	15.24 BSC		0.600 BSC	
M	0°	15°	0°	15°
N	0.51	1.02	0.020	0.040

S SUFFIX
CERDIP PACKAGE
CASE 734A-01



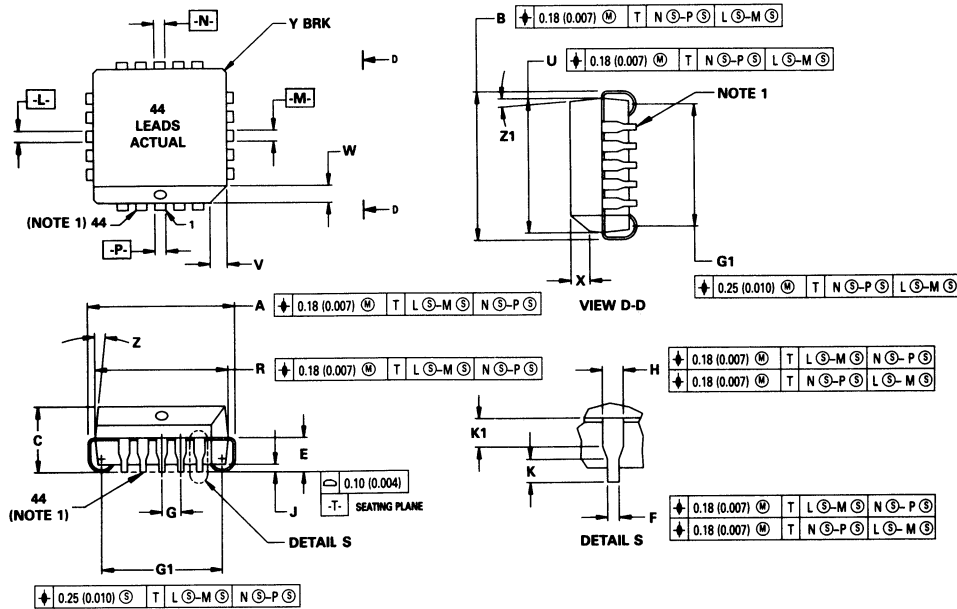
NOTES:

1. DIMENSION "A" IS A DATUM. T IS BOTH A DATUM AND A SEATING PLANE.
2. POSITIONAL TOLERANCE FOR LEADS: (40 PLACES)
 $\phi 0.25 (0.010) \text{ (M)} \text{ T } A \text{ (M)}$
3. DIMENSIONS A & B INCLUDE MENSICUS.
4. DIMENSION L TO CENTER OF LEADS WHEN FORMED PARALLEL.
5. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 982.
6. CONTROLLING DIMENSION: INCH.

DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	51.31	53.23	2.020	2.096
B	12.70	15.94	0.500	0.610
C	4.06	6.09	0.160	0.240
D	0.38	0.55	0.015	0.022
F	1.27	1.65	0.050	0.065
G	2.54 BSC		0.100 BSC	
J	0.20	0.30	0.008	0.012
K	3.17	4.06	0.125	0.160
L	15.24 BSC		0.600 BSC	
M	0°	15°	0°	15°
N	0.51	1.27	0.020	0.050

Freescale Semiconductor, Inc.

FN SUFFIX
PLASTIC-LEADED CHIP CARRIER
CASE 777-02



DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	17.40	17.65	0.685	0.695
B	17.40	17.65	0.685	0.695
C	4.20	4.57	0.165	0.180
E	2.29	2.79	0.090	0.110
F	0.33	0.48	0.013	0.019
G	1.27 BSC		0.050 BSC	
H	0.66	0.81	0.026	0.032
J	0.51	—	0.020	—
K	0.64	—	0.025	—
R	16.51	16.66	0.650	0.656
U	16.51	16.66	0.650	0.656
V	1.07	1.21	0.042	0.048
W	1.07	1.21	0.042	0.048
X	1.07	1.42	0.042	0.056
Y	—	0.50	—	0.020
Z	2°	10°	2°	10°
G1	15.50	16.00	0.610	0.630
K1	1.02	—	0.040	—
Z1	2°	10°	2°	10°

- NOTES:
1. DUE TO SPACE LIMITATION, CASE 777-02 SHALL BE REPRESENTED BY A GENERAL (SMALLER) CASE OUTLINE DRAWING RATHER THAN SHOWING ALL 44 LEADS.
 2. DATUMS -L-, -M-, -N-, AND -P- DETERMINED WHERE TOP OF LEAD SHOULDER EXIT PLASTIC BODY AT MOLD PARTING LINE.
 3. DIM G1, TRUE POSITION TO BE MEASURED AT DATUM -T-, SEATING PLANE.
 4. DIM R AND U DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE MOLD PROTRUSION IS 0.25 (0.010) PER SIDE.
 5. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
 6. CONTROLLING DIMENSION: INCH.

Freescale Semiconductor, Inc.

Freescale Semiconductor, Inc.

**For More Information On This Product,
Go to: www.freescale.com**

Freescale Semiconductor, Inc.

Home Page:

www.freescale.com

email:

support@freescale.com

USA/Europe or Locations Not Listed:

Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
(800) 521-6274
480-768-2130

support@freescale.com

Europe, Middle East, and Africa:

Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:

Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku
Tokyo 153-0064, Japan
0120 191014
+81 2666 8080
support.japan@freescale.com

Asia/Pacific:

Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate,
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:

Freescale Semiconductor
Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
(800) 441-2447
303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

RoHS-compliant and/or Pb- free versions of Freescale products have the functionality and electrical characteristics of their non-RoHS-compliant and/or non-Pb- free counterparts. For further information, see <http://www.freescale.com> or contact your Freescale sales representative.

For information on Freescale's Environmental Products program, go to <http://www.freescale.com/epp>.

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document. Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.



**For More Information On This Product,
Go to: www.freescale.com**