

# HC05

**MC68HC705K1**

TECHNICAL  
DATA




**MOTOROLA**



# MC68HC705K1

## 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 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 Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola 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 Motorola was negligent regarding the design or manufacture of the part. Motorola and  are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.



# TABLE OF CONTENTS

Paragraph	Title	Page
<b>SECTION 1 GENERAL DESCRIPTION</b>		
1.1	Features.....	1-1
1.2	Mask Options.....	1-2
1.3	MCU Structure.....	1-3
1.4	Pin Assignments .....	1-4
1.4.1	V <sub>DD</sub> and V <sub>SS</sub> .....	1-4
1.4.2	OSC1, OSC2, and PB1/OSC3.....	1-5
1.4.2.1	Crystal.....	1-6
1.4.2.2	Ceramic Resonator.....	1-7
1.4.2.3	Two-Pin RC Oscillator.....	1-8
1.4.2.4	Three-Pin RC Oscillator.....	1-8
1.4.2.5	External Clock.....	1-9
1.4.3	RESET .....	1-10
1.4.4	IRQ/V <sub>PP</sub> .....	1-10
1.4.5	PA7-PA0.....	1-10
1.4.6	PB1/OSC3 and PB0.....	1-10
<b>SECTION 2 MEMORY</b>		
2.1	Memory Map.....	2-1
2.2	Input/Output Section.....	2-1
2.3	RAM.....	2-1
2.4	EPROM/OTPROM .....	2-4
2.5	Personality EPROM/OTPROM.....	2-4
<b>SECTION 3 CENTRAL PROCESSOR UNIT</b>		
3.1	CPU Registers.....	3-1
3.1.1	Accumulator.....	3-2
3.1.2	Index Register.....	3-2
3.1.3	Stack Pointer.....	3-3
3.1.4	Program Counter.....	3-4

## TABLE OF CONTENTS (Continued)

Paragraph	Title	Page
3.1.5	Condition Code Register .....	3-4
3.1.5.1	Half-Carry Flag (H) .....	3-4
3.1.5.2	Interrupt Mask (I) .....	3-5
3.1.5.3	Negative Flag (N) .....	3-5
3.1.5.4	Zero Flag (Z) .....	3-5
3.1.5.5	Carry/Borrow Flag (C) .....	3-5
3.2	Arithmetic/Logic Unit (ALU) .....	3-6

### SECTION 4 INTERRUPTS

4.1	Interrupt Types .....	4-1
4.1.1	Software Interrupt .....	4-2
4.1.2	External Interrupts .....	4-2
4.1.2.1	$\overline{\text{IRQ}}$ Pin .....	4-2
4.1.2.2	PA3–PA0 Pins .....	4-3
4.1.2.3	IRQ Status and Control Register (ISCR) .....	4-4
4.1.3	Timer Interrupts .....	4-5
4.1.3.1	Timer Overflow Interrupt .....	4-5
4.1.3.2	Real-Time Interrupt .....	4-5
4.2	Interrupt Processing .....	4-6

### SECTION 5 RESETS

5.1	Reset Types .....	5-1
5.1.1	Power-On Reset .....	5-1
5.1.2	External Reset .....	5-1
5.1.3	Computer Operating Properly (COP) Reset .....	5-2
5.1.4	Illegal Address Reset .....	5-3
5.1.5	Low Voltage Reset .....	5-3
5.2	Reset States .....	5-4
5.2.1	CPU .....	5-4
5.2.2	I/O Port Registers .....	5-4
5.2.3	Multifunction Timer .....	5-4
5.2.4	COP Timer .....	5-4

### SECTION 6 LOW POWER MODES

6.1	Stop Mode .....	6-1
6.2	Wait Mode .....	6-2

## TABLE OF CONTENTS (Continued)

Paragraph	Title	Page
6.3	Halt Mode.....	6-3
6.4	Data-Retention Mode.....	6-3
<b>SECTION 7 PARALLEL I/O</b>		
7.1	I/O Port Function.....	7-1
7.2	Port A.....	7-1
7.2.1	Port A Data Register (PORTA).....	7-1
7.2.2	Data Direction Register A (DDRA).....	7-2
7.2.3	Pulldown Register A (PDRA).....	7-3
7.2.4	Port A External Interrupts.....	7-4
7.3	Port B.....	7-6
7.3.1	Port B Data Register (PORTB).....	7-6
7.3.2	Data Direction Register B (DDRB).....	7-7
7.3.3	Pulldown Register B (PDRB).....	7-8
<b>SECTION 8 MULTIFUNCTION TIMER</b>		
8.1	Timer Status and Control Register (TSCR).....	8-2
8.2	Timer Counter Register (TCNTR).....	8-4
8.3	COP Watchdog.....	8-5
<b>SECTION 9 EPROM/OTPROM</b>		
9.1	EPROM Programming Register (EPROG).....	9-1
9.2	EPROM/OTPROM Programming.....	9-2
9.3	EPROM Erasing.....	9-4
9.4	Mask Option Register (MOR).....	9-5
<b>SECTION 10 PERSONALITY EPROM</b>		
10.1	PEPROM Registers.....	10-2
10.1.1	PEPROM Bit Select Register (PEBSR).....	10-2
10.1.2	PEPROM Status and Control Register (PESCR).....	10-4
10.2	PEPROM Programming.....	10-5
10.3	PEPROM Reading.....	10-5
10.4	PEPROM Erasing.....	10-6

## TABLE OF CONTENTS (Concluded)

Paragraph	Title	Page
<b>SECTION 11</b>		
<b>INSTRUCTION SET</b>		
11.1	Addressing Modes.....	11-1
11.1.1	Inherent.....	11-1
11.1.2	Immediate.....	11-2
11.1.3	Direct.....	11-3
11.1.4	Extended.....	11-5
11.1.5	Indexed, No Offset.....	11-6
11.1.6	Indexed, 8-Bit Offset.....	11-6
11.1.7	Indexed, 16-Bit Offset.....	11-6
11.1.8	Relative.....	11-8
11.2	Instruction Types.....	11-9
11.2.1	Register/Memory Instructions.....	11-9
11.2.2	Read-Modify-Write Instructions.....	11-10
11.2.3	Jump/Branch Instructions.....	11-10
11.2.4	Bit Manipulation Instructions.....	11-12
11.2.5	Control Instructions.....	11-12
11.3	Instruction Set Summary.....	11-13
11.4	Opcode Map.....	11-18
<b>SECTION 12</b>		
<b>ELECTRICAL SPECIFICATIONS</b>		
12.1	Maximum Ratings.....	12-1
12.2	Thermal Characteristics.....	12-1
12.3	Power Considerations.....	12-2
12.4	DC Electrical Characteristics ( $V_{DD} = 5.0$ Vdc).....	12-3
12.5	DC Electrical Characteristics ( $V_{DD} = 3.3$ Vdc).....	12-4
12.6	Control Timing ( $V_{DD} = 5.0$ Vdc).....	12-7
12.7	Control Timing ( $V_{DD} = 3.3$ Vdc).....	12-8
<b>SECTION 13</b>		
<b>MECHANICAL SPECIFICATIONS</b>		
13.1	Plastic Dual In-Line Package (PDIP).....	13-1
13.2	Small Outline Integrated Circuit (SOIC).....	13-2
13.3	Ceramic DIP (Cerdip).....	13-3



## LIST OF FIGURES

Figure	Title	Page
1-1	MC68HC705K1 Block Diagram.....	1-3
1-2	Pin Assignments.....	1-4
1-3	Bypassing Layout Recommendation .....	1-5
1-4	Crystal Connections.....	1-6
1-5	Ceramic Resonator Connections.....	1-7
1-6	Two-Pin RC Oscillator Connections.....	1-8
1-7	Three-Pin RC Oscillator Connections.....	1-9
1-8	External Clock Connections .....	1-9
2-1	Memory Map.....	2-2
2-2	I/O Registers.....	2-3
3-1	Programming Model.....	3-1
3-2	Accumulator.....	3-2
3-3	Index Register.....	3-2
3-4	Stack Pointer.....	3-3
3-5	Program Counter.....	3-4
3-6	Condition Code Register.....	3-4
4-1	External Interrupt Function.....	4-3
4-2	IRQ Status and Control Register (ISCR).....	4-4
4-3	Interrupt Stacking Order .....	4-6
4-4	Interrupt Flowchart.....	4-8
5-1	Reset Sources.....	5-2
5-2	COP Register (COPR).....	5-3
6-1	Stop/Wait/Halt Flowchart.....	6-4
7-1	Port A Data Register (PORTA).....	7-2
7-2	Data Direction Register A (DDRA).....	7-2
7-3	Pulldown Register A (PDRA) .....	7-3
7-4	Port A I/O Circuit.....	7-5
7-5	Port B Data Register (PORTB).....	7-6

## LIST OF FIGURES (Concluded)

Figure	Title	Page
7-6	Data Direction Register B (DDRB).....	7-7
7-7	Pulldown Register B (PDRB).....	7-8
7-8	Port B I/O Circuit.....	7-9
8-1	Timer Block Diagram.....	8-1
8-2	Timer Status and Control Register (TSCR).....	8-2
8-3	Timer Counter Register (TCNTR).....	8-4
8-4	COP Register (COPR).....	8-5
9-1	EPROM Programming Register (EPROG).....	9-1
9-2	Programming Circuit.....	9-3
9-3	Mask Option Register (MOR).....	9-5
10-1	Personality EPROM.....	10-1
10-2	PEPROM Bit Select Register (PEBSR).....	10-2
10-3	PEPROM Status and Control Register (PESCR).....	10-4
12-1	Equivalent Test Load.....	12-2
12-2	Typical High-Side Driver Characteristics.....	12-5
12-3	Typical Low-Side Driver Characteristics.....	12-5
12-4	Typical Supply Current vs Clock Frequency.....	12-6
12-5	Maximum Supply Current vs Clock Frequency.....	12-6
12-6	External Interrupt Timing.....	12-9
12-7	STOP Recovery Timing.....	12-9
12-8	Power-On Reset Timing.....	12-10
12-9	External Reset Timing.....	12-10
13-1	MC68HC705K1P (Case 648-08).....	13-1
13-2	MC68HC705K1DW (Case 751G-01).....	13-2
13-3	MC68HC705K1S (Case 620-09).....	13-2

## LIST OF TABLES

Table	Title	Page
4-1	Reset/Interrupt Vector Addresses .....	4-7
7-1	Port A Pin Functions.....	7-6
7-2	PB0 Pin Functions .....	7-10
7-3	PB1/OSC3 Pin Functions.....	7-10
8-1	Real-Time Interrupt Rate Selection .....	8-3
8-2	COP Watchdog Recommendations.....	8-6
10-1	PEPROM Bit Selection.....	10-3
11-1	Inherent Addressing Instructions .....	11-2
11-2	Immediate Addressing Instructions .....	11-3
11-3	Direct Addressing Instructions.....	11-4
11-4	Extended Addressing Instructions.....	11-5
11-5	Indexed Addressing Instructions.....	11-7
11-6	Relative Addressing Instructions.....	11-8
11-7	Register/Memory Instructions .....	11-9
11-8	Read-Modify-Write Instructions .....	11-10
11-9	Jump and Branch Instructions.....	11-11
11-10	Bit Manipulation Instructions.....	11-12
11-11	Control Instructions.....	11-12
11-12	Instruction Set.....	11-14
11-13	Opcode Map .....	11-18
12-1	Maximum Ratings .....	12-1
12-2	Thermal Resistance.....	12-1
12-3	DC Electrical Characteristics ( $V_{DD} = 5.0$ Vdc).....	12-3
12-4	DC Electrical Characteristics ( $V_{DD} = 3.3$ Vdc).....	12-4
12-5	Control Timing ( $V_{DD} = 5.0$ Vdc).....	12-7
12-6	Control Timing ( $V_{DD} = 3.3$ Vdc).....	12-8



## SECTION 1 GENERAL DESCRIPTION

The MC68HC705K1 is a member of the low-cost, high-performance M68HC05 Family of 8-bit microcontroller units (MCUs). The M68HC05 Family is based on the customer-specified integrated circuit (CSIC) design strategy. All MCUs in the family use the popular M68HC05 central processor unit (CPU) and are available with a variety of subsystems, memory sizes and types, and package types.

On-chip memory of the MC68HC705K1 includes 504 bytes of erasable, programmable ROM (EPROM). In packages without the transparent window for EPROM erasure, the 504 EPROM bytes serve as one-time programmable ROM (OTPROM).

### 1.1 Features

Features of the MCU include the following:

- Popular M68HC05 Central Processor Unit
- Memory-Mapped Input/Output (I/O) Registers
- 504 Bytes of Erasable Programmable ROM/One-Time Programmable ROM (EPROM/OTPROM) including 8 User Vector Locations
- 32 Bytes of User RAM
- 64-Bit Personality EPROM
- 10 Bidirectional I/O Pins with Software-Programmable Pulldown Devices
- 8 mA Sink Capability on 4 I/O Pins
- External Interrupt Capability on 4 I/O Pins
- Hardware Mask and Flag for External Interrupts
- Fully Static Operation with no Minimum Clock Speed
- On-Chip Oscillator with Crystal or Ceramic Resonator Connections or Mask-Optional Two-Pin/Three-Pin Resistor-Capacitor (RC) Connections
- Computer Operating Properly (COP) Watchdog
- 15-Bit Multifunction Timer with Real-Time Interrupt Circuit
- Power-Saving Stop, Wait, Halt, and Data-Retention Modes
- 8 × 8 Unsigned Multiply Instruction
- 16-Pin Plastic Dual In-Line Package (PDIP)
- 16-Pin Small Outline Integrated Circuit (SOIC)
- 16-Pin Ceramic DIP (Cerdip)

## 1.2 Mask Options

The following MC68HC705K1 options are programmable in the mask option register:

- Enabled or disabled COP watchdog
- Edge-triggered or edge- and level-triggered external interrupt pins
- Enabled or disabled port A external interrupt function
- Enabled or disabled low-voltage reset function
- Enabled or disabled STOP instruction
- Oscillator driven by crystal or ceramic resonator or oscillator driven by resistor-capacitor (RC) circuit
- Two-pin RC-driven oscillator or three-pin RC-driven oscillator
- Enabled or disabled port A and port B programmable pulldown transistors

The mask option register is an EPROM/OTPROM byte at location \$0017. **SECTION 9 EPROM/OTPROM** describes the mask option register and the EPROM/OTPROM programming procedure.

### 1.3 MCU Structure

Figure 1-1 shows the structure of the MC68HC705K1 MCU.

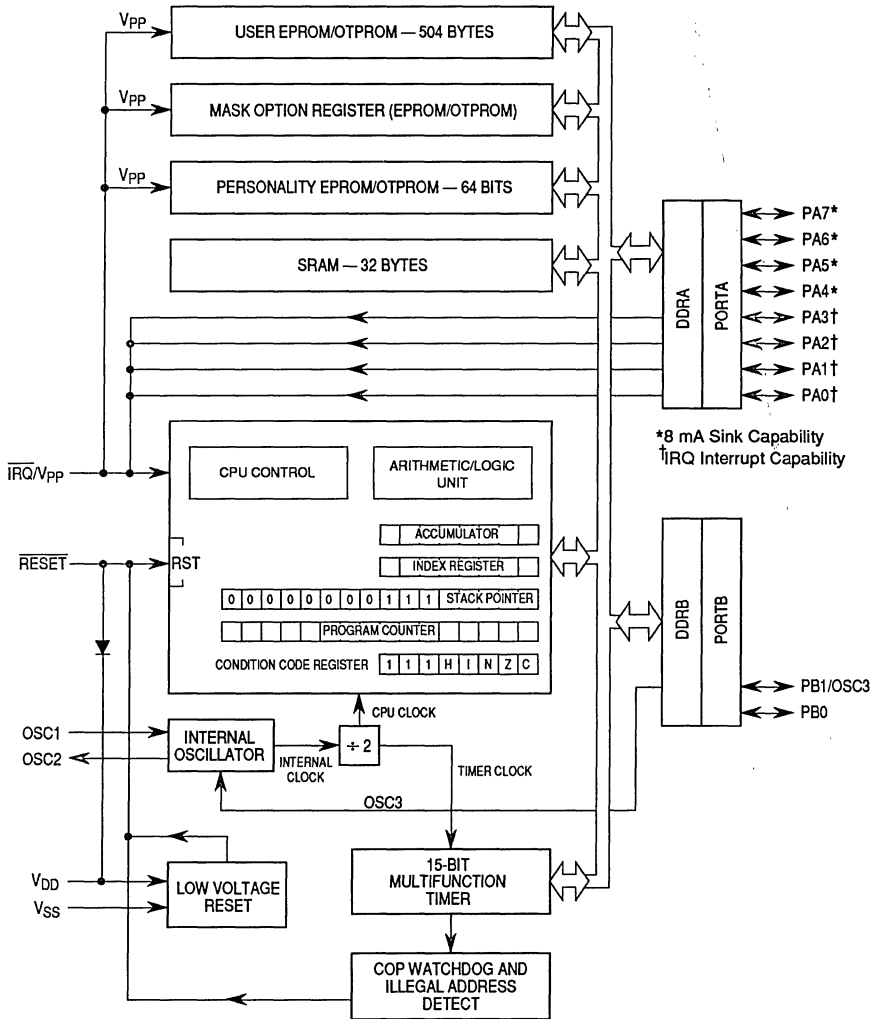


Figure 1-1. MC68HC705K1 Block Diagram

## 1.4 Pin Assignments

Figure 1-2 shows the pin assignments.

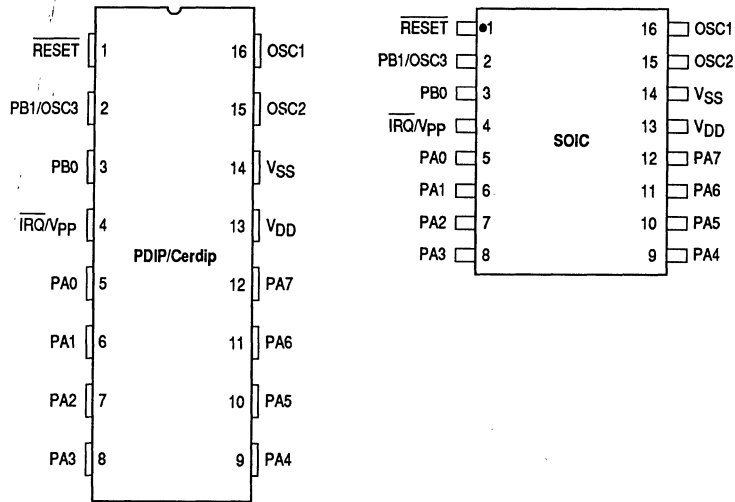


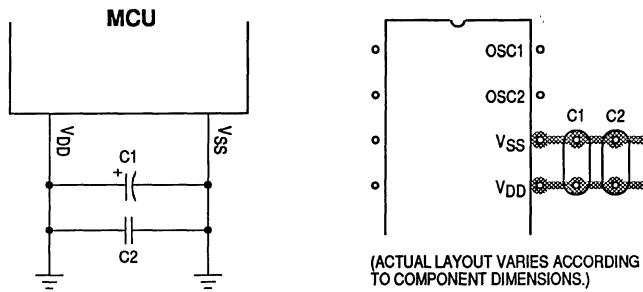
Figure 1-2. Pin Assignments

### 1.4.1 V<sub>DD</sub> and V<sub>SS</sub>

V<sub>DD</sub> and V<sub>SS</sub> are the power supply and ground pins. The MCU operates from a single 5-V power supply.

Very fast signal transitions occur on the MCU pins, placing very high short-duration current demands on the power supply. To prevent noise problems, take special care to provide good power supply bypassing at the MCU. Place bypass capacitors as close to the MCU as possible, as Figure 1-3 shows.





**Figure 1-3. Bypassing Layout Recommendation**

### 1.4.2 OSC1, OSC2, and PB1/OSC3

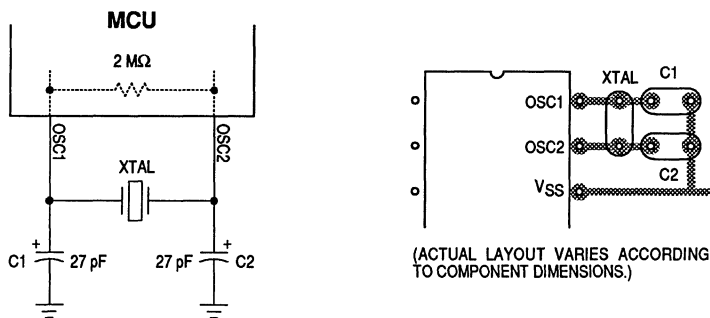
The OSC1, OSC2, and PB1/OSC3 pins are the control connections for the two-pin or three-pin on-chip oscillator. The oscillator can be driven by any of the following:

- Crystal (See Figure 1-4.)
- Ceramic resonator (See Figure 1-5.)
- Resistor-capacitor network (See Figure 1-6.)
- Resistor-resistor-capacitor network (See Figure 1-7.)
- External clock signal (See Figure 1-8.)

The MCU divides the frequency,  $f_{OSC}$ , of the oscillator or external clock source by two to produce the internal operating frequency,  $f_{OP}$ .

### 1.4.2.1 Crystal

The circuit in Figure 1-4 shows a typical crystal oscillator circuit for an AT-cut parallel resonant crystal. Follow the crystal supplier's recommendations, as the crystal parameters determine the external component values required to provide maximum stability and reliable start-up. The load capacitance values used in the oscillator circuit design should include all stray layout capacitances. Mount the crystal and components as close as possible to the pins for start-up stabilization and to minimize output distortion.



**Figure 1-4. Crystal Connections**

To use the crystal-driven oscillator, the RC bit in the mask option register must be clear. See **9.4 Mask Option Register (MOR)**. Clearing the RC bit connects an internal 2-MΩ start-up resistor between OSC1 and OSC2.

### 1.4.2.2 Ceramic Resonator

To reduce cost, use a ceramic resonator in place of the crystal. Use the circuit in Figure 1-5 for a ceramic resonator, and follow the resonator manufacturer's recommendations. The resonator parameters determine the external component values required for maximum stability and reliable starting. The load capacitance values used in the oscillator circuit design should include all stray layout capacitances.

To use the ceramic resonator-driven oscillator, the RC bit in the mask option register must be clear. See **9.4 Mask Option Register (MOR)**. Clearing the RC bit connects an internal 2-M $\Omega$  start-up resistor between OSC1 and OSC2.

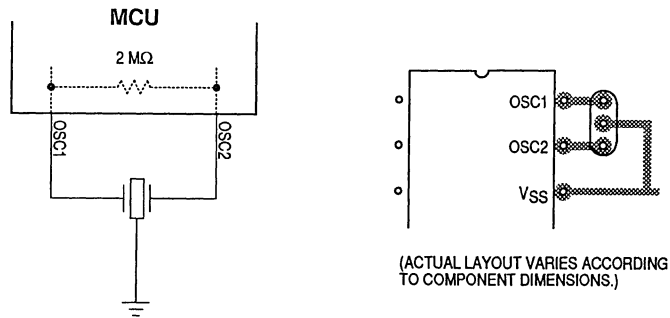


Figure 1-5. Ceramic Resonator Connections

### 1.4.2.3 Two-Pin RC Oscillator

For maximum cost reduction, use the two-pin RC oscillator configuration shown in Figure 1-6. The relationship between  $f_{OSC}$  and the external components is  $f_{OSC} \approx 1 \div 2.28RC$ . The OSC2 signal is a square wave, and the signal on OSC1 is a triangular wave. In the mask option register, the RC bit must be programmed to logic one. Setting the RC bit disconnects the internal start-up resistor. The PIN3 bit in the mask option register must remain erased (logic zero). The PIN3 bit selects the three-pin RC oscillator configuration. The optimum frequency for the two-pin oscillator configuration is 2 MHz.

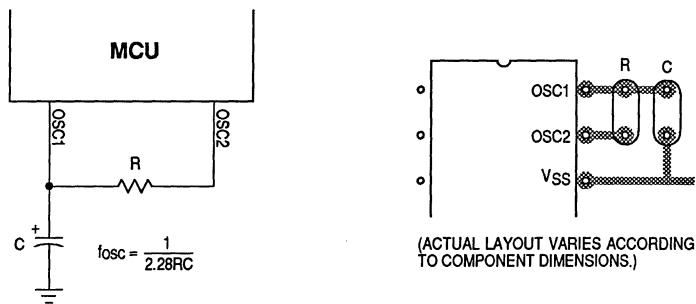
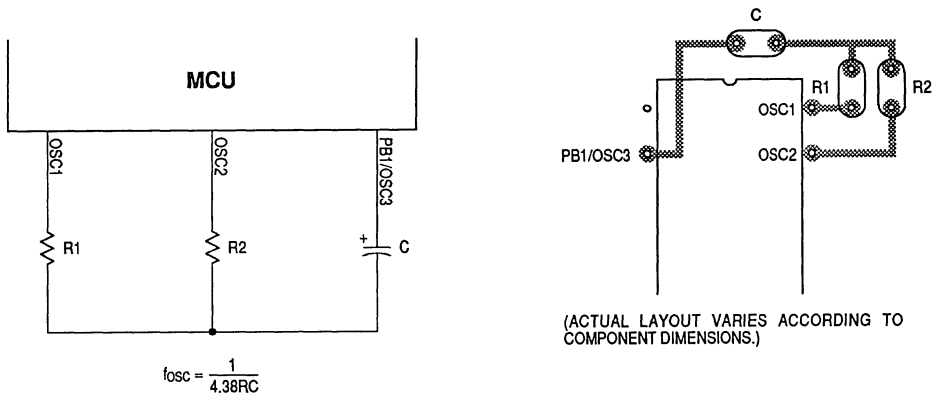


Figure 1-6. Two-Pin RC Oscillator Connections

### 1.4.2.4 Three-Pin RC Oscillator

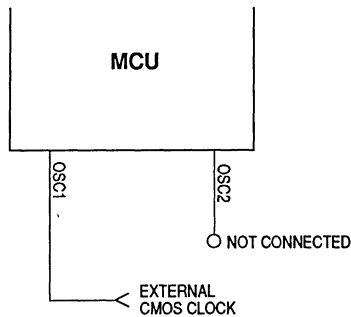
Another low-cost option is the three-pin RC oscillator configuration shown in Figure 1-7. The three-pin oscillator is more stable through temperature and supply voltage variations than the two-pin oscillator. The relationship between  $f_{OSC}$  and the external components is  $f_{OSC} \approx 1 \div 4.38RC$ . The OSC2 and PB1/OSC3 signals are square waves, and the signal on OSC1 is a triangular wave. In the mask option register, both the RC and PIN3 bits must be programmed to logic one to use this configuration. The resistor (0–10 M $\Omega$ ) connected to the OSC1 pin improves accuracy by reducing the effects of clipping by the OSC1 input protection circuit during the RC charge/discharge cycle. The three-pin RC oscillator configuration works best for frequencies under 1 MHz.



**Figure 1-7. Three-Pin RC Oscillator Connections**

#### 1.4.2.5 External Clock

An external clock from another CMOS-compatible device can drive the OSC1 input, with the OSC2 pin not connected, as Figure 1-8 shows. To reduce emission of radio frequency interference from the OSC2 pin, connect OSC2 to ground through a 10-Ω–100-Ω resistance.



**Figure 1-8. External Clock Connections**

### 1.4.3 $\overline{\text{RESET}}$

A logic zero on the RESET pin forces the MCU to a known start-up state. (See 5.1.2 External Reset for more information.)

### 1.4.4 $\overline{\text{IRQ/VPP}}$

The  $\overline{\text{IRQ/VPP}}$  pin has the following functions:

- Applying asynchronous external interrupt signals (See 4.1.2 External Interrupts.)
- Applying the EPROM/OTPROM programming voltage (See 9.2 EPROM/OTPROM Programming.)

### 1.4.5 PA7–PA0

PA7–PA0 are the pins of port A, a general-purpose bidirectional I/O port. (See 7.2 Port A.)

### 1.4.6 PB1/OSC3 and PB0

PB1/OSC3 and PB0 are the pins of port B, a general-purpose bidirectional I/O port. (See 7.3 Port B.)

## SECTION 2 MEMORY

This section describes the organization of the on-chip memory.

### 2.1 Memory Map

The CPU can address 1 Kbyte of memory space. The program counter typically advances one address at a time through the memory, reading the program instructions and data. The EPROM portion of memory holds the program instructions, fixed data, user-defined vectors, and interrupt service routines. The RAM portion of memory holds variable data. I/O registers are memory-mapped so that the CPU can access their locations in the same way that it accesses all other memory locations.

Figure 2-1 is a memory map of the MCU. Refer to Figure 2-2 for a more detailed memory map of the 32-byte I/O register section.

### 2.2 Input/Output Section

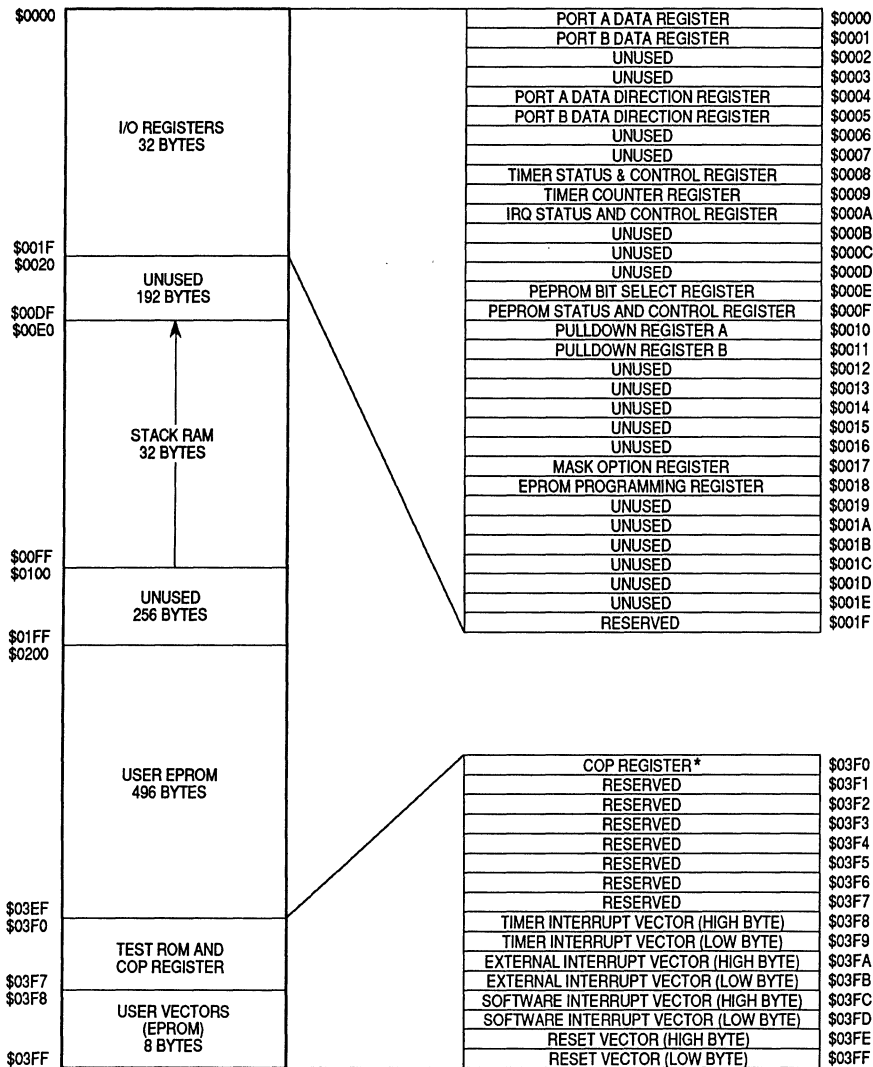
The first 32 addresses of the memory space, \$0000–\$001F, are the I/O section. These are the addresses of the I/O control registers, status registers, and data registers.

### 2.3 RAM

Addresses \$00E0–\$00FF serve as both the stack RAM and the user RAM. The CPU uses five RAM bytes to save all CPU register contents before processing an interrupt. During a subroutine call, the CPU uses two bytes to store the return address. The stack pointer decrements during pushes and increments during pulls.

### NOTE

Be careful when using nested subroutines or multiple interrupt levels. The CPU may overwrite data in the RAM during a subroutine or during the interrupt stacking operation.



\*WRITING ZERO TO BIT 0 OF \$03F0 CLEARS COP TIMER. READING \$03F0 RETURNS TEST ROM DATA.

Figure 2-1. Memory Map



	Bit 7	6	5	4	3	2	1	Bit 0	
\$0000	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0	PORTA
\$0001	0	0	0	0	0	0	PB1	PB0	PORTB
\$0002	—	—	—	—	—	—	—	—	UNUSED
\$0003	—	—	—	—	—	—	—	—	UNUSED
\$0004	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0	DDRA
\$0005	0	0	0	0	0	0	DDRB1	DDRB0	DDRB
\$0006	—	—	—	—	—	—	—	—	UNUSED
\$0007	—	—	—	—	—	—	—	—	UNUSED
\$0008	TOF	RTIF	TOIE	RTIE	TOFR	RTIFR	RT1	RT0	TSCR
\$0009	Bit 7	6	5	4	3	2	1	Bit 0	TCNTR
\$000A	IRQE	0	0	0	IRQF	0	IRQR	0	ISCR
\$000B	—	—	—	—	—	—	—	—	UNUSED
\$000C	—	—	—	—	—	—	—	—	UNUSED
\$000D	—	—	—	—	—	—	—	—	UNUSED
\$000E	PEB7	PEB6	PEB5	PEB4	PEB3	PEB2	PEB1	PEB0	PEBSR
\$000F	PEDATA	0	PEPGM	0	0	0	0	PEPRZF	PESCR
\$0010	PDIA7	PDIA6	PDIA5	PDIA4	PDIA3	PDIA2	PDIA1	PDIA0	PDRA
\$0011	—	—	—	—	—	—	PDIB1	PDIB0	PDRB
\$0012	—	—	—	—	—	—	—	—	UNUSED
.									.
.									.
.									.
\$0016	—	—	—	—	—	—	—	—	UNUSED
\$0017	SWPDI	PIN3	RC	SWAIT	LVIE	PIRQ	LEVEL	COPEN	MOR
\$0018	0	0	0	0	0	ELAT	MGPM	EPGM	EPROG
\$0019	—	—	—	—	—	—	—	—	UNUSED
.									.
.									.
.									.
\$001E	—	—	—	—	—	—	—	—	UNUSED
\$001F	—	—	—	—	—	—	—	—	RESERVED
\$03F0	—	—	—	—	—	—	—	COPC	COPR

Figure 2-2. I/O Registers

## 2.4 EPROM/OTPROM

An MCU with a quartz window has 504 bytes of erasable, programmable ROM (EPROM). The quartz window allows EPROM erasure with ultraviolet light. In an MCU without the quartz window, the EPROM cannot be erased and serves as 504 bytes of one-time programmable ROM (OTPROM). Addresses \$0200–\$03EF contain 496 bytes of user EPROM/OTPROM. The eight addresses from \$03F8–\$03FF are EPROM/OTPROM locations reserved for interrupt vectors and reset vectors.

## 2.5 Personality EPROM/OTPROM

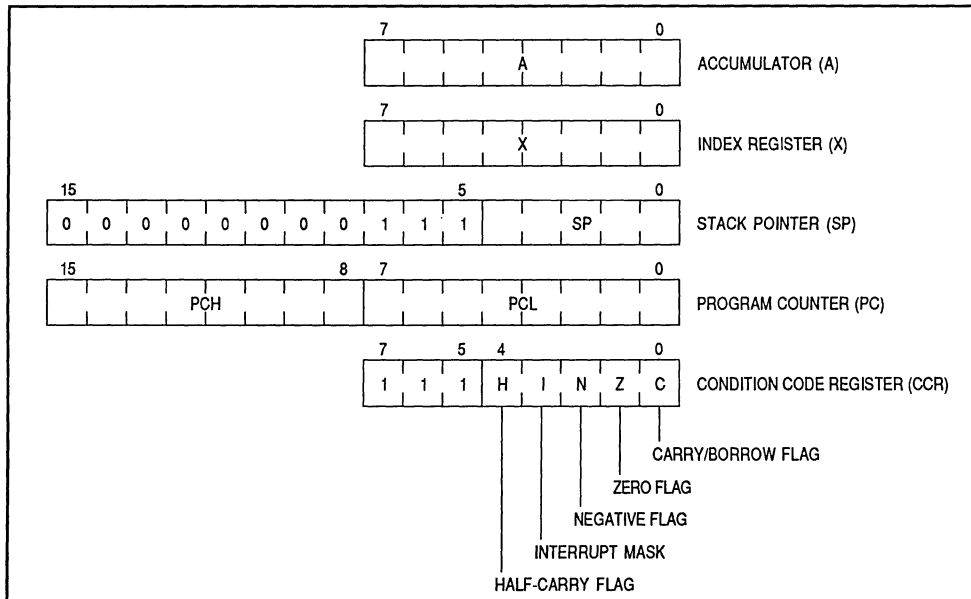
An MCU with a quartz window has a 64-bit array of erasable, programmable ROM (EPROM) to serve as a personality EPROM. The quartz window allows EPROM erasure with ultraviolet light. In an MCU without the quartz window, the personality EPROM cannot be erased and serves as a 64-bit array of one-time programmable ROM (OTPROM).

## SECTION 3 CENTRAL PROCESSOR UNIT

This section describes the CPU registers.

### 3.1 CPU Registers

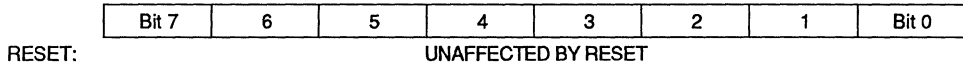
Figure 3-1 shows the five CPU registers. These are hard-wired registers within the CPU and are not part of the memory map.



**Figure 3-1. Programming Model**

### 3.1.1 Accumulator

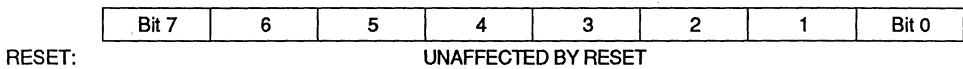
The accumulator is a general-purpose 8-bit register. The CPU uses the accumulator to hold operands and results of arithmetic and nonarithmetic operations.



**Figure 3-2. Accumulator**

### 3.1.2 Index Register

In the indexed addressing modes, the CPU uses the byte in the index register to determine the conditional address of the operand. See **11.1.5 Indexed, No Offset**, **11.1.6 Indexed, 8-Bit Offset**, and **11.1.7 Indexed, 16-Bit Offset**.

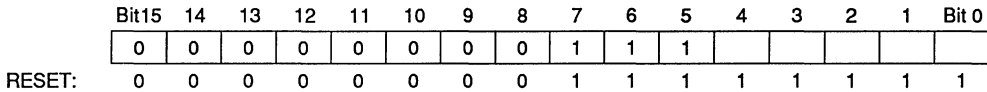


**Figure 3-3. Index Register**

The 8-bit index register can also serve as a temporary data storage location.

### 3.1.3 Stack Pointer

The stack pointer is a 16-bit register that contains the address of the next location on the stack. During a reset or after the reset stack pointer (RSP) instruction, the stack pointer is set to \$00FF. The address in the stack pointer decrements as data is pushed onto the stack and increments as data is pulled from the stack.



**Figure 3-4. Stack Pointer**

The eleven most significant bits of the stack pointer are permanently fixed at 00000000111, so the stack pointer produces addresses from \$00E0 to \$00FF. If subroutines and interrupts use more than 32 stack locations, the stack pointer wraps around to address \$00FF and begins writing over the previously stored data. A subroutine uses two stack locations; an interrupt uses five locations.

### 3.1.4 Program Counter

The program counter is a 16-bit register that contains the address of the next instruction or operand to be fetched. The six most significant bits of the program counter are ignored internally and appear as 000000.

Normally, the address in the program counter automatically increments to the next sequential memory location every time an instruction or operand is fetched. Jump, branch, and interrupt operations load the program counter with an address other than that of the next sequential location.

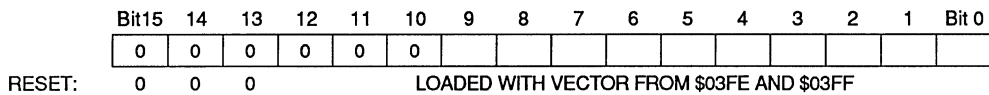


Figure 3-5. Program Counter

### 3.1.5 Condition Code Register

The condition code register is an 8-bit register whose three most significant bits are permanently fixed at 111. The condition code register contains the interrupt mask and four flags that indicate the results of the instruction just executed. The following paragraphs describe the functions of the condition code register.

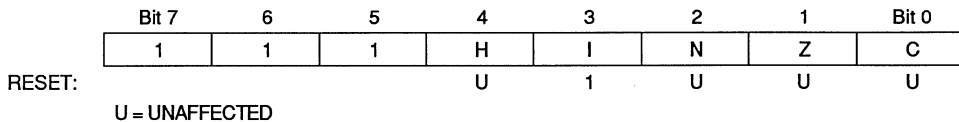


Figure 3-6. Condition Code Register

#### 3.1.5.1 Half-Carry Flag (H)

The CPU sets the half-carry flag when a carry occurs between bits 3 and 4 of the accumulator during an ADD or ADC operation. The half-carry flag is required for binary-coded decimal (BCD) arithmetic operations.

### **3.1.5.2 Interrupt Mask (I)**

Setting the interrupt mask disables interrupts. If an interrupt request occurs while the interrupt mask is logic zero, the CPU saves the CPU registers on the stack, sets the interrupt mask, and then fetches the interrupt vector. If an interrupt request occurs while the interrupt mask is set, the interrupt request is latched. Normally, the CPU processes the latched interrupt as soon as the interrupt mask is cleared again.

A return from interrupt (RTI) instruction pulls the CPU registers from the stack, restoring the interrupt mask to its cleared state. After any reset, the interrupt mask is set and can be cleared only by a CLI instruction.

### **3.1.5.3 Negative Flag (N)**

The CPU sets the negative flag when an arithmetic operation, logical operation, or data manipulation produces a negative result. Bit 7 of the negative result is automatically set, so the negative flag can be used to check an often-tested bit by assigning the tested bit to bit 7 of a register or memory location. Loading the accumulator with the contents of that register or location then sets or clears the negative flag according to the state of the tested bit.

### **3.1.5.4 Zero Flag (Z)**

The CPU sets the zero flag when an arithmetic operation, logical operation, or data manipulation produces a \$00 result.

### **3.1.5.5 Carry/Borrow Flag (C)**

The CPU sets the carry/borrow flag when an addition operation produces a carry out of bit 7 of the accumulator or when a subtraction operation requires a borrow. Some logical operations and data manipulation instructions also clear or set the carry/borrow flag.

### **3.2 Arithmetic/Logic Unit (ALU)**

The ALU performs the arithmetic and logical operations defined by the instruction set.

The decoded instructions set up the ALU for the selected operation. Most binary arithmetic is based on the addition algorithm, carrying out subtraction as negative addition. Multiplication is not performed as a discrete operation but as a chain of addition and shift operations within the ALU. The multiply instruction (MUL) requires 11 internal processor cycles to complete this chain of operations.



## SECTION 4 INTERRUPTS

This section describes how interrupts temporarily change the normal processing sequence.

### 4.1 Interrupt Types

An interrupt temporarily stops normal program execution to process a particular event. Unlike a reset, an interrupt does not stop the operation of the instruction being executed. An interrupt takes effect when the current instruction completes its execution. An interrupt saves the CPU registers on the stack and loads the program counter with a user-defined interrupt vector address. The following conditions produce an interrupt:

- SWI instruction (software interrupt)
- A logic zero applied to the  $\overline{IRQ}/V_{PP}$  pin (external interrupt)
- A logic one applied to one of the PA3–PA0 pins while the PIRQ bit is set (external interrupt)
- A timer overflow or real-time interrupt request (timer interrupt)

### 4.1.1 Software Interrupt

The software interrupt (SWI) instruction causes a nonmaskable interrupt.

### 4.1.2 External Interrupts

The following sources can generate external interrupts:

- $\overline{\text{IRQ}}/\text{V}_{\text{PP}}$  pin
- PA3–PA0 pins

Setting the I bit in the condition code register or clearing the IRQE bit in the interrupt status and control register disables external interrupts.

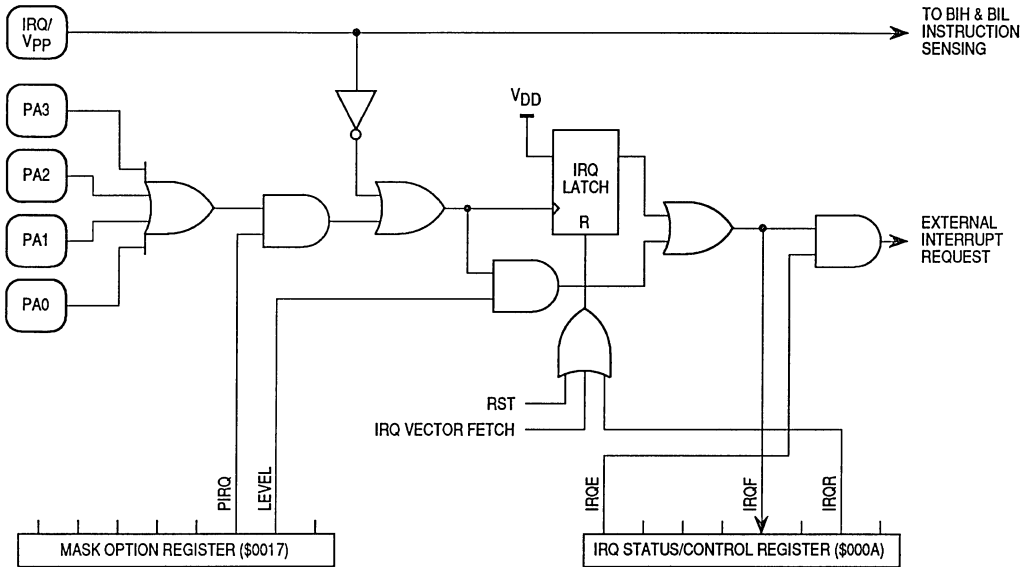
#### 4.1.2.1 $\overline{\text{IRQ}}/\text{V}_{\text{PP}}$ Pin

An interrupt signal on the  $\overline{\text{IRQ}}/\text{V}_{\text{PP}}$  pin latches an external interrupt request. When the CPU completes its current instruction, it tests the IRQ latch. If the IRQ latch is set, the CPU then tests the I bit in the condition code register and the IRQE bit in the interrupt status and control register. If the I bit is clear and the IRQE bit is set, the CPU then begins the interrupt sequence. The CPU clears the IRQ latch while it fetches the interrupt vector, so another external interrupt request can be latched during the interrupt service routine. As soon as the I bit is cleared during the return from interrupt, the CPU can recognize the new interrupt request.

The  $\overline{\text{IRQ}}/\text{V}_{\text{PP}}$  pin can be negative edge-triggered only or negative edge- and low-level-triggered. External interrupt sensitivity is programmable in the mask option register. See **9.4 Mask Option Register (MOR)**.

With the edge- and level-sensitive trigger option, a falling edge or a low level on the  $\overline{\text{IRQ}}/\text{V}_{\text{PP}}$  pin latches an external interrupt request. The edge- and level-sensitive trigger option allows multiple external interrupt sources to be wire-ORed to the  $\overline{\text{IRQ}}/\text{V}_{\text{PP}}$  pin. As long as any source is holding the  $\overline{\text{IRQ}}/\text{V}_{\text{PP}}$  pin low, an external interrupt request is latched, and the CPU continues to execute the interrupt service routine.

With the edge-sensitive-only trigger option, a falling edge on the  $\overline{\text{IRQ}}/\text{V}_{\text{PP}}$  pin latches an external interrupt request. A subsequent interrupt request can be latched only after the voltage level on the  $\overline{\text{IRQ}}/\text{V}_{\text{PP}}$  pin returns to logic one and then falls again to logic zero. Figure 4-1 shows the external interrupt logic.



**Figure 4-1. External Interrupt Function**

#### 4.1.2.2 PA3–PA0 Pins

Programming the PIRQ bit in the mask option register to a one enables pins PA3–PA0 to serve as additional external interrupt sources. A rising edge on a PA3–PA0 pin latches an external interrupt request. After completing the current instruction, the CPU tests the IRQ latch. If the IRQ latch is set, the CPU then tests the I bit in the condition control register and the IRQE bit in the interrupt status and control register. If the I bit is clear and the IRQE bit is set, the CPU then begins the interrupt sequence. The CPU clears the IRQ latch while it fetches the interrupt vector, so another external interrupt request can be latched during the interrupt service routine. As soon as the I bit is cleared during the return from interrupt, the CPU can recognize the new interrupt request.

The PA3–PA0 pins can be edge-triggered or edge- and level-triggered. External interrupt triggering sensitivity is programmable in the mask option register. See 9.4 Mask Option Register (MOR).

### 4.1.2.3 IRQ Status and Control Register (ISCR)

The IRQ status and control register, shown in Figure 4-2, contains an external interrupt mask and an external interrupt flag and flag reset bit.

**ISCR** — IRQ Status and Control Register

**\$000A**

	Bit 7	6	5	4	3	2	1	Bit 0
	IRQE	0	0	0	IRQF	0	IRQR	0
RESET:	1	0	0	0	0	0	0	0

**Figure 4-2. IRQ Status and Control Register (ISCR)**

**IRQE** — External Interrupt Request Enable

This read/write bit enables external interrupts.

1 = External interrupt processing enabled

0 = External interrupt processing disabled

**IRQF** — External Interrupt Request Flag (IRQ flag)

The IRQ flag is a clearable, read-only bit that is set when an external interrupt request is pending.

1 = Interrupt request pending

0 = Interrupt request not pending

The following conditions set the IRQ flag:

- An external interrupt signal on the  $\overline{\text{IRQ}}/\text{V}_{\text{PP}}$  pin
- An external interrupt signal on pin PA3, PA2, PA1, or PA0 when PA3–PA0 are enabled to serve as external interrupt sources

The CPU clears the IRQ flag when fetching the interrupt vector. Writing logic zero to the IRQ flag has no effect. Clear the IRQ flag by writing a logic one to the IRQR bit.

**IRQR** — Interrupt Request Reset

This write-only bit clears the IRQ flag.

1 = IRQ bit cleared

0 = No effect

### 4.1.3 Timer Interrupts

The timer can generate the following interrupts:

- Timer overflow interrupt
- Real-time interrupt

Setting the I bit in the condition code register disables timer interrupts.

#### 4.1.3.1 Timer Overflow Interrupt

A timer overflow interrupt request occurs if the timer overflow flag, TOF, becomes set while the timer overflow interrupt enable bit, TOIE, is also set. See **8.1 Timer Status and Control Register (TSCR)**.

#### 4.1.3.2 Real-Time Interrupt

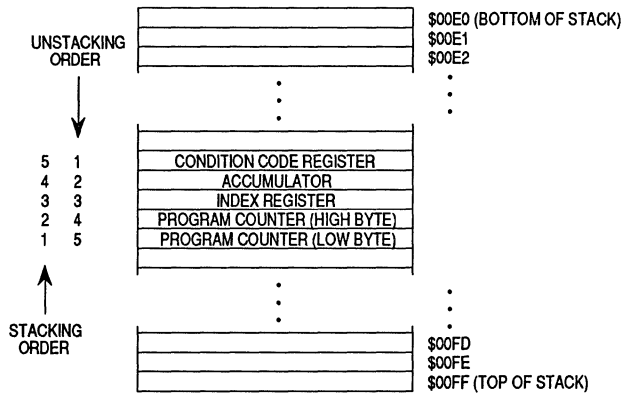
A real-time interrupt request occurs if the real-time interrupt flag, RTIF, becomes set while the real-time interrupt enable bit, RTIE, is also set. RTIF and RTIE are in the timer status and control register. See **8.1 Timer Status and Control Register (TSCR)**.

## 4.2 Interrupt Processing

The CPU does the following things to begin servicing an interrupt:

- Stores the CPU registers on the stack in the order shown in Figure 4-3
- Sets the I bit in the condition code register to prevent further interrupts
- Loads the program counter with the contents of the appropriate interrupt vector locations:
  - \$03FC and \$03FD (software interrupt vector)
  - \$03FA and \$03FB (external interrupt vector)
  - \$03F8 and \$03F9 (timer interrupt vector)

The return from interrupt (RTI) instruction causes the CPU to recover the CPU registers from the stack as shown in Figure 4-3.



**Figure 4-3. Interrupt Stacking Order**

Table 4-1 summarizes the reset and interrupt sources and vector assignments.

**Table 4-1. Reset/Interrupt Vector Addresses**

Function	Source	Local Mask	Global Mask	Priority (1 = High)	Vector Address
Reset	Power-On RESET Pin COP Watchdog Low Voltage Illegal Address	None	None None COPEN <sup>†</sup> LVIE <sup>†</sup> None	1 1 1 1 1	\$03FE-\$03FF
Software Interrupt (SWI)	User Code	None	None	Same Priority As Instruction	\$03FC-\$03FD
External Interrupt	$\overline{\text{IRQ}}/\text{V}_{\text{PP}}$ Pin PA3 Pin PA2 Pin PA1 Pin PA0 Pin	IRQE Bit	I Bit	2	\$03FA-\$03FB
Timer Interrupts	TOF Bit RTIF Bit	TOFE Bit RTIE Bit	I Bit	3	\$03F8-\$03F9

<sup>†</sup>COPEN and LVIE are EPROM/OTPROM bits in the mask option register.

Figure 4-4 shows the sequence of events caused by an interrupt.

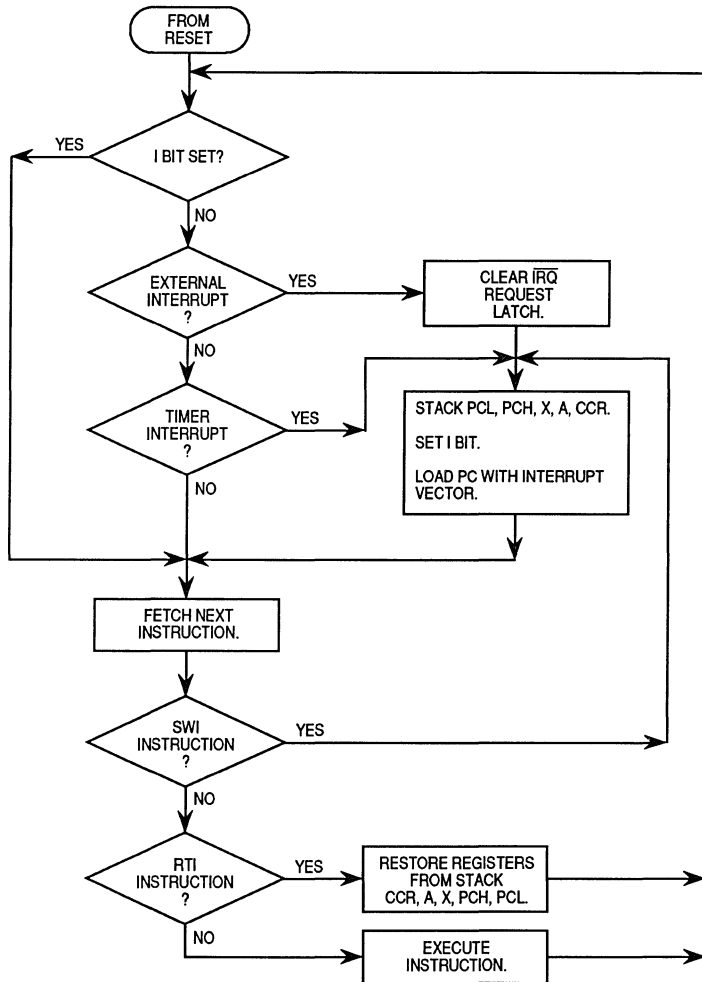


Figure 4-4. Interrupt Flowchart



## SECTION 5 RESETS

This section describes the five reset sources and how they initialize the MCU.

### 5.1 Reset Types

A reset immediately stops the operation of the instruction being executed, initializes certain control bits and loads the program counter with a user-defined reset vector address. The following conditions produce a reset:

- Power-on reset — Initial power-up
- External reset — A logic zero applied to the  $\overline{\text{RESET}}$  pin
- COP reset — Timeout of the COP watchdog (when the COP watchdog is enabled)
- Illegal address reset — An opcode fetch from an address not in the memory map
- Low voltage reset —  $V_{DD}$  voltage below 3.5 V (when the low voltage reset function is enabled)

Figure 5-1 is a block diagram of the reset function.

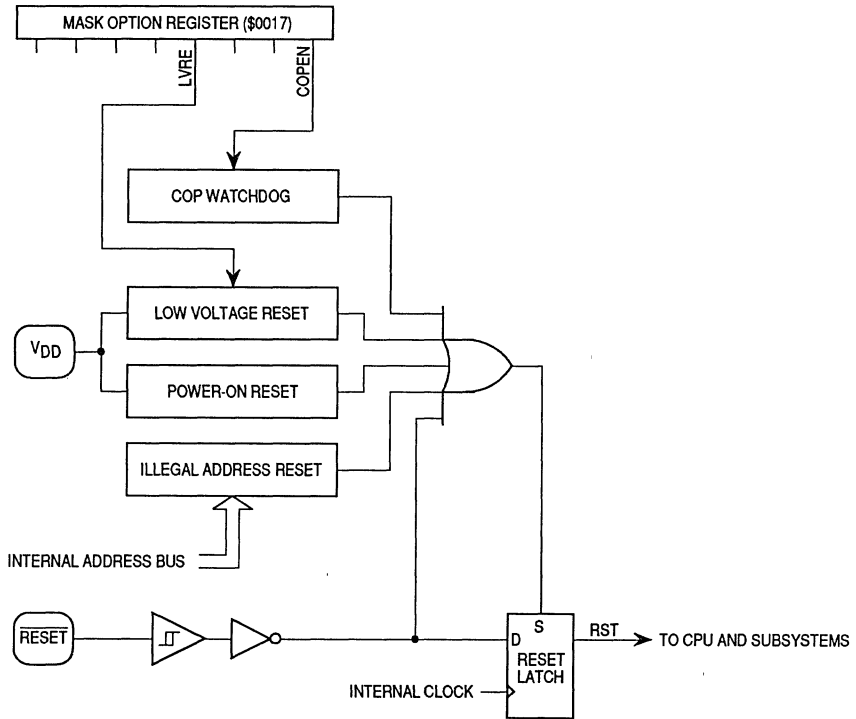
#### 5.1.1 Power-On Reset

A positive transition on the  $V_{DD}$  pin generates a power-on reset. The power-on reset is strictly for power-up conditions and cannot be used to detect drops in power supply voltage.

A 4064  $t_{CYC}$  (internal clock cycle) delay after the oscillator becomes active allows the clock generator to stabilize. If the  $\overline{\text{RESET}}$  pin is at logic zero at the end of 4064  $t_{CYC}$ , the MCU remains in the reset condition until the signal on the  $\overline{\text{RESET}}$  pin goes to logic one.

#### 5.1.2 External Reset

A logic zero applied to the  $\overline{\text{RESET}}$  pin for one and one-half  $t_{CYC}$  generates an external reset. A Schmitt trigger senses the logic level at the  $\overline{\text{RESET}}$  pin.



**Figure 5-1. Reset Sources**

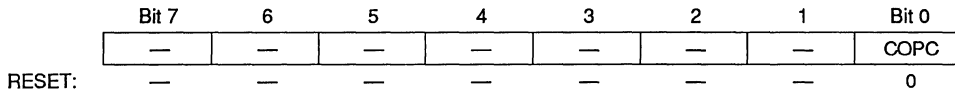
### 5.1.3 Computer Operating Properly (COP) Reset

A timeout of the COP watchdog generates a COP reset. The COP watchdog is part of a software error detection system and must be cleared periodically to start a new timeout period. (See **8.3 COP Watchdog**.) To clear the COP watchdog and prevent a COP reset, write a logic zero to bit 0 (COPC) of the COP register at location \$03F0. The COP register, shown in Figure 5-2, is a write-only register that returns the contents of a ROM location when read.

When erased, the COPEN bit in the mask option register disables the COP watchdog. Programming the COPEN bit to a logic one enables the COP watchdog. See **9.4 Mask Option Register (MOR)**.

**COPR** — COP Register

**\$03F0**



**Figure 5-2. COP Register (COPR)**

**COPC** — COP Clear

COPC is a write-only bit. Periodically writing a logic zero to COPC prevents the COP watchdog from resetting the MCU.

#### **5.1.4 Illegal Address Reset**

An opcode fetch from an address that is not in the EPROM (locations \$0200–\$03FF) or the RAM (\$00E0–\$00FF) generates an illegal address reset.

#### **5.1.5 Low Voltage Reset**

The low voltage reset circuit generates a reset signal if the voltage on the  $V_{DD}$  pin falls below 3.5 V (nominal).  $V_{DD}$  must be set at 5 V  $\pm$ 10% while the low voltage reset circuit is enabled.

Programming the LVRE bit to a logic one enables the low voltage reset function. When erased, the LVRE bit in the mask option register disables the low voltage reset circuit. See **9.4 Mask Option Register (MOR)**.

## 5.2 Reset States

The following paragraphs describe how resets initialize the MCU.

### 5.2.1 CPU

A reset has the following effects on the CPU:

- Loads the stack pointer with \$FF
- Sets the I bit in the condition code register, inhibiting interrupts
- Sets the IRQE bit in the interrupt status and control register
- Loads the program counter with the user-defined reset vector from locations \$03FE and \$03FF
- Clears the stop latch, enabling the CPU clock
- Clears the wait latch, waking the CPU from the wait mode

### 5.2.2 I/O Port Registers

A reset has the following effects on I/O port registers:

- Clears port A data direction register bits DDRA7–DDRA0 so that port A pins are inputs
- Clears port A pulldown register bits PDIA7–PDIA0, turning on the port A pulldown transistors
- Clears port B data direction register bits DDRB1–DDR0 so that both port B pins are inputs
- Clears port B pulldown register bits PDIB1–PDIB0, turning on the port B pulldown transistors
- Has no effect on port A or port B data registers

### 5.2.3 Multifunction Timer

A reset has the following effects on the multifunction timer.

- Clears the timer status and control register
- Clears the timer counter register

### 5.2.4 COP Watchdog

A reset clears the COP watchdog.

## SECTION 6 LOW POWER MODES

This section describes the four low-power modes:

- Stop mode
- Wait mode
- Halt mode
- Data-retention mode

### 6.1 Stop Mode

The STOP instruction puts the MCU in its lowest power-consumption mode and has the following effects on the MCU:

- Clears TOF and RTIF, the timer interrupt flags in the timer status and control register, removing any pending timer interrupts
- Clears TOIE and RTIE, the timer interrupt enable bits in the timer status and control register, disabling further timer interrupts
- Clears the multifunction timer counter
- Sets the IRQE bit in the IRQ status and control register to enable external interrupts
- Clears the I bit in the condition code register, enabling interrupts
- Stops the internal oscillator, turning off the CPU clock and the timer clock, including the COP watchdog

The STOP instruction does not affect any other registers or any I/O lines.

The following conditions bring the MCU out of stop mode:

- An external interrupt signal on the  $\overline{\text{IRQ}}/\text{V}_{\text{PP}}$  pin — A high-to-low transition on the  $\overline{\text{IRQ}}/\text{V}_{\text{PP}}$  pin loads the program counter with the contents of locations \$03FA and \$03FB.
- An external interrupt signal on a port A external interrupt pin — If the PIRQ bit in the mask option register is programmed to logic one, a low-to-high transition on a PA3–PA0 pin loads the program counter with the contents of locations \$03FA and \$03FB.
- External reset — A logic zero on the  $\overline{\text{RESET}}$  pin resets the MCU and loads the program counter with the contents of locations \$03FE and \$03FF.

When the MCU exits stop mode, processing resumes after a stabilization delay of 4064 oscillator cycles.

## 6.2 Wait Mode

The WAIT instruction puts the MCU in an intermediate power-consumption mode and has the following effects on the MCU:

- Clears the I bit in the condition code register, enabling interrupts
- Sets the IRQE bit in the IRQ status and control register, enabling external interrupts
- Stops the CPU clock, but allows the internal oscillator and timer clock to continue to run

The WAIT instruction does not affect any other registers or any I/O lines.

The following conditions restart the CPU clock and bring the MCU out of wait mode:

- An external interrupt signal on the  $\overline{\text{IRQ}}/\text{V}_{\text{PP}}$  pin — A high-to-low transition on the  $\overline{\text{IRQ}}/\text{V}_{\text{PP}}$  pin loads the program counter with the contents of locations \$03FA and \$03FB.
- An external interrupt signal on a port A external interrupt pin — If the PIRQ bit in the mask option register is programmed to logic one, a low-to-high transition on a PA3–PA0 pin loads the program counter with the contents of locations \$03FA and \$03FB.
- A timer interrupt — A timer overflow or a real-time interrupt request loads the program counter with the contents of locations \$03F8 and \$03F9.

- A COP watchdog reset — A timeout of the COP watchdog resets the MCU and loads the program counter with the contents of locations \$03FE and \$03FF. Software can enable real-time interrupts so that the MCU can periodically exit wait mode to reset the COP watchdog.
- External reset — A logic zero on the  $\overline{\text{RESET}}$  pin resets the MCU and loads the program counter with the contents of locations \$03FE and \$03FF.

### 6.3 Halt Mode

The STOP instruction puts the MCU in halt mode if the SWAIT bit in the mask option register is programmed to a logic one. The halt mode is identical to the wait mode, except that a recovery delay of 1–4064 internal clock cycles occurs when the MCU exits the halt mode. When the SWAIT bit is set, the COP watchdog cannot be inadvertently turned off by a STOP instruction.

Figure 6-1 shows the sequence of events in stop, wait, and halt modes.

### 6.4 Data-Retention Mode

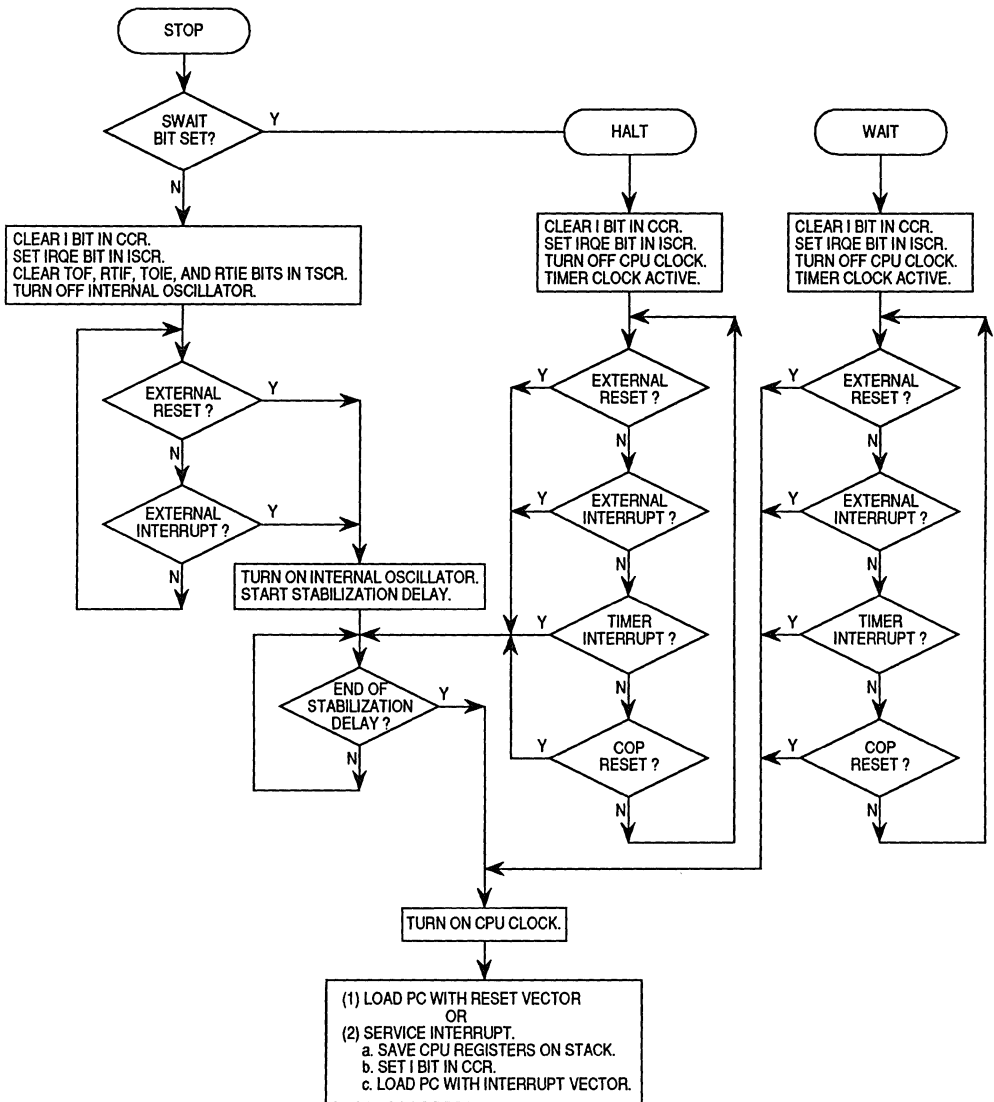
In data-retention mode, the MCU retains RAM contents and CPU register contents at  $V_{DD}$  voltages as low as 2.0 Vdc. The data-retention feature allows the MCU to remain in a low power-consumption state during which it retains data, but the CPU cannot execute instructions.

To put the MCU in data-retention mode:

1. Drive the  $\overline{\text{RESET}}$  pin to logic zero.
2. Lower the  $V_{DD}$  voltage. The  $\overline{\text{RESET}}$  pin must remain low continuously during data-retention mode.

To take the MCU out of data-retention mode:

1. Return  $V_{DD}$  to normal operating voltage.
2. Return the  $\overline{\text{RESET}}$  pin to logic one.



**Figure 6-1. Stop/Wait/Halt Flowchart**



## SECTION 7 PARALLEL I/O

This section describes the two bidirectional I/O ports.

### 7.1 I/O Port Function

The ten bidirectional I/O pins form two parallel I/O ports. Each I/O pin is programmable as an input or an output. The contents of the data direction registers determine the data direction for each I/O pin.

All ten I/O pins have software-programmable pulldown transistors.

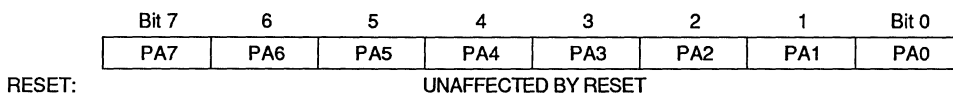
### 7.2 Port A

Port A is an 8-bit general-purpose bidirectional I/O port with the following features:

- Programmable pulldown transistors
- 8 mA current sinking capability (pins PA7–PA4)
- External interrupt capability (pins PA3–PA0)

#### 7.2.1 Port A Data Register (PORTA)

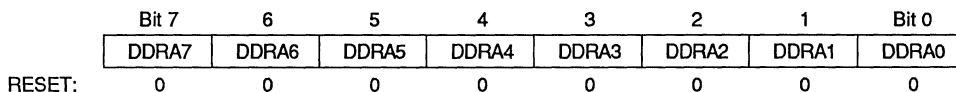
The port A data register contains a bit for each of the port A pins. When a port A pin is programmed to be an output, the state of its data register bit determines the state of the output pin. When a port A pin is programmed to be an input, reading the port A data register returns the logic state of the pin. Figure 7-1 shows the port A data register.

**PORTA** — Port A Data Register**\$0000****Figure 7-1. Port A Data Register (PORTA)****PA7–PA0** — Port A Data Bits

These read/write bits are software-programmable. Data direction of each bit is under the control of the corresponding data direction register bit.

**7.2.2 Data Direction Register A (DDRA)**

The contents of data direction register A determine whether each port A pin is an input or an output. See Figure 7-2. Writing a 1 to a DDRA bit enables the output buffer for the associated port A pin; a 0 disables the output buffer. A reset initializes all DDRA bits to 0, configuring all port A pins as inputs.

**DDRA** — Data Direction Register A**\$0004****Figure 7-2. Data Direction Register A (DDRA)****DDRA7–DDRA0** — Port A Data Direction Bits

These read/write bits control port A data direction.

1 = Corresponding port A pin configured as output

0 = Corresponding port A pin configured as input

**NOTE**

Avoid glitches on port A pins by writing to the port A data register before changing DDRA bits from 0 to 1.

### 7.2.3 Pulldown Register A (PDRA)

All port A pins have programmable pulldown transistors that typically sink 100  $\mu$ A. Clearing the PDIA7–PDIA0 bits in pulldown register A turns on the pulldown transistors. See Figure 7-3. Pulldown register A can turn on a port A pulldown transistor only when the port A pin is an input. Reset clears the PDIA7–PDIA0 bits, turning on all the port A pulldown transistors.

**PDRA** — Pulldown Register A

**\$0010**

	Bit 7	6	5	4	3	2	1	Bit 0
	PDIA7	PDIA6	PDIA5	PDIA4	PDIA3	PDIA2	PDIA1	PDIA0
RESET:	0	0	0	0	0	0	0	0

**Figure 7-3. Pulldown Register A (PDRA)**

**PDIA7–PDIA0** — Port A Pulldown Inhibit Bits 7–0

Writing logic zeros to these write-only bits turns on the port A pulldown transistors. Reading pulldown register A returns undefined data.

- 1 = Corresponding port A pin pulldown transistor turned off
- 0 = Corresponding port A pin pulldown transistor turned on

Programming the SWPDI bit in the mask option register to logic one turns off all port A and port B pulldown transistors and disables software control of the pulldown transistors. See **9.4 Mask Option Register (MOR)**.

#### **NOTE**

Avoid a floating port A input by clearing its pulldown register bit before changing its DDRA bit from 1 to 0.

#### **NOTE**

Do not use read-modify-write instructions on pulldown register A.

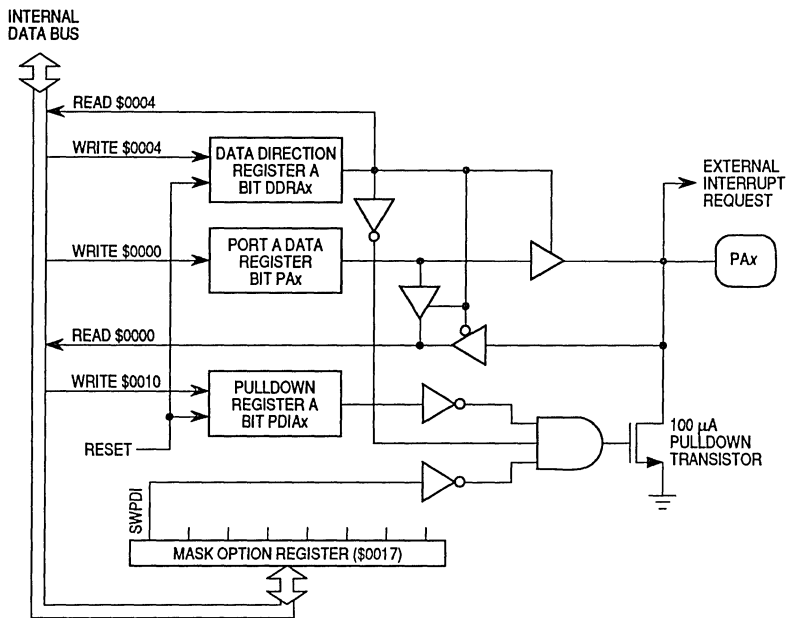
### 7.2.4 Port A External Interrupts

Programming the PIRQ bit in the mask option register to logic one enables the PA3–PA0 pins to serve as external interrupt pins in addition to the  $\overline{\text{IRQ}}/\text{V}_{\text{PP}}$  pin. The active interrupt state for the PA3–PA0 pins is a logic one or a rising edge. The active interrupt state for the  $\overline{\text{IRQ}}/\text{V}_{\text{PP}}$  pin is a logic zero or a falling edge. The state of the LEVEL bit in the mask option register determines whether external interrupt inputs are edge-sensitive only or both edge- and level-sensitive. See 9.4 Mask Option Register (MOR).

#### NOTE

When testing for external interrupts, the BIH and BIL instructions test the voltage on the  $\overline{\text{IRQ}}/\text{V}_{\text{PP}}$  pin, not the state of the internal IRQ signal. Therefore, BIH and BIL instructions cannot test the port A external interrupt pins.

Figure 7-4 shows the port A I/O logic.



**Figure 7-4. Port A I/O Circuit**

When a port A pin is programmed as an output, reading the port bit actually reads the value of the data latch and not the voltage on the pin itself. When a port A pin is programmed as an input, reading the port bit reads the voltage level on the pin. The data latch can always be written, regardless of the state of its DDR bit. Table 7-1 summarizes the operations of the port A pins.

**Table 7-1. Port A Pin Functions**

SWPDI	PDIAx	DDRAx	I/O Pin Mode	Accesses to PDRA		Accesses to DDRA	Accesses to PORTA	
				Read	Write	Read/Write	Read	Write
1	X	0	Input, Hi-Z	U	PDIA7-0	DDRA7-0	Pin	PA7-0
1	X	1	Output	U	PDIA7-0	DDRA7-0	PA7-0	PA7-0
0	0	0	Input, Pulldown On	U	PDIA7-0	DDRA7-0	Pin	PA7-0
0	0	1	Output, Pulldown On	U	PDIA7-0	DDRA7-0	PA7-0	PA7-0
0	1	0	Input, Hi-Z	U	PDIA7-0	DDRA7-0	Pin	PA7-0
0	1	1	Output	U	PDIA7-0	DDRA7-0	PA7-0	PA7-0

**NOTES:**

1. X = don't care.
2. U = undefined.

### 7.3 Port B

Port B is a 2-bit general-purpose bidirectional I/O port with the following features:

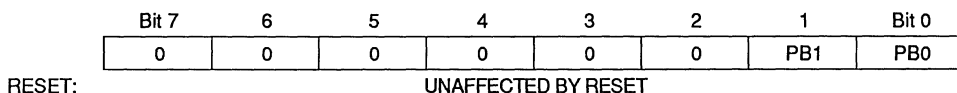
- Programmable pulldown transistors
- Three-pin oscillator connection

#### 7.3.1 Port B Data Register (PORTB)

The port B data register contains a bit for each of the port B pins. When a port B pin is programmed to be an output, the state of its data register bit determines the state of the output pin. When a port B pin is programmed to be an input, reading the port B data register returns the logic state of the pin. Figure 7-5 shows the port B data register.

#### PORTB — Port B Data Register

**\$0001**



**Figure 7-5. Port B Data Register (PORTB)**

#### PB1–PB0 — Port B Data Bits

These read/write bits are software-programmable. Data direction of each bit is under the control of the corresponding DDRB bit.

#### Bits 7–2 — Not used

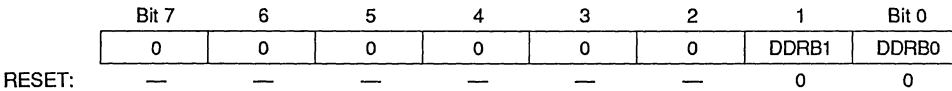
Bits 7–2 always read as logic zeros. Writes to these bits are ignored.

### 7.3.2 Data Direction Register B (DDRB)

The contents of DDRB determine whether each port B pin is an input or an output. See Figure 7-6. Writing a 1 to a DDRB bit enables the output buffer for the associated port B pin; a 0 disables the output buffer. A reset initializes all DDRB bits to 0, configuring all port B pins as inputs.

**DDRB** — Data Direction Register B

**\$0005**



**Figure 7-6. Data Direction Register B (DDRB)**

**DDRB1–DDRB0** — Port B Data Direction Bits

These read/write bits control port B data direction.

1 = Corresponding port B pin configured as output

0 = Corresponding port B pin configured as input

**Bits 7–2** — Not used

Bits 7–2 always read as logic zeros.

#### **NOTE**

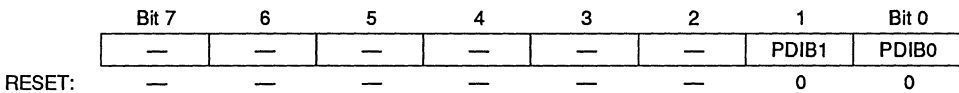
Avoid glitches on port B pins by writing to the port B data register before changing DDRB bits from 0 to 1.

### 7.3.3 Pulldown Register B (PDRB)

Both port B pins have programmable pulldown transistors that typically sink 100  $\mu$ A. Clearing the PDIB1 and PDIB0 bits in pulldown register B turns on the pulldown transistors. See Figure 7-7. Pulldown register B can turn on a port B pulldown transistor only when the port B pin is an input. Reset clears bits PDIB1 and PDIB0, turning on the port B pulldown transistors.

**PDRB** — Pulldown Register B

**\$0011**



**Figure 7-7. Pulldown Register B (PDRB)**

**PDIB1, PDIB0** — Port B Pulldown Inhibit Bits 1 and 0

Writing logic zeros to these write-only bits turns on the port B pulldown transistors.

Reading pulldown register B returns undefined data.

1 = Corresponding port B pin pulldown transistor turned off

0 = Corresponding port B pin pulldown transistor turned on

**Bits 7–2** — Not used

Bits 7–2 always read as logic zeros.

Programming the SWPDI bit in the mask option register to logic one turns off all port A and port B pulldown transistors and disables software control of the pulldown transistors. See **9.4 Mask Option Register (MOR)**.

#### **NOTE**

Avoid a floating port B input by clearing its pulldown register bit before changing its DDRB bit from 1 to 0.

#### **NOTE**

Do not use read-modify-write instructions on pulldown register B.



Figure 7-8 shows the port B I/O logic.

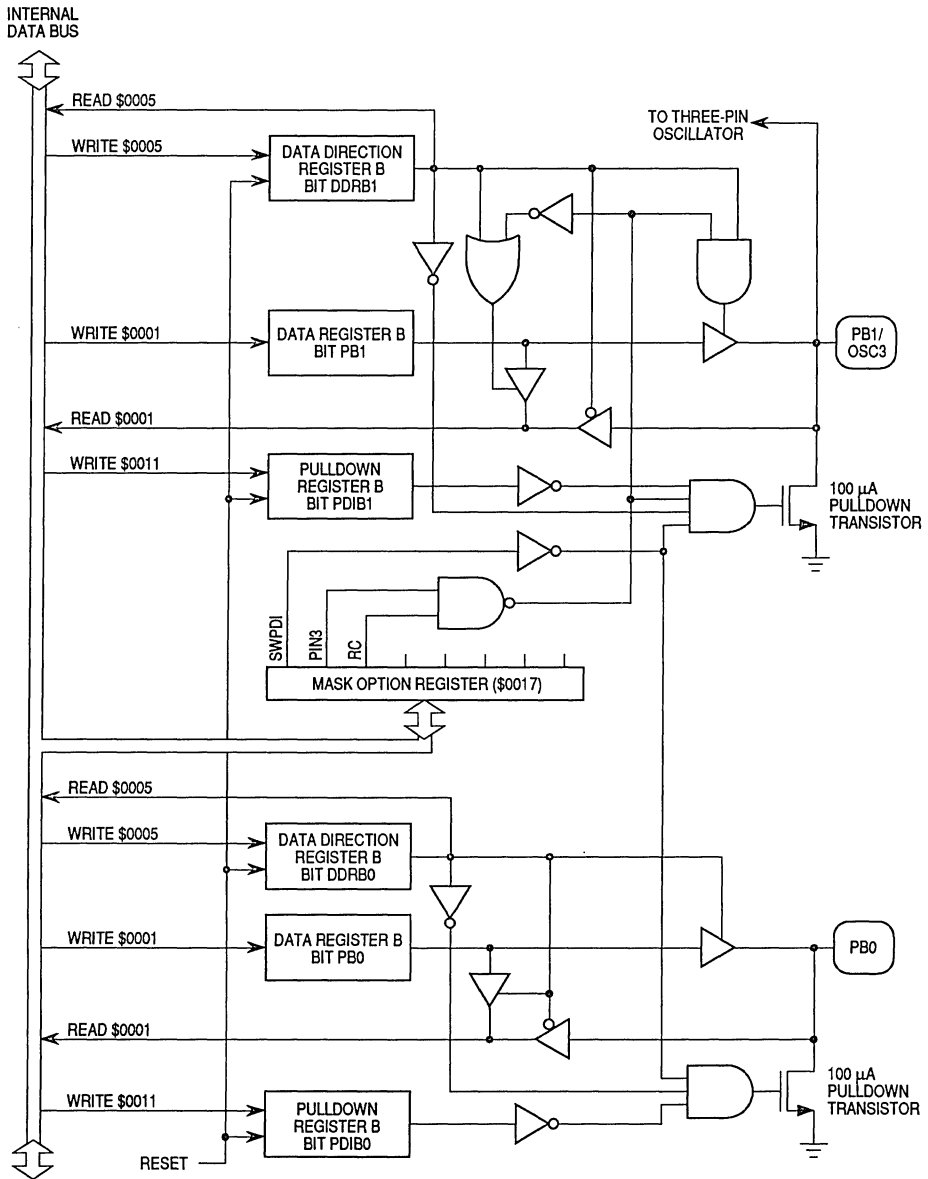


Figure 7-8. Port B I/O Circuit

When a port B pin is programmed as an output, reading the port bit reads the value of the data latch and not the voltage on the pin itself. When a port B pin is programmed as an input, reading the port bit reads the voltage level on the pin. The data latch can always be written, regardless of the state of its DDR bit. Table 7-2 and Table 7-3 summarize the operation of the port B pins.

**Table 7-2. PB0 Pin Functions**

SWPDI	PDIB0	DDRB0	PB0 Pin Mode	Accesses to PDRB		Accesses to DDRB	Accesses to PORTB	
				Read	Write	Read/Write	Read	Write
1	X	0	Input, Hi-Z	U	PDIB0	DDRB0	Pin	PB0
1	X	1	Output	U	PDIB0	DDRB0	PB0	PB0
0	0	0	Input, Pulldown On	U	PDIB0	DDRB0	Pin	PB0
0	0	1	Output, Pulldown On	U	PDIB0	DDRB0	PB0	PB0
0	1	0	Input, Hi-Z	U	PDIB0	DDRB0	Pin	PB0
0	1	1	Output	U	PDIB0	DDRB0	PB0	PB0

NOTES:

1. X = don't care
2. U = undefined

**Table 7-3. PB1/OSC3 Pin Functions**

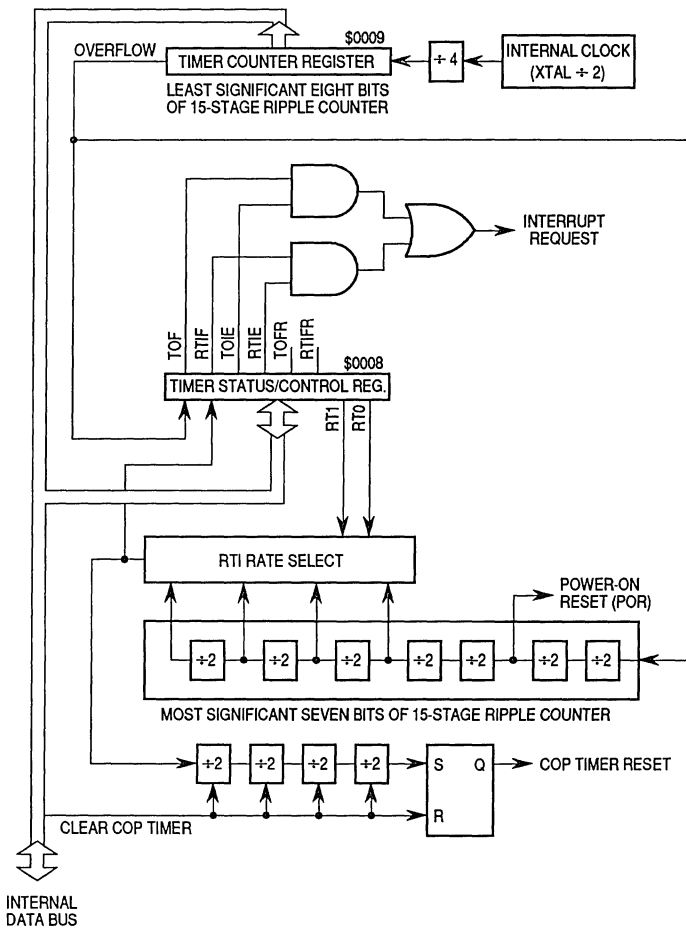
MOR Bits			PDIB1	DDRB1	I/O Pin Mode	Accesses to PDRB		Accesses to DDRB	Accesses to PORTB	
RC	PIN3	SWPDI				Read	Write	Read/Write	Read	Write
0	X	1	X	0	Input, Hi-Z	U	PDIB1	DDRB1	Pin	PB1
0	X	1	X	1	Output	U	PDIB1	DDRB1	PB1	PB1
0	X	0	0	0	Input, Pulldown On	U	PDIB1	DDRB1	Pin	PB1
0	X	0	0	1	Output, Pulldown On	U	PDIB1	DDRB1	PB1	PB1
0	X	0	1	0	Input, Hi-Z	U	PDIB1	DDRB1	Pin	PB1
0	X	0	1	1	Output	U	PDIB1	DDRB1	PB1	PB1
1	0	1	X	0	Input, Hi-Z	U	PDIB1	DDRB1	Pin	PB1
1	0	1	X	1	Output	U	PDIB1	DDRB1	PB1	PB1
1	0	0	0	0	Input, Pulldown On	U	PDIB1	DDRB1	Pin	PB1
1	0	0	0	1	Output, Pulldown On	U	PDIB1	DDRB1	PB1	PB1
1	0	0	1	0	Input, Hi-Z	U	PDIB1	DDRB1	Pin	PB1
1	0	0	1	1	Output	U	PDIB1	DDRB1	PB1	PB1
1	1	X	X	X	RC Oscillator Connection	U	PDRB1	DDRB1	PB1	PB1

NOTES:

1. X = don't care
2. U = undefined
3. In three-pin oscillator configuration, DDRB1 and PB1 are available as read/write storage locations; DDRB1 is cleared by reset; PB1 is not affected by reset.
4. In three-pin oscillator configuration, the PB1 pulldown is disabled regardless of the SWPDI bit.

## SECTION 8 MULTIFUNCTION TIMER

This section describes the operation of the timer and the COP watchdog. Figure 8-1 shows the organization of the timer subsystem.



**Figure 8-1. Timer Block Diagram**

## 8.1 Timer Status and Control Register (TSCR)

Timer interrupt flags, timer interrupt enable bits, and real-time interrupt rate select bits are in the read/write timer status and control register.

**TSCR** — Timer Status and Control Register

**\$0008**

	Bit 7	6	5	4	3	2	1	Bit 0
	TOF	RTIF	TOIE	RTIE	TOFR	RTIFR	RT1	RT0
RESET:	0	0	0	0	—	—	1	1

**Figure 8-2. Timer Status and Control Register (TSCR)**

### TOF — Timer Overflow Flag

This read-only bit becomes set when the first eight stages of the counter roll over from \$FF to \$00. TOF generates a timer overflow interrupt request if TOIE is also set. Clear TOF by writing a logic one to the TOFR bit. Writing to TOF has no effect.

### RTIF — Real-Time Interrupt Flag

This read-only bit becomes set when the selected RTI output becomes active. RTIF generates a real-time interrupt request if RTIE is also set. Clear RTIF by writing a logic one to the RTIFR bit. Writing to RTIF has no effect.

### TOIE — Timer Overflow Interrupt Enable

This read/write bit enables timer overflow interrupts.

1 = Timer overflow interrupts enabled

0 = Timer overflow interrupts disabled

### RTIE — Real-Time Interrupt Enable

This read/write bit enables real-time interrupts

1 = Real-time interrupts enabled

0 = Real-time interrupts disabled

**TOFR — Timer Overflow Flag Reset**

Writing a logic one to this write-only bit clears the TOF bit. TOFR always reads as logic zero.

**RTIFR — Real-Time Interrupt Flag Reset**

Setting this write-only bit clears the RTIF bit. RTIFR always reads as logic zero.

**RT1, RT0 — Real-Time Interrupt Select 1 and 0**

These read/write bits select one of four real-time interrupt rates. See Table 8-1. Because the selected RTI output drives the COP watchdog, changing the real-time interrupt rate also changes the counting rate of the COP watchdog.

**NOTE**

Changing RT1 and RT0 when a COP timeout is imminent or uncertain may cause a real-time interrupt to be missed or an additional real-time interrupt to be generated. Clear the COP timer just before changing RT1 and RT0.

**Table 8-1. Real-Time Interrupt Rate Selection**

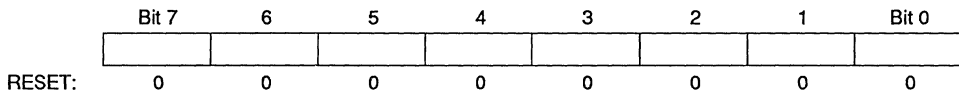
RT1:RT0	RTI Rate	RTI Period ( $f_{OP} = 2 \text{ MHz}$ )	COP Timeout Period (-0/+1 RTI Period)	Minimum COP Timeout Period ( $f_{OP} = 2 \text{ MHz}$ )
00	$f_{OP} + 2^{14}$	8.2 ms	$8 \times \text{RTI Period}$	66 ms
01	$f_{OP} + 2^{15}$	16.4 ms	$8 \times \text{RTI Period}$	131 ms
10	$f_{OP} + 2^{16}$	32.8 ms	$8 \times \text{RTI Period}$	262 ms
11	$f_{OP} + 2^{17}$	65.5 ms	$8 \times \text{RTI Period}$	524 ms

## 8.2 Timer Counter Register (TCNTR)

A 15-stage ripple counter is the core of the timer. The value of the first eight stages is readable at any time from the read-only timer counter register shown in Figure 8-3.

**TCNTR** — Timer Counter Register

**\$0009**



**Figure 8-3. Timer Counter Register (TCNTR)**

Power-on clears the entire counter chain and begins clocking the counter. After 4064 cycles, the power-on reset circuit is released, clearing the counter again and allowing the MCU to come out of reset.

A timer overflow function at the eighth counter stage allows a timer interrupt every 1024 internal clock cycles.

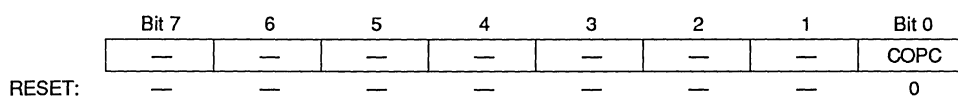
Each count of the timer counter register takes eight internal oscillator cycles or four cycles of the internal clock.

### 8.3 COP Watchdog

Four counter stages at the end of the timer make up the computer operating properly (COP) watchdog. (See Figure 8-1.) The COP watchdog is a software error detection system that automatically times out and resets the MCU if not cleared periodically by a program sequence. Writing a logic zero to bit 0 of the COP register clears the COP watchdog and prevents a COP reset. When erased, the COPEN bit in the mask option register disables the COP watchdog. Programming the COPEN bit to a logic one enables the COP watchdog.

**COPR** — COP Register

**\$03F0**



**Figure 8-4. COP Register (COPR)**

**COPC** — COP Clear

This write-only bit resets the COP watchdog. Reading address \$03F0 returns the ROM data at that address.

The COP watchdog is active in the run, wait, and halt modes of operation if the COPEN bit in the mask option register is set.

The STOP instruction turns off the COP watchdog. In applications that depend on the COP watchdog, the STOP instruction can be disabled by programming the SWAIT bit in the mask option register to logic one. In applications that have wait cycles longer than the COP timeout period, the COP watchdog can be disabled by not programming the COPEN bit to logic one in the mask option register.

#### **NOTE**

A voltage exceeding  $2 \times V_{DD}$  on the  $\overline{IRQ}/V_{PP}$  pin turns off the COP watchdog.

Table 8-2 summarizes recommended conditions for enabling and disabling the COP watchdog.

**Table 8-2. COP Watchdog Recommendations**

<b>Voltage on IRQ/V<sub>PP</sub> Pin</b>	<b>SWAIT Bit</b>	<b>WAIT/HALT Time</b>	<b>Recommended COP Watchdog Condition</b>
Less than $2 \times V_{DD}$	1	Less than COP Timeout Period	Enabled
Less than $2 \times V_{DD}$	1	Greater than COP Timeout Period	Disabled
Less than $2 \times V_{DD}$	0	X	Disabled
More than $2 \times V_{DD}$	X	X	Automatically Disabled

NOTES:

1. X = don't care
2. SWAIT bit in mask option register (MOR) converts stop instructions to halt instructions.



## SECTION 9 EPROM/OTPROM

This section describes how to program the 504-byte EPROM/OTPROM.

### NOTE

In packages with no quartz window, the 504 bytes of EPROM function as one-time programmable ROM (OTPROM).

### 9.1 EPROM Programming Register (EPROG)

The EPROM programming register shown in Figure 9-1 contains the control bits for programming the EPROM/OTPROM. In normal operation, the EPROM programming register is a read-only register that contains all logic zeros.

**EPROG** — EPROM Programming Register

**\$0018**

	Bit 7	6	5	4	3	2	1	Bit 0
	0	0	0	0	0	ELAT	MPGM	EPGM
RESET:	—	—	—	—	—	0	0	0

**Figure 9-1. EPROM Programming Register (EPROG)**

**ELAT** — EPROM Bus Latch

This read/write bit configures address and data buses for programming the EPROM/OTPROM array. EPROM/OTPROM data cannot be read when ELAT is set. Clearing the ELAT bit also clears the EPGM bit.

1 = Address and data buses configured for EPROM/OTPROM programming

0 = Address and data buses configured for normal operation

#### MPGM — Mask Option Register (MOR) Programming

This read/write bit applies programming power from the  $\overline{\text{IRQ}}/V_{PP}$  pin to the MOR.

1 = MOR programming power switched on

0 = MOR programming power switched off

#### EPGM — EPROM Programming

This read/write bit applies the voltage from the  $\overline{\text{IRQ}}/V_{PP}$  pin to the EPROM/OTPROM. To write the EPGM bit, the ELAT bit must already be set.

1 = EPROM/OTPROM programming power switched on

0 = EPROM/OTPROM programming power switched off

### NOTE

Writing logic ones to both the ELAT and EPGM bits with a single instruction sets ELAT and clears EPGM. ELAT must be set first by a separate instruction.

#### Bits 7–3 — Reserved

Bits 7–3 are factory test bits that always read as logic zeros.

## 9.2 EPROM/OTPROM Programming

Factory-provided software for programming the EPROM/OTPROM is available through the Motorola Freeware Bulletin Board Service (BBS). The number is (512) 891-FREE. After making the connection, type bbs in lowercase letters and press the return key to start the BBS software.

The programming software copies to the 496-byte space located at EPROM/OTPROM addresses \$0200–\$03EF, to the 8-byte space at addresses \$03F8–\$03FF, and to the mask option register at address \$0017.

Figure 9-2 shows the circuit used to download to the on-chip EPROM/OTPROM using the factory-provided programming software.

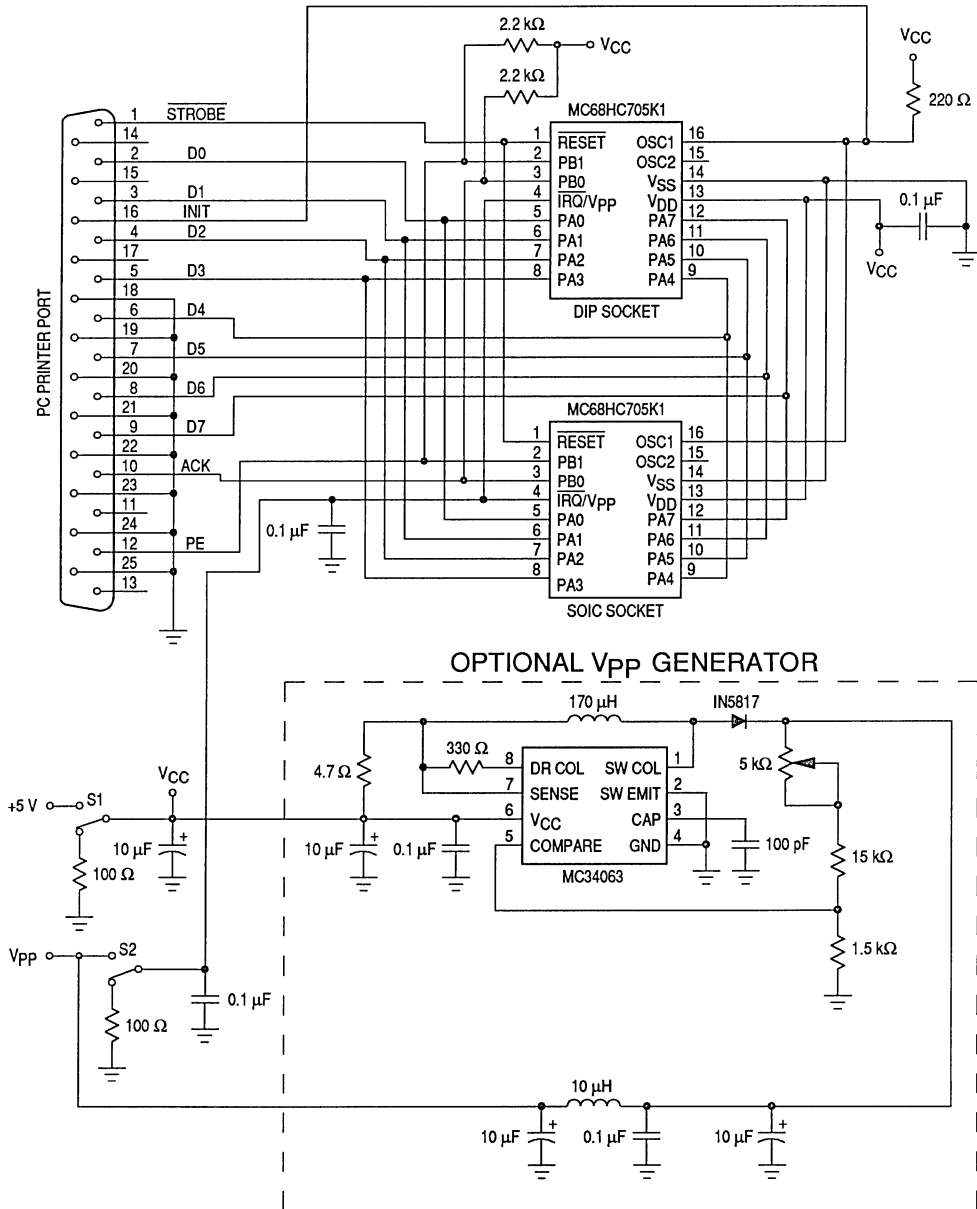


Figure 9-2. Programming Circuit

The following sequence shows the steps in programming a byte of EPROM/OTPROM:

1. Switch S1 powers up the MC68HC705K1.
2. Software synchronizes the external oscillator to the internal clock.
3. Switch S2 applies  $V_{PP}$  to the  $\overline{IRQ}/V_{PP}$  pin.
4. Software sets the ELAT bit.
5. Software writes to an EPROM/OTPROM address.
6. Software sets the EPGM bit for a time  $t_{EPGM}$  to apply the programming voltage.
7. Software clears the ELAT bit.

### 9.3 EPROM Erasing

MCUs with windowed packages permit EPROM erasure with ultraviolet light. Erase the EPROM by exposing it to  $15 \text{ Ws/cm}^2$  of ultraviolet light with a wavelength of 2537 angstroms. Position the ultraviolet light source 1 inch from the window. Do not use a shortwave filter. The erased state of an EPROM bit is logic zero.

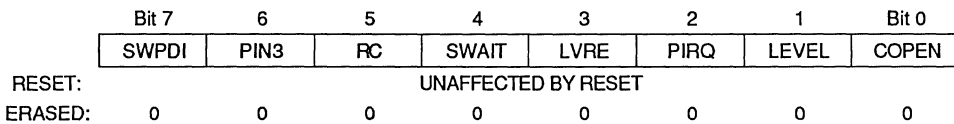
### 9.4 Mask Option Register (MOR)

The mask option register shown in Figure 9-3 is an EPROM/OTPROM byte that controls the following options:

- Port A and port B software programmable pulldown transistors (enabled or disabled)
- Oscillator connections (two-pin or three-pin RC oscillator)
- Oscillator connections (RC oscillator or crystal/ceramic resonator)
- STOP instruction (enable or disable)
- Low voltage reset (enable or disable)
- Port A external interrupt function (enable or disable)
- IRQ trigger sensitivity (edge-triggered only or both edge- and level-triggered)
- COP watchdog (enable or disable)

**MOR** — Mask Option Register

**\$0017**



**Figure 9-3. Mask Option Register (MOR)**

#### SWPDI — Software Pulldown Inhibit

This EPROM bit inhibits software control of the port A and port B pulldown transistors.

- 1 = Software pulldown inhibited
- 0 = Software pulldown enabled

#### PIN3 — Three-Pin RC Oscillator

This EPROM bit configures the on-chip oscillator as either a three-pin oscillator or as a two-pin oscillator. Bit PIN3 should be cleared when the RC bit is clear.

- 1 = Three-pin oscillator configured
- 0 = Two-pin oscillator configured

#### RC — RC Oscillator

This EPROM configures the on-chip oscillator for an external RC network.

1 = Oscillator configured for external RC network

0 = Oscillator configured for external crystal, ceramic resonator, or clock source

#### SWAIT — Stop Conversion to Wait

This EPROM bit disables the STOP instruction and prevents inadvertently turning off the COP watchdog with a STOP instruction. When the SWAIT bit is set, a STOP instruction puts the MCU in halt mode. Halt mode is a wait-like low-power state. The internal oscillator and timer clock continue to run, but the CPU clock stops. When the SWAIT bit is clear, a STOP instruction stops the internal oscillator, the internal clock, the CPU clock, and the timer clock.

1 = STOP instruction converted to WAIT instruction

0 = STOP instruction not converted to WAIT instruction

#### LVRE — Low Voltage Reset Enable

This EPROM bit enables the low voltage reset (LVR) circuit.

1 = LVR circuit enabled

0 = LVR circuit disabled

#### PIRQ — Port A IRQ Enable

This EPROM bit enables the PA3–PA0 pins to function as external interrupt sources.

1 = PA3–PA0 enabled as external interrupt sources

0 = PA3–PA0 not enabled as external interrupt sources

#### LEVEL — External Interrupt Sensitivity

This EPROM bit makes the external interrupt inputs level-triggered as well as edge-triggered.

1 =  $\overline{\text{IRQ}}/\text{V}_{\text{PP}}$  pin negative edge-triggered and low level-triggered

PA3–PA0 pins positive-edge triggered and high level-triggered

0 =  $\overline{\text{IRQ}}/\text{V}_{\text{PP}}$  pin negative edge-triggered only

PA3–PA0 pins positive-edge triggered only

#### COPEN — COP Watchdog Enable

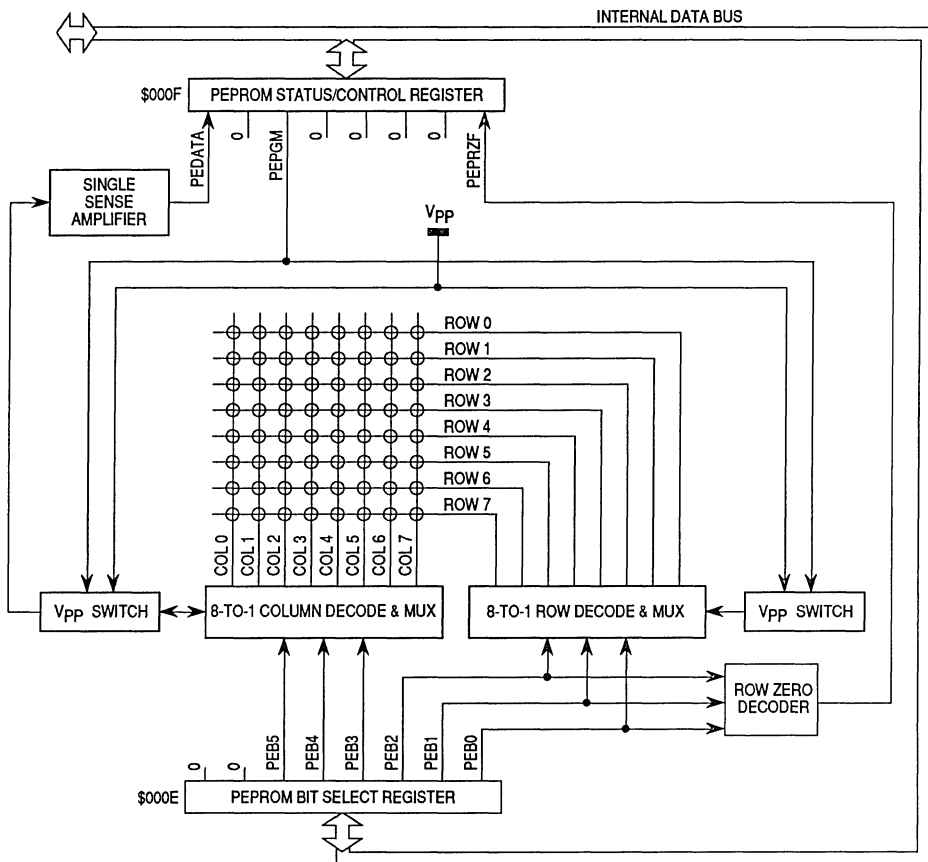
This EPROM bit enables the COP watchdog.

1 = COP watchdog enabled

0 = COP watchdog disabled

## SECTION 10 PERSONALITY EPROM

This section describes how to program the 64-bit personality EPROM (PEPROM). Figure 10-1 shows the structure of the PEPROM subsystem.



**Figure 10-1. Personality EPROM**

## 10.1 PEPROM Registers

Two I/O registers control programming and reading of the PEPROM:

- The PEPROM bit select register (PEBSR)
- The PEPROM status and control register (PESCR)

### 10.1.1 PEPROM Bit Select Register (PEBSR)

The PEPROM bit select register shown in Figure 10-2 selects one of 64 bits in the PEPROM array.

**PEBSR** — PEPROM Bit Select Register

**\$000E**

	Bit 7	6	5	4	3	2	1	Bit 0
	PEB7	PEB6	PEB5	PEB4	PEB3	PEB2	PEB1	PEB0
RESET:	0	0	0	0	0	0	0	0

**Figure 10-2. PEPROM Bit Select Register (PEBSR)**

PEB7, PEB6 — Not used

These read/write bits are available as storage locations.

PEB5–PEB0 — PEPROM Bit Select Bits

These read/write bits select one of 64 bits in the personality EPROM as shown in Table 10-1. Bits PEB2–0 select the PEPROM row, and bits PEB5–3 select the PEPROM column.



**Table 10-1. PEPROM Bit Selection**

PEBSR	PEPROM Bit Selected	
\$00	Row 0	Column 0
\$01	Row 1	Column 0
\$02	Row 2	Column 0
.	.	.
.	.	.
.	.	.
\$08	Row 0	Column 1
\$09	Row 1	Column 1
\$0A	Row 2	Column 1
.	.	.
.	.	.
.	.	.
\$10	Row 0	Column 2
\$11	Row 1	Column 2
\$12	Row 2	Column 2
.	.	.
.	.	.
.	.	.
\$38	Row 0	Column 7
\$39	Row 1	Column 7
\$3A	Row 2	Column 7
\$3B	Row 3	Column 7
\$3C	Row 4	Column 7
\$3D	Row 5	Column 7
\$3E	Row 6	Column 7
\$3F	Row 7	Column 7

Bits 6 and 7 — Not used

Bits 6 and 7 are not connected and can be used as read/write storage locations.

### 10.1.2 PEPROM Status and Control Register (PESCR)

The PEPROM status and control register shown in Figure 10-3 controls the PEPROM programming voltage. This register also transfers the PEPROM bits to the internal data bus and contains a row zero flag.

**PESCR — PEPROM Status and Control Register** **\$000F**

	Bit 7	6	5	4	3	2	1	Bit 0
	PEDATA	0	PEPGM	0	0	0	0	PEPRZF
RESET:	0	0	0	0	0	0	0	1

**Figure 10-3. PEPROM Status and Control Register (PESCR)**

#### PEDATA — PEPROM Data

This read-only bit is the state of the PEPROM sense amplifier and shows the state of the currently selected bit.

#### PEPGM — PEPROM Program Control

This read/write bit controls the switches that apply the programming voltage,  $V_{PP}$ , to the selected PEPROM cell.

- 1 = Programming voltage applied
- 0 = Programming voltage not applied

#### PEPRZF — PEPROM Row Zero Flag

This read-only bit is set when the PEPROM bit select register selects the first row (row zero) of the PEPROM array. Selecting any other row clears PEPRZF. Monitoring PEPRZF can reduce the code needed to access one byte of PEPROM.

- 1 = Row zero selected
- 0 = Row zero not selected

## 10.2 PEPROM Programming

Factory-provided software for programming the PEPROM is available through the Motorola Freeware Bulletin Board Service (BBS). The number is (512) 891-FREE. After making the connection, type bbs in lowercase letters and press the return key to start the BBS software.

The same circuit shown in **9.2 EPROM/OTPROM Programming** can be used to program the PEPROM with the factory-provided programming software.

The PEPROM can also be programmed by user software with  $V_{PP}$  applied to the  $\overline{IRQ}/V_{PP}$  pin. The following sequence shows how to program each PEPROM bit:

1. Select a PEPROM bit by writing to PEBSR.
2. Set the PEPGM bit in PESCR.
3. Wait 3 ms.
4. Clear the PEPGM bit.

### NOTE

While the PEPGM bit is set and  $V_{PP}$  is on the  $\overline{IRQ}/V_{PP}$  pin, do not access bits that are to be left unprogrammed (erased).

## 10.3 PEPROM Reading

The following sequence shows how to read the PEPROM:

1. Select a bit by writing to PEBSR.
2. Read the PEDATA bit in the PESCR.
3. Store PEDATA bit in RAM or in a register.
4. Select another bit by changing PEBSR.
5. Continue storing the PEDATA bits until the required personality EPROM data is stored.
6. Continue reading and storing PEPROM data.

Reading the PEPROM is easiest when each PEPROM column contains one byte. Selecting a row 0 bit selects the first bit, and incrementing the PEPROM bit select register (PEBSR) selects the next (row 1) bit from the same column. Incrementing PEBSR seven more times selects the remaining bits of the column and selects the row 0 bit of the next column, setting the row 0 flag, PEPRZF.

A PEPROM byte that has been read can be transferred to the personality EPROM bit select register (PEBSR) so that subsequent reads of the PEBSR will quickly yield that PEPROM byte.

#### **10.4 PEPROM Erasing**

MCUs with windowed packages permit PEPROM erasure with ultraviolet light. Erase the PEPROM by exposing it to 15 Ws/cm<sup>2</sup> of ultraviolet light with a wavelength of 2537 angstroms. Position the ultraviolet light source 1 inch from the window. Do not use a shortwave filter. The erased state of a PEPROM bit is logic zero.

## SECTION 11 INSTRUCTION SET

This section describes the addressing modes and the types of instructions.

### 11.1 Addressing Modes

The CPU uses eight addressing modes for flexibility in accessing data. These addressing modes define the manner in which the CPU finds the data required to execute an instruction. The eight addressing modes are as follows:

- Inherent
- Immediate
- Direct
- Extended
- Indexed, no offset
- Indexed, 8-bit offset
- Indexed, 16-bit offset
- Relative

#### 11.1.1 Inherent

Inherent instructions are those that have no operand, such as return from interrupt (RTI) and stop (STOP). Some of the inherent instructions act on data in the CPU registers, such as set carry flag (SEC) and increment accumulator (INCA). Inherent instructions require no memory address and are one byte long. Table 11-1 lists the instructions that use the inherent addressing mode.

**Table 11-1. Inherent Addressing Instructions**

<b>Instruction</b>	<b>Mnemonic</b>
Arithmetic Shift Left	ASLA, ASLX
Arithmetic Shift Right	ASRA, ASRX
Clear Carry Bit	CLC
Clear Interrupt Mask	CLI
Clear	CLRA, CLRX
Complement	COMA, COMX
Decrement	DECA, DECX
Increment	INCA, INCX
Logical Shift Left	LSLA, LSLX
Logical Shift Right	LSRA, LSRX
Multiply	MUL
Negate	NEGA, NEGX
No Operation	NOP
Rotate Left through Carry	ROLA, ROLX
Rotate Right through Carry	RORA, RORX
Reset Stack Pointer	RSP
Return from Interrupt	RTI
Return from Subroutine	RTS
Set Carry Bit	SEC
Set Interrupt Mask	SEI
Enable IRQ and Stop Oscillator	STOP
Software Interrupt	SWI
Transfer Accumulator to Index Register	TAX
Test for Negative or Zero	TSTA, TSTX
Transfer Index Register to Accumulator	TXA
Enable Interrupt and Half Processor	WAIT

### 11.1.2 Immediate

Immediate instructions are those that contain a value to be used in an operation with the value in the accumulator or index register. Immediate instructions require no memory address and are two bytes long. The opcode is the first byte and the immediate data value is the second byte. Table 11-2 lists the instructions that use the immediate addressing mode.

**Table 11-2. Immediate Addressing Instructions**

<b>Instruction</b>	<b>Mnemonic</b>
Add with Carry	ADC
Add	ADD
Logical AND	AND
Bit Test Memory with Accumulator	BIT
Compare Accumulator with Memory	CMP
Compare Index Register with Memory	CPX
Exclusive OR Memory with Accumulator	EOR
Load Accumulator from Memory	LDA
Load Index Register from Memory	LDX
Inclusive OR	ORA
Subtract with Carry	SBC
Subtract	SUB

### **11.1.3 Direct**

Direct instructions can access any of the first 256 memory addresses with only two bytes. The first byte is the opcode and the second byte is the low byte of the operand address. In the direct addressing mode, the CPU automatically uses \$00 as the high byte of the operand address. BRSET and BRCLR are three-byte instructions that use direct addressing to access the operand and relative addressing to specify a branch destination. Table 11-3 lists the instructions that use the direct addressing mode.

**Table 11-3. Direct Addressing Instructions**

<b>Instruction</b>	<b>Mnemonic</b>
Add with Carry	ADC
Add	ADD
Logical AND	AND
Arithmetic Shift Left	ASL
Arithmetic Shift Right	ASR
Clear Bit in Memory	BCLR
Bit Test Memory with Accumulator	BIT
Branch if Bit n Is Clear	BRCLR
Branch if Bit n Is Set	BRSET
Set Bit in Memory	BSET
Clear	CLR
Compare Accumulator with Memory	CMP
Complement	COM
Compare Index Register with Memory	CPX
Decrement	DEC
Exclusive OR Memory with Accumulator	EOR
Increment	INC
Jump	JMP
Jump to Subroutine	JSR
Load Accumulator from Memory	LDA
Load Index Register from Memory	LDX
Logical Shift Left	LSL
Logical Shift Right	LSR
Negate	NEG
Inclusive OR	ORA
Rotate Left through Carry	ROL
Rotate Right through Carry	ROR
Subtract with Carry	SBC
Store Accumulator in Memory	STA
Store Index Register in Memory	STX
Subtract	SUB
Test for Negative or Zero	TST



#### 11.1.4 Extended

Extended instructions can access any address in memory with only three bytes. The first byte is the opcode; the second and third bytes are the high and low bytes of the operand address.

When using the Motorola assembler, the programmer does not need to specify whether an instruction is direct or extended. The assembler automatically selects the shortest form of the instruction. Table 11-4 lists the instructions that use the extended addressing mode.

**Table 11-4. Extended Addressing Instructions**

<b>Instruction</b>	<b>Mnemonic</b>
Add with Carry	ADC
Add	ADD
Logical AND	AND
Bit Test Memory with Accumulator	BIT
Compare Accumulator with Memory	CMP
Compare Index Register with Memory	CPX
Exclusive OR Memory with Accumulator	EOR
Jump	JMP
Jump to Subroutine	JSR
Load Accumulator from Memory	LDA
Load Index Register from Memory	LDX
Inclusive OR	ORA
Subtract with Carry	SBC
Store Accumulator in Memory	STA
Store Index Register in Memory	STX
Subtract	SUB

### 11.1.5 Indexed, No Offset

Indexed instructions with no offset are one-byte instructions that can access data with variable addresses within the first 256 memory locations. The index register contains the low byte of the operand conditional address. The CPU automatically uses \$00 as the high byte of the operand conditional address, so these instructions can address locations \$0000–\$00FF.

Indexed, no offset instructions are often used to move a pointer through a table or to hold the address of a frequently used RAM or I/O location. Table 11-5 lists the instructions that use the indexed, no offset addressing mode.

### 11.1.6 Indexed, 8-Bit Offset

Indexed, 8-bit offset instructions are two-byte instructions that can access data with variable addresses within the first 511 memory locations. The CPU adds the unsigned byte in the index register to the unsigned byte following the opcode. The sum is the conditional address of the operand. These instructions can address locations \$0000–\$01FE.

Indexed, 8-bit offset instructions are useful for selecting the *k*th element in an *n*-element table. The table can begin anywhere within the first 256 memory locations and could extend as far as location 510 (\$01FE). The *k* value would typically be in the index register, and the address of the beginning of the table would be in the byte following the opcode. Table 11-5 lists the instructions that use the indexed, 8-bit offset addressing mode.

### 11.1.7 Indexed, 16-Bit Offset

Indexed, 16-bit offset instructions are three-byte instructions that can access data with variable addresses at any location in memory. The CPU adds the unsigned byte in the index register to the two unsigned bytes following the opcode. The sum is the conditional address of the operand. The first byte after the opcode is the high byte of the 16-bit offset; the second byte is the low byte of the offset. These instructions can address any location in memory.

Indexed, 16-bit offset instructions are useful for selecting the *k*th element in an *n*-element table anywhere in memory.

As with direct and extended addressing, the Motorola assembler determines the shortest form of indexed addressing. Table 11-5 lists the instructions that can use the indexed, 16-bit offset addressing mode.

**Table 11-5. Indexed Addressing Instructions**

Instruction	Mnemonic	No Offset	8-Bit Offset	16-Bit Offset
Add with Carry	ADC	√	√	√
Add	ADD	√	√	√
Logical AND	AND	√	√	√
Arithmetic Shift Left	ASL	√	√	
Arithmetic Shift Right	ASR	√	√	
Bit Test Memory with Accumulator	BIT	√	√	√
Clear	CLR	√	√	
Compare Accumulator with Memory	CMP	√	√	√
Complement	COM	√	√	
Compare Index Register with Memory	CPX	√	√	√
Decrement	DEC	√	√	
Exclusive OR Memory with Accumulator	EOR	√	√	√
Increment	INC	√	√	
Jump	JMP	√	√	√
Jump to Subroutine	JSR	√	√	√
Load Accumulator from Memory	LDA	√	√	√
Load Index Register from Memory	LDX	√	√	√
Logical Shift Left	LSL	√	√	
Logical Shift Right	LSR	√	√	
Negate	NEG	√	√	
Inclusive OR	ORA	√	√	√
Rotate Left through Carry	ROL	√	√	
Rotate Right through Carry	ROR	√	√	
Subtract with Carry	SBC	√	√	√
Store Accumulator in Memory	STA	√	√	√
Store Index Register in Memory	STX	√	√	√
Subtract	SUB	√	√	√
Test for Negative or Zero	TST	√	√	

### 11.1.8 Relative

The relative addressing mode is only for branch instructions and bit test and branch instructions. The CPU finds the conditional branch destination by adding the signed byte following the opcode to the contents of the program counter if the branch condition is true. If the branch condition is not true, the CPU goes to the next instruction. To permit branching either forward or backward, the offset is a signed, two's complement byte that gives a branching range of -127 to +128 bytes from the address of the next location after the branch instruction.

When using the Motorola assembler, the programmer does not need to calculate the offset, because the assembler determines the proper offset and verifies that it is within the span of the branch. Table 11-6 lists the instructions that use the relative addressing mode.

**Table 11-6. Relative Addressing Instructions**

Instruction	Mnemonic
Branch if Carry Clear	BCC
Branch if Carry Set	BCS
Branch if Equal	BEQ
Branch if Half-Carry Clear	BHCC
Branch if Half-Carry Set	BHCS
Branch if Higher	BHI
Branch if Higher or Same	BHS
Branch if Interrupt Line High	BIH
Branch if Interrupt Line Low	BIL
Branch if Lower	BLO
Branch if Lower or Same	BLS
Branch if Interrupt Mask Clear	BMC
Branch if Minus	BMI
Branch if Interrupt Mask Set	BMS
Branch if Not Equal	BNE
Branch if Plus	BPL
Branch Always	BRA
Branch if Bit n Clear	BRCLR
Branch if Bit n Set	BRSET
Branch Never	BRN
Branch to Subroutine	BSR

## 11.2 Instruction Types

The MCU instructions fall into the following five categories:

- Register/memory
- Read-modify-write
- Jump/branch
- Bit manipulation
- Control

### 11.2.1 Register/Memory Instructions

Most of these instructions use two operands. One operand is in either the accumulator or the index register. The CPU finds the other operand in memory using one of the addressing modes. Most register/memory instructions use the following addressing modes:

- Immediate
- Direct
- Extended
- Indexed, no offset
- Indexed, 8-bit offset
- Indexed, 16-bit offset

Table 11-7 lists the register/memory instructions.

**Table 11-7. Register/Memory Instructions**

Instruction	Mnemonic
Load Accumulator from Memory	LDA
Load Index Register from Memory	LDX
Store Accumulator in Memory	STA
Store Index Register in Memory	STX
Add Memory to Accumulator	ADD
Add Memory and Carry to Accumulator	ADC
Subtract Memory	SUB
Subtract Memory from Accumulator with Borrow	SBC
AND Memory with Accumulator	AND
OR Memory with Accumulator	ORA
Arithmetic Compare Accumulator with Memory	CMP
Arithmetic Compare Index Register with Memory	CPX
Bit Test Memory with Accumulator (Logical Compare)	BIT
Multiply	MUL

### 11.2.2 Read-Modify-Write Instructions

These instructions read a memory location or a register, modify 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 because it does not write a replacement value. Read-modify-write instructions use the following addressing modes:

- Inherent
- Direct
- Indexed, no offset
- Indexed, 8-bit offset

Table 11-8 lists the read-modify-write instructions.

**Table 11-8. Read-Modify-Write Instructions**

Instruction	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (Two's Complement)	NEG
Rotate Left through Carry	ROL
Rotate Right through Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST

### 11.2.3 Jump/Branch Instructions

Jump instructions allow the CPU to interrupt the normal sequence of the program counter. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Jump instructions use the following addressing modes:

- Direct
- Extended
- Indexed, no offset
- Indexed, 8-bit offset
- Indexed, 16-bit offset

Branch instructions allow the CPU to interrupt the normal sequence of the program counter when a test condition is met. If the test condition is not met, the branch is not performed. All branch instructions are used in the relative addressing mode.

Bit test and branch instructions cause a branch based on the condition of any readable bit in the first 256 memory locations. These three-byte instructions use a combination of direct addressing and relative addressing. The direct address of the byte to be tested is in the byte following the opcode. The third byte is the signed offset byte. The CPU finds the conditional branch destination by adding the third byte to the program counter if the specified bit tests true. The bit to be tested and its condition (set or clear) is part of the opcode. The span of branching is from  $-128$  to  $+127$  from the address of the next location after the branch instruction. The CPU also transfers the tested bit to the carry/borrow bit of the condition code register. Table 11-9 lists the jump and branch instructions.

**Table 11-9. Jump and Branch Instructions**

Instruction	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Bit n of M = 0	BRCLR
Branch if Bit n of M = 1	BRSET
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	BND
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 Clear	BMC
Branch if Interrupt Mask Set	BMS
Branch if Interrupt Line Low	BIL
Branch if Interrupt Line High	BIH
Branch to Subroutine	BSR
Jump Unconditional	JMP
Jump to Subroutine	JSR

### 11.2.4 Bit Manipulation Instructions

The CPU can set or clear any writable bit in the first 256 bytes of memory. Port register, port data direction registers, timer registers, and on-chip RAM locations are in the first 256 bytes of memory. The CPU can also test and branch based on the state of any bit in any of the first 256 memory locations. Bit manipulation instructions use the direct addressing mode. Table 11-10 lists these instructions.

**Table 11-10. Bit Manipulation Instructions**

Instruction	Mnemonic
Set Bit n	BSET n (n = 0 . . . 7)
Clear Bit n	BCLR n (n = 0 . . . 7)
Branch if Bit n of M = 0	BRCLR
Branch if Bit n of M = 1	BRSET

### 11.2.5 Control Instructions

These register reference instructions control CPU operation during program execution. Control instructions, listed in Table 11-11, use the inherent addressing mode.

**Table 11-11. Control Instructions**

Instruction	Mnemonic
Transfer Accumulator to Index Register	TAX
Transfer Index Register to Accumulator	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask	SEI
Clear Interrupt Mask	CLI
Software Interrupt	SWI
Return from Subroutine	RTI
Reset Stack Pointer	RSP
No Operation	NOP
Stop	STOP
Wait	WAIT



### 11.3 Instruction Set Summary

Table 11-12 shows all MC68HC705K1 instructions in all possible addressing modes. For each instruction, the operand construction and the execution time in internal clock cycles ( $t_{CYC}$ ) are shown. One internal clock cycle equals two oscillator input cycles. The following legend summarizes the symbols and abbreviations used in Table 11-12.

#### Abbreviations and Symbols

A	Accumulator	PCH	Program counter high byte
C	Carry/borrow flag	PCL	Program counter low byte
CCR	Condition code register	REL	Relative addressing mode
dd	Address of operand in direct addressing	rel	Offset byte for relative addressing
dd rr	Address (dd) of operand and offset (rr) of branch instruction for bit test instructions	rr	Offset byte of branch instruction
DIR	Direct addressing mode	SP	Stack pointer
ee ff	High (ee) and low (ff) bytes of offset in indexed, 16-bit offset addressing	X	Index register
EXT	Extended addressing mode	Z	Zero flag
ff	Offset byte in indexed, 8-bit offset addressing	•	AND
H	Half-carry flag	—	Not affected
hh ll	High (hh) and low (ll) bytes of operand address in extended addressing	?	If
I	Interrupt mask	—	NOT
ii	Operand byte for immediate addressing	( )	Contents of
IMM	Immediate addressing mode	←	Is loaded with
INH	Inherent addressing mode	:	Concatenated with
IX	Indexed, no offset addressing mode	×	Multiplication
IX1	Indexed, 8-bit offset addressing mode	-( )	Negation (two's complement)
IX2	Indexed, 16-bit offset addressing mode	+	Inclusive OR
M	Any memory location (1 byte)	↕	Set if true; clear if not true
N	Negative flag	⊕	Exclusive OR
n	Any bit (7,6,5 . . . 0)	+	Addition
opr	Operand byte	-	Subtraction
PC	Program counter		

Table 11-12. Instruction Set (Sheet 1 of 4)

Source Form(s)	Operation	Description	Addressing Mode for Operand	Machine Coding (hexadecimal)		Cycles	Condition Code				
				Opcode	Operand		H	I	N	Z	C
ADC opr	Add with carry	$A \leftarrow (A) + (M) + C$	IMM DIR EXT IX2 IX1 IX	A9 B9 C9 D9 E9 F9	ii dd hh ll ee ff ff	2 3 4 5 4 3	↓	-	↓	↓	↓
ADD opr	Add without carry	$A \leftarrow (A) + (M)$	IMM DIR EXT IX2 IX1 IX	AB BB CB DB EB FB	ii dd hh ll ee ff ff	2 3 4 5 4 3	↓	-	↓	↓	↓
AND opr	Logical AND	$A \leftarrow (A) \bullet (M)$	IMM DIR EXT IX2 IX1 IX	A4 B4 C4 D4 E4 F4	ii dd hh ll ee ff ff	2 3 4 5 4 3	-	-	↓	↓	-
ASL opr ASLA ASLX ASL opr ASL opr	Arithmetic shift left		DIR INH INH IX1 IX	38 48 58 68 78	dd ff	5 3 3 6 5	-	-	↓	↓	↓
ASR opr ASRA ASRX ASR opr ASR opr	Arithmetic shift right		DIR INH INH IX1 IX	37 47 57 67 77	dd ff	5 3 3 6 5	-	-	↓	↓	↓
BCC rel	Branch if carry bit clear	? C = 0	REL	24	rr	3	-	-	-	-	-
BCLR n opr	Clear bit n	$M_n \leftarrow 0$	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	5 5 5 5 5 5 5 5	-	-	-	-	-
BCS rel	Branch if carry bit set	? C = 1	REL	25	rr	3	-	-	-	-	-
BEQ rel	Branch if equal	? Z = 1	REL	27	rr	3	-	-	-	-	-
BHCC rel	Branch if half carry bit clear	? H = 0	REL	28	rr	3	-	-	-	-	-
BHCS rel	Branch if half carry bit set	? H = 1	REL	29	rr	3	-	-	-	-	-
BHI rel	Branch if higher	? C + Z = 0	REL	22	rr	3	-	-	-	-	-
BHS rel	Branch if higher or same	? C = 0	REL	24	rr	3	-	-	-	-	-
BIH rel	Branch if IRQ pin high	? IRQ = 1	REL	2F	rr	3	-	-	-	-	-
BIL rel	Branch if IRQ pin low	? IRQ = 0	REL	2E	rr	3	-	-	-	-	-
BIT rel	Bit test accumulator contents with memory contents	$(A) \bullet (M)$	IMM DIR EXT IX2 IX1 IX	A5 B5 C5 D5 E5 F5	ii dd hh ll ee ff ff	2 3 4 5 4 3	-	-	↓	↓	-
BLO rel	Branch if lower	? C = 1	REL	25	rr	3	-	-	-	-	-
BLS rel	Branch if lower or same	? C + Z = 1	REL	23	rr	3	-	-	-	-	-
BMC rel	Branch if interrupt mask clear	? I = 0	REL	2C	rr	3	-	-	-	-	-
BMI rel	Branch if minus	? N = 1	REL	2B	rr	3	-	-	-	-	-
BMS rel	Branch if interrupt mask set	? I = 0	REL	2D	rr	3	-	-	-	-	-
BNE rel	Branch if not equal	? Z = 0	REL	26	rr	3	-	-	-	-	-
BPL rel	Branch if plus	? N = 0	REL	2A	rr	3	-	-	-	-	-

Table 11-12. Instruction Set (Sheet 2 of 4)

Source Form(s)	Operation	Description	Addressing Mode for Operand	Machine Coding (hexadecimal)		Cycles	Condition Code				
				Opcode	Operand		H	I	N	Z	C
BRA rel	Branch always	? 1 = 1	REL	20	rr	3	-	-	-	-	-
BRCLR n opr rel	Branch if bit n clear	? Mn = 0	DIR (b0)	01	dd rr	5	-	-	-	-	⚡
			DIR (b1)	03	dd rr	5					
			DIR (b2)	05	dd rr	5					
			DIR (b3)	07	dd rr	5					
			DIR (b4)	09	dd rr	5					
			DIR (b5)	0B	dd rr	5					
			DIR (b6)	0D	dd rr	5					
DIR (b7)	0F	dd rr	5								
BRN rel	Branch never	? 1 = 0	REL	21	rr	3	-	-	-	-	
BRSET n opr rel	Branch if bit n set	? Mn = 1	DIR (b0)	00	dd rr	5	-	-	-	-	⚡
			DIR (b1)	02	dd rr	5					
			DIR (b2)	04	dd rr	5					
			DIR (b3)	06	dd rr	5					
			DIR (b4)	08	dd rr	5					
			DIR (b5)	0A	dd rr	5					
			DIR (b6)	0C	dd rr	5					
DIR (b7)	0E	dd rr	5								
BSET n opr	Set bit n	Mn ← 1	DIR (b0)	10	dd	5	-	-	-	-	-
			DIR (b1)	12	dd	5					
			DIR (b2)	14	dd	5					
			DIR (b3)	16	dd	5					
			DIR (b4)	18	dd	5					
			DIR (b5)	1A	dd	5					
			DIR (b6)	1C	dd	5					
DIR (b7)	1E	dd	5								
BSR rel	Branch to subroutine	PC ← (PC) + 2; push (PCL) SP ← (SP) - 1; push (PCH) SP ← (SP) - 1 PC ← (PC) + rel	REL	AD	rr	6	-	-	-	-	
CLC	Clear carry bit	C ← 0	INH	98		2	-	-	-	0	
CLI	Clear interrupt mask	I ← 0	INH	9A		2	-	0	-	-	
CLR opr CLRA CLR X CLR opr CLR opr	Clear register	M ← \$00 A ← \$00 X ← \$00 M ← \$00 M ← \$00	DIR	3F	dd	5	-	-	0	1	-
			INH	4F		3					
			INH	5F		3					
			IX1	6F	ff	6					
			IX	7F		5					
CMP opr	Compare accumulator contents with memory contents	(A) - (M)	IMM	A1	ii	2	-	-	⚡	⚡	⚡
			DIR	B1	dd	3					
			EXT	C1	hh ll	4					
			IX2	D1	ee ff	5					
			IX1	E1	ff	4					
			IX	F1		3					
COM opr COMA COM X COM opr COM opr	Complement register contents (ones complement)	M ← M = \$FF - (M) A ← A = \$FF - (A) X ← X = \$FF - (X) M ← M = \$FF - (M) M ← M = \$FF - (M)	DIR	33	dd	5	-	-	⚡	⚡	1
			INH	43		3					
			INH	53		3					
			IX1	63	ff	6					
			IX	73		5					
CPX opr	Compare index register contents with memory contents	(X) - (M)	IMM	A3	ii	2	-	-	⚡	⚡	⚡
			DIR	B3	dd	3					
			EXT	C3	hh ll	4					
			IX2	D3	ee ff	5					
			IX1	E3	ff	4					
			IX	F3		3					
DEC opr DECA DEC X DEC opr DEC opr	Decrement register contents	M ← (M) - 1 A ← (A) - 1 X ← (X) - 1 M ← (M) - 1 M ← (M) - 1	DIR	3A	dd	5	-	-	⚡	⚡	-
			INH	4A		3					
			INH	5A		3					
			IX1	6A	ff	6					
			IX	7A		5					

Table 11-12. Instruction Set (Sheet 3 of 4)

Source Form(s)	Operation	Description	Addressing Mode for Operand	Machine Coding (hexadecimal)		Cycles	Condition Code				
				Opcode	Operand		H	I	N	Z	C
EOR opr	Exclusive OR accumulator contents with memory contents	$A \leftarrow (A) \oplus (M)$	IMM	A8	ii	2	-	-	↓	↓	-
			DIR	B8	dd	3					
			EXT	C8	hh ll	4					
			IX2	D8	ee ff	5					
			IX1	E8	ff	4					
IX	F8		3								
INC opr INCA INCX INC opr INC opr	Increment memory or register contents	$M \leftarrow (M) + 1$ $A \leftarrow (A) + 1$ $X \leftarrow (X) + 1$ $M \leftarrow (M) + 1$ $M \leftarrow (M) + 1$	DIR	3C	dd	5	-	-	↓	↓	-
			INH	4C		3					
			INH	5C		3					
			IX1	6C	ff	6					
			IX	7C		5					
JMP opr	Unconditional jump	PC ← jump address	DIR	BC	dd	2	-	-	-	-	-
			EXT	CC	hh ll	3					
			IX2	DC	ee ff	4					
			IX1	EC	ff	3					
			IX	FC		2					
JSR opr	Jump to subroutine	PC ← (PC) + n (n = 1, 2, or 3) Push (PCL); SP ← (SP) - 1 Push (PCH); SP ← (SP) - 1 PC ← conditional address	DIR	BD	dd	5	-	-	-	-	-
			EXT	CD	hh ll	6					
			IX2	DD	ee ff	7					
			IX1	ED	ff	6					
			IX	FD		5					
LDA opr	Load accumulator with memory contents	$A \leftarrow (M)$	IMM	A6	ii	2	-	-	↓	↓	-
			DIR	B6	dd	3					
			EXT	C6	hh ll	4					
			IX2	D6	ee ff	5					
			IX1	E6	ff	4					
IX	F6		3								
LDX opr	Load index register with memory contents	$X \leftarrow (M)$	IMM	AE	ii	2	-	-	↓	↓	-
			DIR	BE	dd	3					
			EXT	CE	hh ll	4					
			IX2	DE	ee ff	5					
			IX1	EE	ff	4					
IX	FE		3								
LSL opr LSLA LSLX LSL opr LSL opr	Logical shift left		DIR	38	dd	5	-	-	↓	↓	↓
			INH	48		3					
			INH	58		3					
			IX1	68	ff	6					
			IX	78		5					
LSR opr LSRA LSRX LSR opr LSR opr	Logical shift right		DIR	34	dd	5	-	-	0	↓	↓
			INH	44		3					
			INH	54		3					
			IX1	64	ff	6					
			IX	74		5					
MUL	Unsigned multiply	$X : A \leftarrow (X) \times (A)$	INH	42		11	0	-	-	-	0
NEG opr NEGA NEGX NEG opr NEG opr	Negate memory or register contents (two's complement)	$M \leftarrow \neg(M) = \$00 - (M)$ $A \leftarrow \neg(A) = \$00 - (A)$ $X \leftarrow \neg(X) = \$00 - (X)$ $M \leftarrow \neg(M) = \$00 - (M)$ $M \leftarrow \neg(M) = \$00 - (M)$	DIR	30	dd	5	-	-	↓	↓	↓
			INH	40		3					
			INH	50		3					
			IX1	60	ff	6					
			IX	70		5					
NOP	No operation		INH	9D		2	-	-	-	-	-
ORA opr	Inclusive OR accumulator contents with memory contents	$A \leftarrow (A) + (M)$	IMM	AA	ii	2	-	-	↓	↓	-
			DIR	BA	dd	3					
			EXT	CA	hh ll	4					
			IX2	DA	ee ff	5					
			IX1	EA	ff	4					
IX	FA		3								
ROL opr ROLA ROLX ROL opr ROL opr	Rotate left through carry		DIR	39	dd	5	-	-	↓	↓	↓
			INH	49		3					
			INH	59		3					
			IX1	69	ff	6					
			IX	79		5					

Table 11-12. Instruction Set (Sheet 4 of 4)

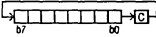
Source Form(s)	Operation	Description	Addressing Mode for Operand	Machine Coding (hexadecimal)		Cycles	Condition Code				
				Opcode	Operand		H	I	N	Z	C
ROR opr RORA RORX ROR opr ROR opr	Rotate right through carry		DIR INH INH IX1 IX	36 46 56 66 76	dd  ff	5 3 3 6 5	-	-	⚡	⚡	⚡
RSP	Reset stack pointer	$SP \leftarrow \$00FF$	INH	9C		2	From Stack				
RTI	Return from interrupt	$SP \leftarrow (SP) + 1$ ; pull (CCR) $SP \leftarrow (SP) + 1$ ; pull (A) $SP \leftarrow (SP) + 1$ ; pull (X) $SP \leftarrow (SP) + 1$ ; pull (PCH) $SP \leftarrow (SP) + 1$ ; pull (PCL)	INH	80		9	⚡	⚡	⚡	⚡	⚡
RTS	Return from subroutine	$SP \leftarrow (SP) + 1$ ; pull (PCH) $SP \leftarrow (SP) + 1$ ; pull (PCL)	INH	81		6	-	-	-	-	-
SBC opr	Subtract memory contents and carry bit from accumulator contents	$A \leftarrow (A) - (M) - C$	IMM DIR EXT IX2 IX1 IX	A2 B2 C2 D2 E2 F2	ii dd hh ll ee ff ff	2 3 4 5 4 3	-	-	⚡	⚡	⚡
SEC	Set carry bit	$C \leftarrow 1$	INH	99		2	-	-	-	-	1
SEI	Set interrupt mask	$I \leftarrow 1$	INH	9B		2	-	1	-	-	-
STA opr	Store accumulator contents in memory	$M \leftarrow (A)$	DIR EXT IX2 IX1 IX	B7 C7 D7 E7 F7	dd hh ll ee ff ff	4 5 6 5 4	-	-	⚡	⚡	-
STOP	Enable $\overline{IRQ}$ ; stop oscillator		INH	8E		2	-	0	-	-	-
STX opr	Store index register contents in memory	$M \leftarrow (X)$	DIR EXT IX2 IX1 IX	BF CF DF EF FF	dd hh ll ee ff ff	4 5 6 5 4	-	-	⚡	⚡	-
SUB opr	Subtract memory contents from accumulator contents	$A \leftarrow (A) - (M)$	IMM DIR EXT IX2 IX1 IX	A0 B0 C0 D0 E0 F0	ii dd hh ll ee ff ff	2 3 4 5 4 3	-	-	⚡	⚡	⚡
SWI	Software interrupt	$PC \leftarrow (PC) + 1$ ; push (PCL) $SP \leftarrow (SP) - 1$ ; push (PCH) $SP \leftarrow (SP) - 1$ ; push (X) $SP \leftarrow (SP) - 1$ ; push (A) $SP \leftarrow (SP) - 1$ ; push (CCR) $SP \leftarrow (SP) - 1$ ; $I \leftarrow 1$ PCH $\leftarrow$ Interrupt vector hi byte PCL $\leftarrow$ Int. vector low byte	INH	83		10	-	1	-	-	-
TAX	Transfer accumulator contents to index register	$X \leftarrow (A)$	INH	97		2	-	-	-	-	-
TST opr TSTA TSTX TST opr TST opr	Test memory, accumulator, or index register contents for negative or zero	$(M) - \$00$	DIR INH INH IX1 IX	3D 4D 5D 6D 7D	dd  ff	4 3 3 5 4	-	-	⚡	⚡	-
TXA	Transfer index register contents to accumulator	$A \leftarrow (X)$	INH	9F		2	-	-	-	-	-
WAIT	Enable interrupts; halt CPU		INH	8F		2	-	0	-	-	-

Table 11-13. Opcode Map

HI	Bit-Manipulation			Branch		Read-Modify-Write				Control		Register/Memory					HI	LO
	DIR	DIR	REL	DIR	INH	INH	IX1	IX	INH	INH	IMM	DIR	EXT	IX2	IX1	IX		
	0	0001	0010	3	4	5	6	7	8	9	A	B	C	D	E	F		
0	BRSET <sup>5</sup> <sub>0</sub>	BSET <sup>5</sup> <sub>0</sub>	BRA <sup>3</sup> <sub>REL</sub>	NEG <sup>5</sup> <sub>DIR</sub>	NEGA <sup>3</sup> <sub>INH</sub>		NEG <sup>6</sup> <sub>IX1</sub>	NEG <sup>5</sup> <sub>IX</sub>	RTI <sup>9</sup> <sub>INH</sub>		SUB <sup>2</sup> <sub>IMM</sub>	SUB <sup>3</sup> <sub>DIR</sub>	SUB <sup>4</sup> <sub>EXT</sub>	SUB <sup>5</sup> <sub>IX2</sub>	SUB <sup>4</sup> <sub>IX1</sub>	SUB <sup>3</sup> <sub>IX</sub>	0000	
1	BRCLR <sup>5</sup> <sub>0</sub>	BCLR <sup>5</sup> <sub>0</sub>	BRN <sup>3</sup> <sub>REL</sub>						RTS <sup>6</sup> <sub>INH</sub>		CMP <sup>2</sup> <sub>IMM</sub>	CMP <sup>3</sup> <sub>DIR</sub>	CMP <sup>4</sup> <sub>EXT</sub>	CMP <sup>5</sup> <sub>IX2</sub>	CMP <sup>4</sup> <sub>IX1</sub>	CMP <sup>3</sup> <sub>IX</sub>	0001	
2	BRSET <sup>5</sup> <sub>1</sub>	BSET <sup>5</sup> <sub>1</sub>	BHI <sup>3</sup> <sub>REL</sub>		MUL <sup>11</sup> <sub>INH</sub>						SBC <sup>2</sup> <sub>IMM</sub>	SBC <sup>3</sup> <sub>DIR</sub>	SBC <sup>4</sup> <sub>EXT</sub>	SBC <sup>5</sup> <sub>IX2</sub>	SBC <sup>4</sup> <sub>IX1</sub>	SBC <sup>3</sup> <sub>IX</sub>	0010	
3	BRCLR <sup>5</sup> <sub>1</sub>	BSCLR <sup>5</sup> <sub>1</sub>	BLS <sup>3</sup> <sub>REL</sub>	COM <sup>5</sup> <sub>DIR</sub>	COMA <sup>3</sup> <sub>INH</sub>	COMX <sup>3</sup> <sub>INH</sub>	COM <sup>6</sup> <sub>IX1</sub>	COM <sup>5</sup> <sub>IX</sub>	SWI <sup>10</sup> <sub>INH</sub>		CPX <sup>2</sup> <sub>IMM</sub>	CPX <sup>3</sup> <sub>DIR</sub>	CPX <sup>4</sup> <sub>EXT</sub>	CPX <sup>5</sup> <sub>IX2</sub>	CPX <sup>4</sup> <sub>IX1</sub>	CPX <sup>3</sup> <sub>IX</sub>	0011	
4	BRSET <sup>5</sup> <sub>2</sub>	BSET <sup>5</sup> <sub>2</sub>	BCC <sup>3</sup> <sub>REL</sub>	LSR <sup>5</sup> <sub>DIR</sub>	LSRA <sup>3</sup> <sub>INH</sub>	LSRX <sup>3</sup> <sub>INH</sub>	LSR <sup>6</sup> <sub>IX1</sub>	LSR <sup>5</sup> <sub>IX</sub>			AND <sup>2</sup> <sub>IMM</sub>	AND <sup>3</sup> <sub>DIR</sub>	AND <sup>4</sup> <sub>EXT</sub>	AND <sup>5</sup> <sub>IX2</sub>	AND <sup>4</sup> <sub>IX1</sub>	AND <sup>3</sup> <sub>IX</sub>	0100	
5	BRCLR <sup>5</sup> <sub>2</sub>	BCLR <sup>5</sup> <sub>2</sub>	BCS <sup>3</sup> <sub>REL</sub>								BIT <sup>2</sup> <sub>IMM</sub>	BIT <sup>3</sup> <sub>DIR</sub>	BIT <sup>4</sup> <sub>EXT</sub>	BIT <sup>5</sup> <sub>IX2</sub>	BIT <sup>4</sup> <sub>IX1</sub>	BIT <sup>3</sup> <sub>IX</sub>	0101	
6	BRSET <sup>5</sup> <sub>3</sub>	BSET <sup>5</sup> <sub>3</sub>	BNE <sup>3</sup> <sub>REL</sub>	ROR <sup>5</sup> <sub>DIR</sub>	RORA <sup>3</sup> <sub>INH</sub>	RORX <sup>3</sup> <sub>INH</sub>	ROR <sup>6</sup> <sub>IX1</sub>	ROR <sup>5</sup> <sub>IX</sub>			LDA <sup>2</sup> <sub>IMM</sub>	LDA <sup>3</sup> <sub>DIR</sub>	LDA <sup>4</sup> <sub>EXT</sub>	LDA <sup>5</sup> <sub>IX2</sub>	LDA <sup>4</sup> <sub>IX1</sub>	LDA <sup>3</sup> <sub>IX</sub>	0110	
7	BRCLR <sup>5</sup> <sub>3</sub>	BCLR <sup>5</sup> <sub>3</sub>	BEQ <sup>3</sup> <sub>REL</sub>	ASR <sup>5</sup> <sub>DIR</sub>	ASRA <sup>3</sup> <sub>INH</sub>	ASRX <sup>3</sup> <sub>INH</sub>	ASR <sup>6</sup> <sub>IX1</sub>	ASR <sup>5</sup> <sub>IX</sub>	TAX <sup>2</sup> <sub>INH</sub>		STA <sup>4</sup> <sub>DIR</sub>	STA <sup>5</sup> <sub>EXT</sub>	STA <sup>6</sup> <sub>IX2</sub>	STA <sup>5</sup> <sub>IX1</sub>	STA <sup>4</sup> <sub>IX</sub>	0111		
8	BRSET <sup>5</sup> <sub>4</sub>	BSET <sup>5</sup> <sub>4</sub>	BHCC <sup>3</sup> <sub>REL</sub>	LSL <sup>5</sup> <sub>DIR</sub>	LSLA <sup>3</sup> <sub>INH</sub>	LSLX <sup>3</sup> <sub>INH</sub>	LSL <sup>6</sup> <sub>IX1</sub>	LSL <sup>5</sup> <sub>IX</sub>		CLC <sup>2</sup> <sub>INH</sub>	EOR <sup>2</sup> <sub>IMM</sub>	EOR <sup>3</sup> <sub>DIR</sub>	EOR <sup>4</sup> <sub>EXT</sub>	EOR <sup>5</sup> <sub>IX2</sub>	EOR <sup>4</sup> <sub>IX1</sub>	EOR <sup>3</sup> <sub>IX</sub>	1000	
9	BRCLR <sup>5</sup> <sub>4</sub>	BCLR <sup>5</sup> <sub>4</sub>	BHCS <sup>3</sup> <sub>REL</sub>	ROL <sup>5</sup> <sub>DIR</sub>	ROLA <sup>3</sup> <sub>INH</sub>	ROLX <sup>3</sup> <sub>INH</sub>	ROL <sup>6</sup> <sub>IX1</sub>	ROL <sup>5</sup> <sub>IX</sub>	SEC <sup>2</sup> <sub>INH</sub>	ADC <sup>2</sup> <sub>IMM</sub>	ADC <sup>3</sup> <sub>DIR</sub>	ADC <sup>4</sup> <sub>EXT</sub>	ADC <sup>5</sup> <sub>IX2</sub>	ADC <sup>4</sup> <sub>IX1</sub>	ADC <sup>3</sup> <sub>IX</sub>	1001		
A	BRSET <sup>5</sup> <sub>5</sub>	BSET <sup>5</sup> <sub>5</sub>	BPL <sup>3</sup> <sub>REL</sub>	DEC <sup>5</sup> <sub>DIR</sub>	DECA <sup>3</sup> <sub>INH</sub>	DECX <sup>3</sup> <sub>INH</sub>	DEC <sup>6</sup> <sub>IX1</sub>	DEC <sup>5</sup> <sub>IX</sub>	CLI <sup>2</sup> <sub>INH</sub>	ORA <sup>2</sup> <sub>IMM</sub>	ORA <sup>3</sup> <sub>DIR</sub>	ORA <sup>4</sup> <sub>EXT</sub>	ORA <sup>5</sup> <sub>IX2</sub>	ORA <sup>4</sup> <sub>IX1</sub>	ORA <sup>3</sup> <sub>IX</sub>	1010		
B	BRCLR <sup>5</sup> <sub>5</sub>	BCLR <sup>5</sup> <sub>5</sub>	BMI <sup>3</sup> <sub>REL</sub>						SEI <sup>2</sup> <sub>INH</sub>	ADD <sup>2</sup> <sub>IMM</sub>	ADD <sup>3</sup> <sub>DIR</sub>	ADD <sup>4</sup> <sub>EXT</sub>	ADD <sup>5</sup> <sub>IX2</sub>	ADD <sup>4</sup> <sub>IX1</sub>	ADD <sup>3</sup> <sub>IX</sub>	1011		
C	BRSET <sup>5</sup> <sub>6</sub>	BSET <sup>5</sup> <sub>6</sub>	BMC <sup>3</sup> <sub>REL</sub>	INC <sup>5</sup> <sub>DIR</sub>	INCA <sup>3</sup> <sub>INH</sub>	INCX <sup>3</sup> <sub>INH</sub>	INC <sup>6</sup> <sub>IX1</sub>	INC <sup>5</sup> <sub>IX</sub>	RSP <sup>2</sup> <sub>INH</sub>	JMP <sup>2</sup> <sub>DIR</sub>	JMP <sup>3</sup> <sub>EXT</sub>	JMP <sup>4</sup> <sub>IX2</sub>	JMP <sup>5</sup> <sub>IX1</sub>	JMP <sup>4</sup> <sub>IX</sub>	1100			
D	BRCLR <sup>5</sup> <sub>6</sub>	BCLR <sup>5</sup> <sub>6</sub>	BMS <sup>3</sup> <sub>REL</sub>	TST <sup>4</sup> <sub>DIR</sub>	TSTA <sup>3</sup> <sub>INH</sub>	TSTX <sup>3</sup> <sub>INH</sub>	TST <sup>6</sup> <sub>IX1</sub>	TST <sup>4</sup> <sub>IX</sub>	NOP <sup>2</sup> <sub>INH</sub>	BSR <sup>6</sup> <sub>REL</sub>	JSR <sup>5</sup> <sub>DIR</sub>	JSR <sup>6</sup> <sub>EXT</sub>	JSR <sup>7</sup> <sub>IX2</sub>	JSR <sup>6</sup> <sub>IX1</sub>	JSR <sup>5</sup> <sub>IX</sub>	1101		
E	BRSET <sup>5</sup> <sub>7</sub>	BSET <sup>5</sup> <sub>7</sub>	BIL <sup>3</sup> <sub>REL</sub>						STOP <sup>2</sup> <sub>INH</sub>	LDX <sup>2</sup> <sub>IMM</sub>	LDX <sup>3</sup> <sub>DIR</sub>	LDX <sup>4</sup> <sub>EXT</sub>	LDX <sup>5</sup> <sub>IX2</sub>	LDX <sup>4</sup> <sub>IX1</sub>	LDX <sup>3</sup> <sub>IX</sub>	1110		
F	BRCLR <sup>5</sup> <sub>7</sub>	BCLR <sup>5</sup> <sub>7</sub>	BIH <sup>3</sup> <sub>REL</sub>	CLR <sup>5</sup> <sub>DIR</sub>	CLRA <sup>3</sup> <sub>INH</sub>	CLR <sup>3</sup> <sub>INH</sub>	CLR <sup>6</sup> <sub>IX1</sub>	CLR <sup>5</sup> <sub>IX</sub>	WAIT <sup>2</sup> <sub>INH</sub>	TXA <sup>2</sup> <sub>INH</sub>		STX <sup>2</sup> <sub>DIR</sub>	STX <sup>3</sup> <sub>EXT</sub>	STX <sup>4</sup> <sub>IX2</sub>	STX <sup>5</sup> <sub>IX1</sub>	STX <sup>4</sup> <sub>IX</sub>	1111	

Table 11-13 is an opcode map of the M68HC05 instruction set.

ABBREVIATIONS FOR ADDRESSING MODES

INH	Inherent	REL	Relative
IMM	Immediate	IX	Indexed, No Offset
DIR	Direct	IX1	Indexed, 8-Bit Offset
EXT	Extended	IX2	Indexed, 16-Bit Offset

LEGEND

F	High Byte of Opcode in Hexadecimal
1111	High Byte of Opcode in Binary
SUB	Low Byte of Opcode in Hexadecimal
3 0	Low Byte of Opcode in Binary
1	Number of Cycles
2	Opcode Mnemonic
1	Number of Bytes/Addressing Mode

## SECTION 12 ELECTRICAL SPECIFICATIONS

This section contains parametric and timing information.

### 12.1 Maximum Ratings

The MCU contains circuitry that protects the inputs against damage from high static voltages; however, do not apply voltages higher than those shown in Table 12-1. Keep  $V_{IN}$  and  $V_{OUT}$  within the range  $V_{SS} \leq (V_{IN} \text{ or } V_{OUT}) \leq V_{DD}$ . Connect unused inputs to the appropriate logic level, either  $V_{SS}$  or  $V_{DD}$ .

**Table 12-1. Maximum Ratings**

Rating	Symbol	Value	Unit
Supply Voltage	$V_{DD}$	-0.3 to +7.0	V
Input Voltage	$V_{IN}$	$V_{SS} - 0.3$ to $V_{DD} + 0.3$	V
Current Drain Per Pin (Excluding $V_{DD}$ and $V_{SS}$ )	I	25	mA
Operating Temperature Range MC68HC705K1P, DW, S MC68HC705K1CP, CDW, CS	$T_A$	0 to +70 -40 to +85	°C
Storage Temperature Range	$T_{STG}$	-65 to +150	°C

**NOTES:**

1. P = Plastic dual in-line package (PDIP)
2. DW = Small outline integrated circuit (SOIC)
3. S = Ceramic dual in-line package (Cerdip)
4. C = Extended temperature range

### 12.2 Thermal Characteristics

**Table 12-2. Thermal Resistance**

Characteristic	Symbol	Value	Unit
Maximum Junction Temperature	$T_J$	150	°C
Thermal Resistance	$\theta_{JA}$	60	°C/W
Plastic DIP			
Plastic SOIC			

### 12.3 Power Considerations

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

$$T_J = T_A + (P_D \times \theta_{JA}) \tag{1}$$

where:

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

$\theta_{JA}$  = 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  $P_{I/O} \ll P_{INT}$  and can be neglected.

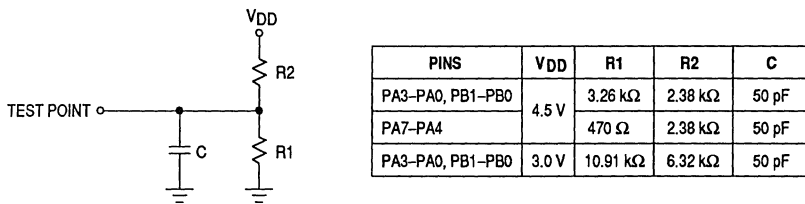
The following is an approximate relationship between  $P_D$  and  $T_J$  (neglecting  $P_{I/O}$ ):

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

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

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

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



**Figure 12-1. Equivalent Test Load**



## 12.4 DC Electrical Characteristics ( $V_{DD} = 5.0 \text{ Vdc}$ )

**Table 12-3. DC Electrical Characteristics ( $V_{DD} = 5.0 \text{ Vdc}$ )**

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage $I_{LOAD} = 10.0 \mu\text{A}$ $I_{LOAD} = -10.0 \mu\text{A}$	$V_{OL}$ $V_{OH}$	— $V_{DD} - 0.1$	— —	0.1 —	V
Output High Voltage ( $I_{LOAD} = -0.8 \text{ mA}$ ) PA7–PA0, PB1/OSC3, PB0	$V_{OH}$	$V_{DD} - 0.8$	—	—	V
Output Low Voltage ( $I_{LOAD} = 1.6 \text{ mA}$ ) PA3–PA0, PB1/OSC3, PB0 Output Low Voltage ( $I_{LOAD} = 8.0 \text{ mA}$ ) PA7–PA4	$V_{OL}$	— —	— —	0.4 0.4	V
Input High Voltage PA7–PA0, PB1/OSC3, PB0, $\overline{IRQ}$ , $\overline{RESET}$ , OSC1	$V_{IH}$	$0.7 \times V_{DD}$	—	$V_{DD}$	V
Input Low Voltage PA7–PA0, PB1–PB0, $\overline{IRQ}$ , $\overline{RESET}$ , OSC1	$V_{IL}$	$V_{SS}$	—	$0.2 \times V_{DD}$	V
Supply Current (NOTES 2–5) Run Wait Stop 25 °C 0 to +70 °C (Standard) –40 to +85 °C (Extended)	$I_{DD}$	— — — —	2.6 0.9 200 700 1000	— — — — —	mA mA nA nA nA
I/O Ports High-Z Leakage Current PA7–PA0, PB1/OSC3, PB0 (Pulldowns off)	$I_{OZ}$	—	—	$\pm 10$	$\mu\text{A}$
Input Pulldown Current PA7–PA0, PB1/OSC3, PB0 (Pulldowns on)	$I_{IL}$	50	100	200	$\mu\text{A}$
Input Current $\overline{RESET}$ , $\overline{IRQ}/V_{PP}$ , OSC1	$I_{IN}$	—	—	$\pm 1$	$\mu\text{A}$
Capacitance Ports (as input or output) $\overline{RESET}$ , $\overline{IRQ}/V_{PP}$	$C_{OUT}$ $C_{IN}$	— —	— —	12 8	pF
Low Voltage Reset Threshold (NOTE 6)	$V_{LVR}$	2.8	3.5	4.2	V
Crystal/Ceramic Resonator Oscillator Mode Internal Resistor (OSC1 to OSC2)	$R_{OSC}$	1.0	2.0	3.0	$\text{M}\Omega$
Programming Voltage (NOTE 7)	$V_{PP}$	—	16.5	—	V
Programming Current	$I_{PP}$	—	5	10	mA
Programming Time per Byte	$t_{EPGM}$	—	3	—	ms

### NOTES:

1. Typical values at midpoint of voltage range, 25 °C only.
2. Run (operating)  $I_{DD}$  and Wait  $I_{DD}$  measured using external square wave clock source ( $f_{OSC} = 4.2 \text{ MHz}$ ) with all inputs 0.2 V from rail; no dc loads; less than 50 pF on all outputs;  $C_L = 20 \text{ pF}$  on OSC2.
3. Wait  $I_{DD}$  and Stop  $I_{DD}$ : All ports configured as inputs;  $V_{IL} = 0.2 \text{ V}$ ,  $V_{IH} = V_{DD} - 0.2 \text{ V}$ .
4. Stop  $I_{DD}$  measured with  $OSC1 = V_{DD}$ .
5. OSC2 capacitance linearly affects Wait  $I_{DD}$ .
6. All MCUs guaranteed to operate at  $V_{DD} = 5 \text{ V} \pm 10\%$ . Each MCU guaranteed to operate at its  $V_{LVR}$ .
7. Programming voltage measured at  $\overline{IRQ}/V_{PP}$  pin.

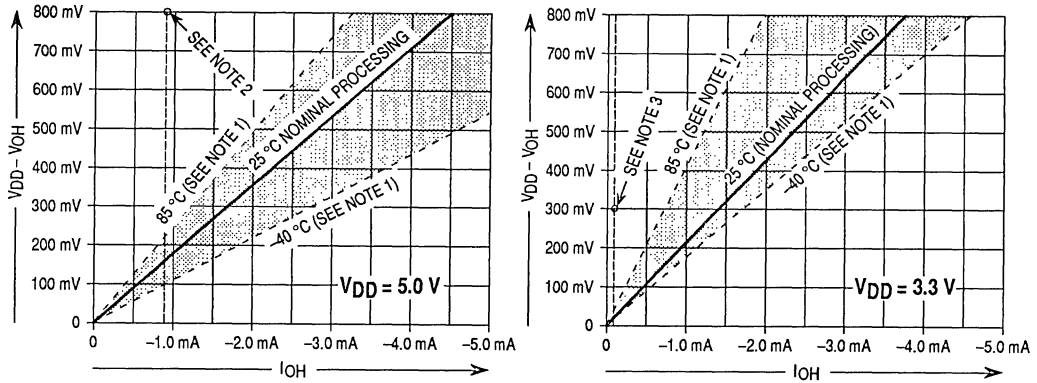
## 12.5 DC Electrical Characteristics (V<sub>DD</sub> = 3.3 Vdc)

**Table 12-4. DC Electrical Characteristics (V<sub>DD</sub> = 3.3 Vdc)**

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage I <sub>LOAD</sub> = 10.0 μA I <sub>LOAD</sub> = -10.0 μA	V <sub>OL</sub> V <sub>OH</sub>	— V <sub>DD</sub> - 0.1	— —	0.1 —	V
Output High Voltage (I <sub>LOAD</sub> = -0.4 mA) PA7-PA0, PB1/OSC3, PB0	V <sub>OH</sub>	V <sub>DD</sub> - 0.3	—	—	V
Output Low Voltage (I <sub>LOAD</sub> = 0.4 mA) PA3-PA0, PB1/OSC3, PB0 Output Low Voltage (I <sub>LOAD</sub> = 3.0 mA) PA7-PA4	V <sub>OL</sub>	— —	— —	0.3 0.3	V
Input High Voltage PA7-PA0, PB1/OSC3, PB0, $\overline{\text{IRQ}}$ , $\overline{\text{RESET}}$ , OSC1	V <sub>IH</sub>	0.7 × V <sub>DD</sub>	—	V <sub>DD</sub>	V
Input Low Voltage PA7-PA0, PB1/OSC3, PB0, $\overline{\text{IRQ}}$ , $\overline{\text{RESET}}$ , OSC1	V <sub>IL</sub>	V <sub>SS</sub>	—	0.2 × V <sub>DD</sub>	V
Supply Current (NOTES 2-4) Run Wait Stop 25 °C 0 to +70 °C (Standard) -40 to +85 °C (Extended)	I <sub>DD</sub>	— — — — —	0.7 300 50 500 1000	— — — — —	mA μA nA nA nA
I/O Ports High-Z Leakage Current PA7-PA0, PB1-PB0 (Pulldowns off)	I <sub>oz</sub>	—	—	±10	μA
Input Pulldown Current PA7-PA0, PB1/OSC3, PB0 (Pulldowns on)	I <sub>IL</sub>	20	30	100	μA
Input Current RESET, $\overline{\text{IRQ}}$ , OSC1	I <sub>IN</sub>	—	—	±1	μA
Capacitance Ports (as input or output) RESET, $\overline{\text{IRQ}}$	C <sub>OUT</sub> C <sub>IN</sub>	— —	— —	12 8	pF
Crystal/Ceramic Resonator Oscillator Mode Internal Resistor (OSC1 to OSC2)	R <sub>OSC</sub>	1.0	2.0	3.0	MΩ

### NOTES:

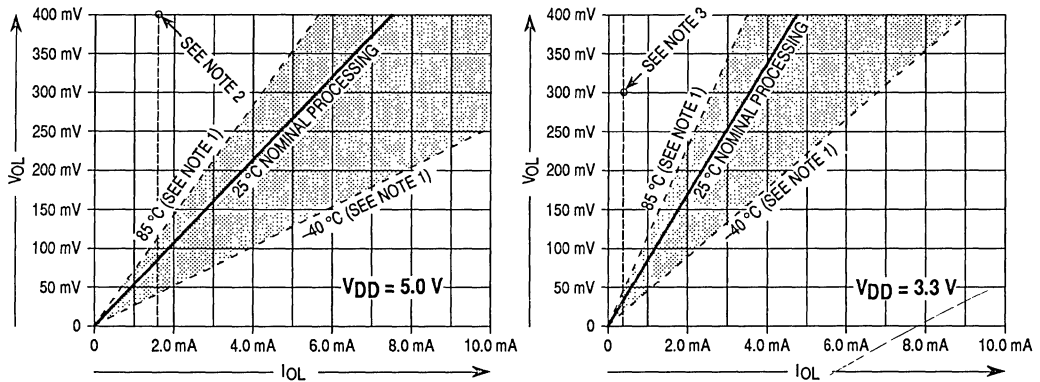
1. Typical values at midpoint of voltage range, 25 °C only.
2. Run (operating) I<sub>DD</sub> and Wait I<sub>DD</sub> measured using external square wave clock source (f<sub>osc</sub> = 2.1 MHz) with all inputs 0.2 V from rail; no dc loads; less than 50 pF on all outputs; C<sub>L</sub> = 20 pF on OSC2.
3. Wait I<sub>DD</sub> and Stop I<sub>DD</sub>: All ports configured as inputs; V<sub>IL</sub> = 0.2 V, V<sub>IH</sub> = V<sub>DD</sub> - 0.2 V.
4. Stop I<sub>DD</sub> measured with OSC1 = V<sub>DD</sub>.
5. OSC2 capacitance linearly affects Wait I<sub>DD</sub>.



NOTES:

1. Shaded area indicates variation in driver characteristics due to changes in temperature and for normal processing tolerances. Within the limited range of values shown, V vs I curves are approximately straight lines.
2. At  $V_{DD} = 5.0$  V, devices are specified and tested for  $(V_{DD} - V_{OH}) \leq 800$  mV @  $I_{OH} = -0.8$  mA.
3. At  $V_{DD} = 3.3$  V, devices are specified and tested for  $(V_{DD} - V_{OH}) \leq 300$  mV @  $I_{OH} = -0.2$  mA.

Figure 12-2. Typical High-Side Driver Characteristics



NOTES:

1. Shaded area indicates variation in driver characteristics due to changes in temperature and for normal processing tolerances. Within the limited range of values shown, V vs I curves are approximately straight lines.
2. At  $V_{DD} = 5.0$  V, devices are specified and tested for  $V_{OL} \leq 400$  mV @  $I_{OL} = 1.6$  mA.
3. At  $V_{DD} = 3.3$  V, devices are specified and tested for  $V_{OL} \leq 300$  mV @  $I_{OL} = 0.4$  mA.

Figure 12-3. Typical Low-Side Driver Characteristics

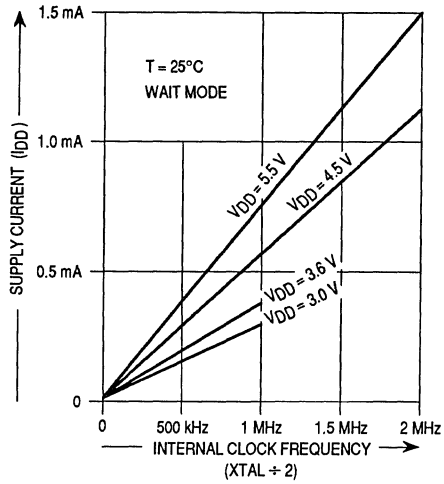
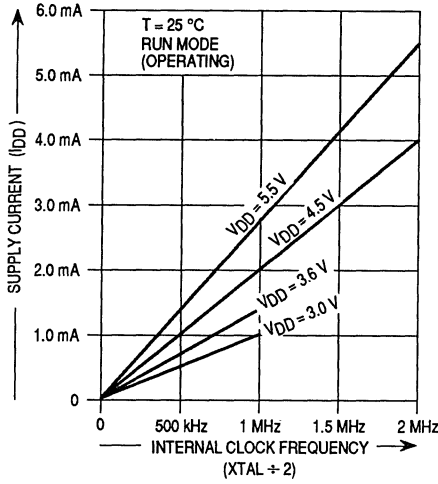
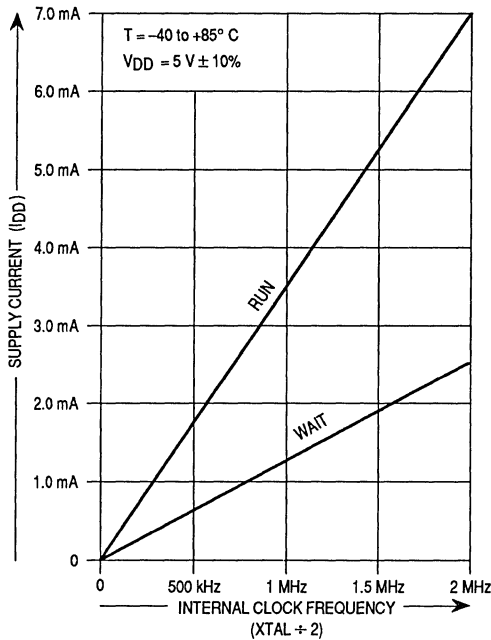
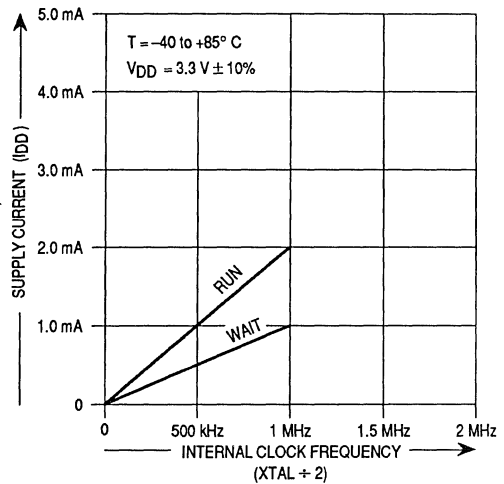


Figure 12-4. Typical Supply Current vs Clock Frequency



NOTE: Maximum STOP I<sub>DD</sub> = 100 μA when V<sub>DD</sub> = 5 V.



NOTE: Maximum STOP I<sub>DD</sub> = 50 μA when V<sub>DD</sub> = 3 V.

Figure 12-5. Maximum Supply Current vs Clock Frequency

## 12.6 Control Timing ( $V_{DD} = 5.0$ Vdc)

**Table 12-5. Control Timing ( $V_{DD} = 5.0$  Vdc)**

Characteristic	Symbol	Min	Max	Unit
Oscillator Frequency 3-Pin RC Oscillator Option 2-Pin RC Oscillator Option Crystal Option RC Clock Option	fosc	0.01 0.1 0.1 dc	1.0 2.0 4.0 4.0	MHz
Internal Operating Frequency 3-Pin RC Oscillator (fosc + 2) Crystal Oscillator (fosc + 2) 2-Pin RC Oscillator (fosc + 2) External Clock (fosc + 2)	fOP	— — — dc	0.5 2.0 1.0 2.0	MHz
Cycle Time ( $1 + f_{OP}$ )	tcyc	500	—	ns
RC Oscillator Stabilization Time	tRCON	—	1	ms
Crystal Oscillator Start-Up Time	toXON	—	100	ms
Stop Recovery Start-Up Time	tILCH	—	100	ms
RESET Pulse Width Low	tRL	1.5	—	tcyc
Timer Resolution (NOTE 2)	tRESL	4.0	—	tcyc
IRQ Pulse Width Low (Edge-Triggered)	tILIH	250	—	ns
IRQ Pulse Period	tILIL	(NOTE 3)	—	tcyc
PA0–PA3 Interrupt Pulse Width High (Edge Triggered)	tIHIL	250	—	ns
PA0–PA3 Interrupt Pulse Period	tIHIH	(NOTE 3)	—	tcyc
OSC1 Pulse Width	toH, toL	90	—	ns
Programming Time per Byte	tEPGM	4	—	ms
2-Pin RC Oscillator Frequency Combined Stability (NOTE 4) fosc = 2.0 MHz, $V_{DD} = 5.0$ Vdc $\pm 10\%$ , $T_A = -40$ °C to $+85$ °C fosc = 2.0 MHz, $V_{DD} = 5.0$ Vdc $\pm 10\%$ , $T_A = 0$ °C to $+40$ °C	$\Delta f_{osc}$	— —	$\pm 25$ $\pm 15$	%
3-Pin RC Oscillator Frequency Combined Stability (NOTE 4) fosc = 1.0 MHz, $V_{DD} = 5.0$ Vdc $\pm 10\%$ , $T_A = -40$ °C to $+85$ °C fosc = 1.0 MHz, $V_{DD} = 5.0$ Vdc $\pm 10\%$ , $T_A = 0$ °C to $+40$ °C	$\Delta f_{osc}$	— —	$\pm 15$ $\pm 10$	%

### NOTES:

- ( $V_{DD} = 5.0$  Vdc  $\pm 10\%$ ,  $V_{SS} = 0$  Vdc;  $T_A = T_L$  to  $T_H$ )
- The 2-bit timer prescaler is the limiting factor in determining timer resolution.
- The minimum period  $t_{ILIL}$  or  $t_{IHIH}$  should not be less than the number of cycles it takes to execute the interrupt service routine plus 19 tcyc.
- Including processing tolerances and variations in temperature and supply voltage. Excluding tolerances of external R and C.

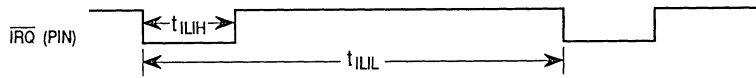
## 12.7 Control Timing ( $V_{DD} = 3.3 \text{ Vdc}$ )

**Table 12-6. Control Timing ( $V_{DD} = 3.3 \text{ Vdc}$ )**

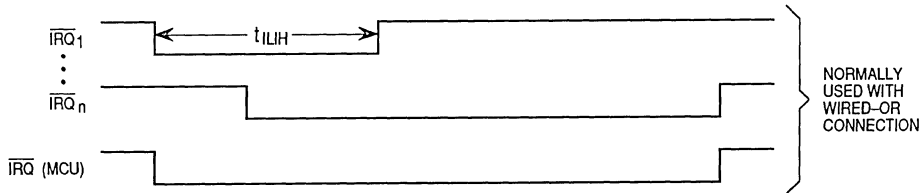
Characteristic	Symbol	Min	Max	Unit
Oscillator Frequency 3-Pin RC Oscillator Option 2-Pin RC Oscillator Option Crystal Option RC Clock Option	$f_{osc}$	0.01 0.1 0.1 dc	1.0 2.0 2.0 2.0	MHz
Internal Operating Frequency ( $f_{osc} + 2$ ) 3-Pin RC Oscillator Crystal Oscillator RC Oscillator External Clock	$f_{op}$	— — — dc	0.5 1.0 1.0 1.0	MHz
Cycle Time ( $1 + f_{op}$ ) 3-Pin RC Oscillator Crystal Oscillator RC Oscillator External Clock	$t_{cyc}$	2000 1000 1000 1000	— — — —	ns
RC Oscillator Stabilization Time	$t_{rCON}$	—	1	ms
Crystal Oscillator Start-Up Time	$t_{oXON}$	—	100	ms
Stop Recovery Start-Up Time	$t_{iLCH}$	—	100	ms
RESET Pulse Width Low	$t_{rL}$	1.5	—	$t_{cyc}$
Timer Resolution (NOTE 2)	$t_{rESL}$	4.0	—	$t_{cyc}$
IRQ Pulse Width Low (Edge-Triggered)	$t_{iLlH}$	250	—	ns
IRQ Pulse Period	$t_{iLlL}$	(NOTE 3)	—	$t_{cyc}$
PA0–PA3 Interrupt Pulse Width High (Edge-Triggered)	$t_{iHlL}$	250	—	ns
PA0–PA3 Interrupt Pulse Period	$t_{iHlH}$	(NOTE 3)	—	$t_{cyc}$
OSC1 Pulse Width	$t_{oH}, t_{oL}$	200	—	ns
Programming Time per Byte	$t_{EPGM}$	4	—	ms
2-Pin RC Oscillator Frequency Combined Stability (NOTE 4) $f_{osc} = 2.0 \text{ MHz}$ , $V_{DD} = 3.0 \text{ Vdc} \pm 10\%$ , $T_A = -40 \text{ }^\circ\text{C}$ to $+85 \text{ }^\circ\text{C}$ $f_{osc} = 2.0 \text{ MHz}$ , $V_{DD} = 3.0 \text{ Vdc} \pm 10\%$ , $T_A = 0 \text{ }^\circ\text{C}$ to $+40 \text{ }^\circ\text{C}$	$\Delta f_{osc}$	— —	$\pm 35$ $\pm 20$	%
3-Pin RC Oscillator Frequency Combined Stability (NOTE 4) $f_{osc} = 1.0 \text{ MHz}$ , $V_{DD} = 3.0 \text{ Vdc} \pm 10\%$ , $T_A = -40 \text{ }^\circ\text{C}$ to $+85 \text{ }^\circ\text{C}$ $f_{osc} = 1.0 \text{ MHz}$ , $V_{DD} = 3.0 \text{ Vdc} \pm 10\%$ , $T_A = 0 \text{ }^\circ\text{C}$ to $+40 \text{ }^\circ\text{C}$	$\Delta f_{osc}$	— —	$\pm 15$ $\pm 10$	%
EPROM Programming Time (EPROM or PEPROM)	$t_{EPGM}$	—	15	ms

### NOTES:

- ( $V_{DD} = 3.3 \text{ Vdc} \pm 10\%$ ,  $V_{SS} = 0 \text{ Vdc}$ ;  $T_A = T_L$  to  $T_H$ )
- The 2-bit timer prescaler is the limiting factor in determining timer resolution.
- The minimum period  $t_{iLlL}$  or  $t_{iHlH}$  should not be less than the number of cycles it takes to execute the interrupt service routine plus 19  $t_{cyc}$ .
- Including processing tolerances and variations in temperature and supply voltage. Excluding tolerances of external R and C.

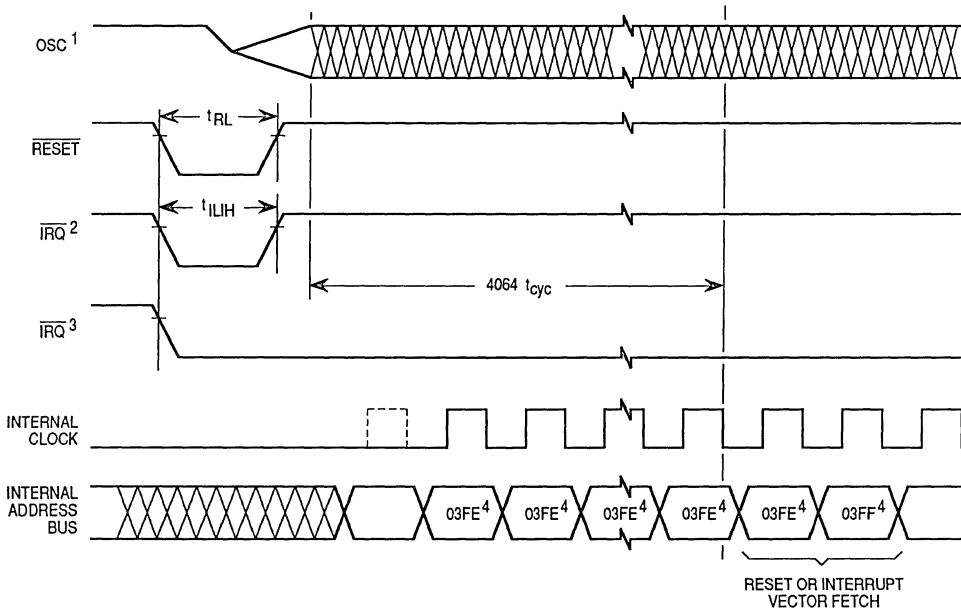


Edge-Sensitive Trigger — The minimum  $t_{ILIH}$  is either 125 ns ( $V_{DD} = 5$  V) or 250 ns ( $V_{DD} = 3$  V). The period  $t_{ILIL}$  should not be less than the number of  $t_{cyc}$  cycles it takes to execute the interrupt service routine plus 19  $t_{cyc}$  cycles.



Edge and Level-Sensitive Trigger — If  $\overline{IRQ}$  remains low after interrupt is serviced, the next interrupt is recognized.

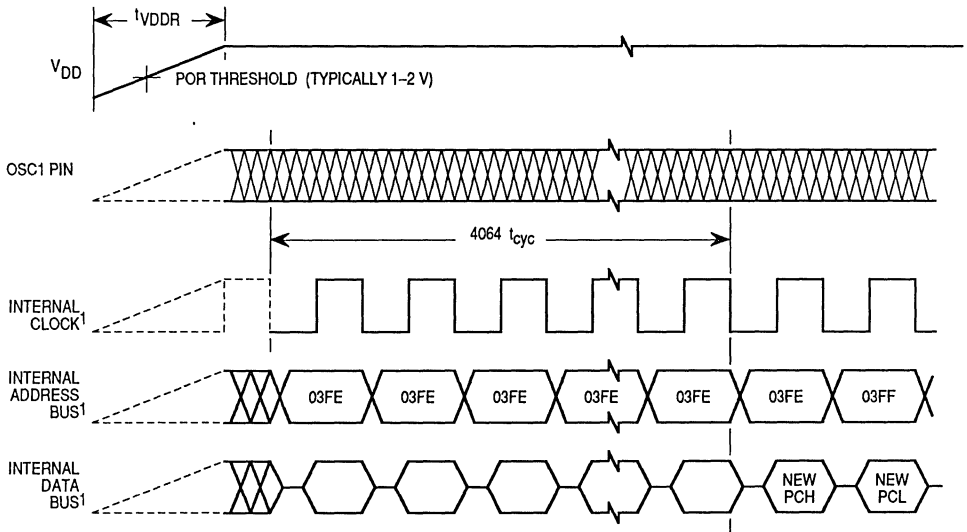
**Figure 12-6. External Interrupt Timing**



**NOTES:**

1. Represents the internal clocking of OSC1 pin
2.  $\overline{IRQ}$  pin edge-sensitive option
3.  $\overline{IRQ}$  pin level- and edge-sensitive option
4. Reset vector shown for timing example

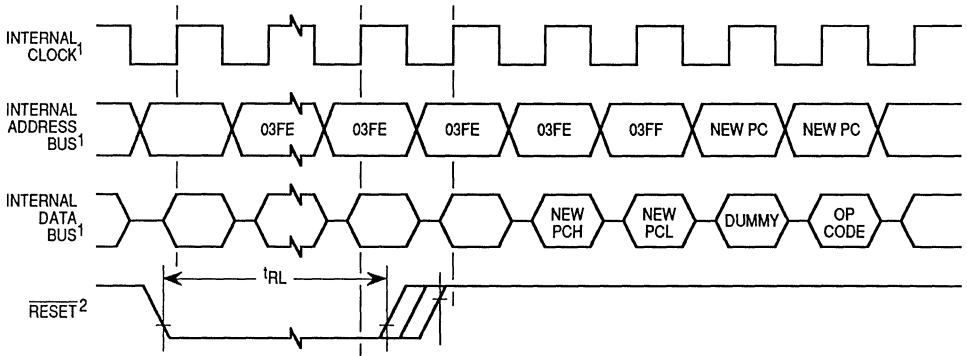
**Figure 12-7. STOP Recovery Timing**



NOTES:

1. Internal clock, internal address bus, and internal data bus are not available externally.

**Figure 12-8. Power-On Reset Timing**



NOTES:

1. Internal clock, internal address bus, and internal data bus signals are not available externally.
2. Next rising edge of internal clock after rising edge of RESET initiates reset sequence.

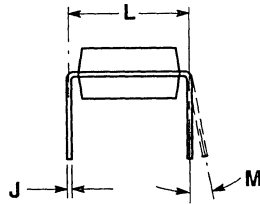
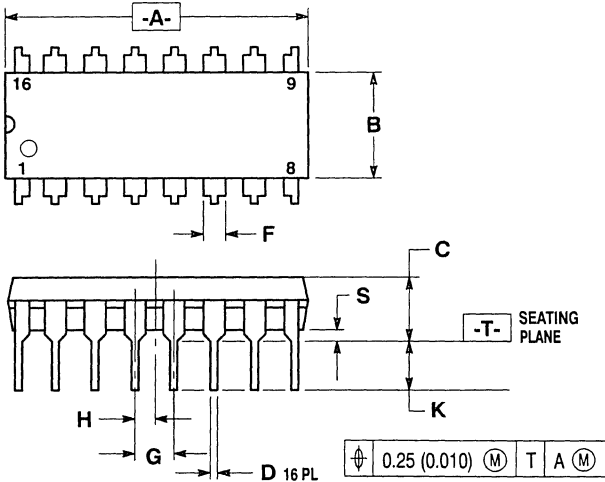
**Figure 12-9. External Reset Timing**



## SECTION 13 MECHANICAL SPECIFICATIONS

This section gives the dimensions of the plastic dual in-line package (PDIP), small outline integrated circuit (SOIC) package, and the ceramic DIP (Cerdip) package.

### 13.1 Plastic Dual In-Line Package (PDIP)



**NOTES:**

1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: INCH.
3. DIMENSION "L" TO CENTER OF LEADS WHEN FORMED PARALLEL.
4. DIMENSION "B" DOES NOT INCLUDE MOLD FLASH.
5. ROUNDED CORNERS OPTIONAL.

DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	18.80	19.55	0.740	0.770
B	6.35	6.85	0.250	0.270
C	3.69	4.44	0.145	0.175
D	0.39	0.53	0.015	0.021
F	1.02	1.77	0.040	0.070
G	2.54 BSC		0.100 BSC	
H	1.27 BSC		0.050 BSC	
J	0.21	0.38	0.008	0.015
K	2.80	3.30	0.110	0.130
L	7.50	7.74	0.295	0.305
M	0°	10°	0°	10°
S	0.51	1.01	0.020	0.040

CASE 648-08

**STYLE 1:**

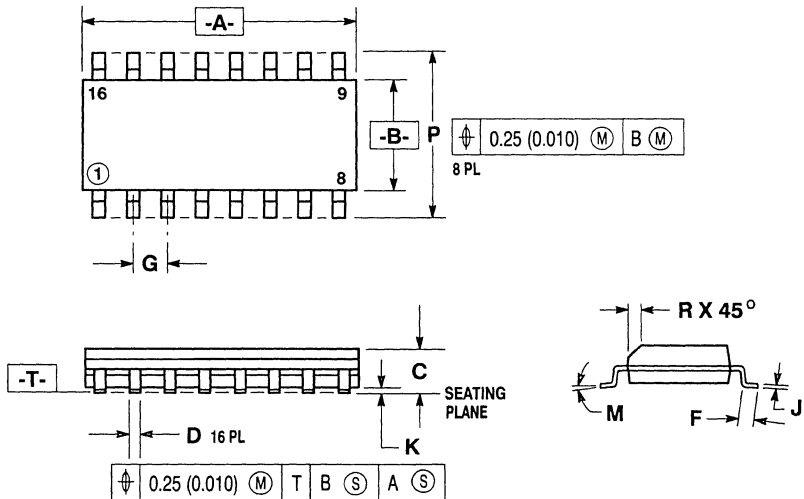
- PIN 1. CATHODE
2. CATHODE
3. CATHODE
4. CATHODE
5. CATHODE
6. CATHODE
7. CATHODE
8. CATHODE
9. ANODE
10. ANODE
11. ANODE
12. ANODE
13. ANODE
14. ANODE
15. ANODE
16. ANODE

**STYLE 2:**

- PIN 1. COMMON DRAIN
2. COMMON DRAIN
3. COMMON DRAIN
4. COMMON DRAIN
5. COMMON DRAIN
6. COMMON DRAIN
7. COMMON DRAIN
8. COMMON DRAIN
9. GATE
10. SOURCE
11. GATE
12. SOURCE
13. GATE
14. SOURCE
15. GATE
16. SOURCE

**Figure 13-1. MC68HC705K1P (Case 648-08)**

### 13.2 Small Outline Integrated Circuit (SOIC)



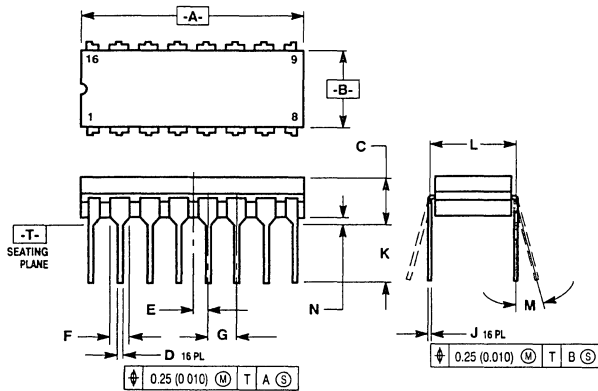
- NOTES:
1. DIMENSIONS A AND B ARE DATUMS AND T IS A DATUM SURFACE.
  2. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
  3. CONTROLLING DIMENSION: MILLIMETER.
  4. DIMENSION A AND B DO NOT INCLUDE MOLD PROTRUSION.
  5. MAXIMUM MOLD PROTRUSION 0.15 (0.006) PER SIDE.

DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	10.15	10.45	0.400	0.411
B	7.40	7.60	0.292	0.299
C	2.35	2.65	0.093	0.104
D	0.35	0.49	0.014	0.019
F	0.50	0.90	0.020	0.035
G	1.27 BSC		0.050 BSC	
J	0.25	0.32	0.010	0.012
K	0.10	0.25	0.004	0.009
M	0°	7°	0°	7°
P	10.05	10.55	0.395	0.415
R	0.25	0.75	0.010	0.029

751G-01

Figure 13-2. MC68HC705K1DW (Case 751G-01)

### 13.3 Ceramic DIP (Cerdip)



STYLE 1:

- PIN 1. CATHODE
- 2. CATHODE
- 3. CATHODE
- 4. CATHODE
- 5. CATHODE
- 6. CATHODE
- 7. CATHODE
- 8. CATHODE
- 9. ANODE
- 10. ANODE
- 11. ANODE
- 12. ANODE
- 13. ANODE
- 14. ANODE
- 15. ANODE
- 16. ANODE

NOTES

1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982
2. CONTROLLING DIMENSION, INCH.
3. DIMENSION L TO CENTER OF LEAD WHEN FORMED PARALLEL.
4. DIM F MAY NARROW TO 0.76 (0.030) WHERE THE LEAD ENTERS THE CERAMIC BODY.

DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	19.05	19.55	0.750	0.770
B	6.10	7.36	0.240	0.290
C	—	4.19	—	0.165
D	0.39	0.53	0.015	0.021
E	1.27 BSC	—	0.050 BSC	—
F	1.40	1.77	0.055	0.070
G	2.54 BSC	—	0.100 BSC	—
J	0.23	0.27	0.009	0.011
K	—	5.08	—	0.200
L	7.62 BSC	—	0.300 BSC	—
M	0°	15°	0°	15°
N	0.39	0.88	0.015	0.035

Figure 13-3. MC68HC705K1S (Case 620-09)





**Literature Distribution Centers:**

USA: Motorola Literature Distribution; P.O. Box 20912; Phoenix, Arizona 85036.

EUROPE: Motorola Ltd.; European Literature Centre; 88 Tanners Drive, Blakelands, Milton Keynes, MK14 5BP, England.

JAPAN: Nippon Motorola Ltd.; 4-32-1, Nishi-Gotanda, Shinagawa-ku, Tokyo 141 Japan.

ASIA-PACIFIC: Motorola Semiconductors H.K. Ltd.; Silicon Harbour Center, No. 2 Dai King Street, Tai Po Industrial Estate,  
Tai Po, N.T., Hong Kong.



**MOTOROLA**

