

# HCO5

**MC68HC05P6**

TECHNICAL  
DATA




**MOTOROLA**



# MC68HC05P6

## HCMOS MICROCONTROLLER UNIT

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters can and do vary in different applications. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Motorola does not 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
-----------	-------	------

## SECTION 1 GENERAL DESCRIPTION

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 and OSC2.....	1-5
1.4.2.1	Crystal.....	1-5
1.4.2.2	Ceramic Resonator.....	1-6
1.4.2.3	RC Oscillator.....	1-7
1.4.2.4	External Clock .....	1-8
1.4.3	RESET .....	1-8
1.4.4	IRQ.....	1-8
1.4.5	PA7–PA0.....	1-9
1.4.6	PB7/SCK .....	1-9
1.4.7	PB6/SDI.....	1-9
1.4.8	PB5/SDO.....	1-9
1.4.9	PC7/V <sub>RH</sub> .....	1-9
1.4.10	PC6/AN0.....	1-9
1.4.11	PC5/AN1.....	1-9
1.4.12	PC4/AN2.....	1-10
1.4.13	PC3/AN3.....	1-10
1.4.14	PC2–PC0 .....	1-10
1.4.15	PD7/TCAP.....	1-10
1.4.16	PD5.....	1-10
1.4.17	TCMP .....	1-10

## SECTION 2 MEMORY

2.1	Memory Map.....	2-1
2.2	Input/Output Section.....	2-1
2.3	RAM.....	2-1
2.4	ROM.....	2-4
2.5	Self-Check ROM .....	2-4

# TABLE OF CONTENTS (Continued)

Paragraph	Title	Page
<b>SECTION 3 CENTRAL PROCESSOR UNIT (CPU)</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
3.1.5	Condition Code Register.....	3-4
3.1.5.1	Half-Carry Flag (H).....	3-5
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
<b>SECTION 4 INTERRUPTS</b>		
4.1	Interrupt Sources.....	4-1
4.1.1	Software Interrupt.....	4-1
4.1.2	External Interrupt.....	4-1
4.1.3	Timer Interrupts.....	4-3
4.1.3.1	Input Capture Interrupt.....	4-3
4.1.3.2	Output Compare Interrupt.....	4-3
4.1.3.3	Timer Overflow Interrupt.....	4-3
4.2	Interrupt Processing.....	4-4
<b>SECTION 5 RESETS</b>		
5.1	Reset Sources.....	5-1
5.1.1	Power-On Reset.....	5-1
5.1.2	External Reset.....	5-1
5.1.3	Computer Operating Properly (COP) Watchdog Reset.....	5-2
5.2	Reset States.....	5-3
5.2.1	CPU.....	5-3
5.2.2	I/O Port Registers.....	5-3
5.2.3	Capture/Compare Timer.....	5-3
5.2.4	Serial I/O Port (SIOP).....	5-4
5.2.5	COP Watchdog.....	5-4
5.2.6	Analog-to-Digital Converter (ADC).....	5-4

# TABLE OF CONTENTS (Continued)

Paragraph	Title	Page
<b>SECTION 6 LOW POWER MODES</b>		
6.1	Stop Mode.....	6-1
6.2	Wait Mode .....	6-2
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
7.3	Port B.....	7-4
7.3.1	Port B Data Register (PORTB).....	7-4
7.3.2	Data Direction Register B (DDRB).....	7-5
7.4	Port C.....	7-7
7.4.1	Port C Data Register (PORTC).....	7-7
7.4.2	Data Direction Register C (DDRC).....	7-8
7.5	Port D.....	7-10
7.5.1	Port D Data Register (PORTD).....	7-10
7.5.2	Data Direction Register D (DDRD).....	7-11
<b>SECTION 8 CAPTURE/COMPARE TIMER</b>		
8.1	Timer Operation.....	8-2
8.1.1	Input Capture.....	8-2
8.1.2	Output Compare.....	8-3
8.2	Timer I/O Registers.....	8-4
8.2.1	Timer Control Register (TCR).....	8-5
8.2.2	Timer Status Register (TSR).....	8-6
8.2.3	Timer Registers (TRH and TRL).....	8-7
8.2.4	Alternate Timer Registers (ATRH and ATRL).....	8-8
8.2.5	Input Capture Registers (ICRH and ICRL).....	8-9
8.2.6	Output Compare Registers (OCRH and OCRL).....	8-10

# TABLE OF CONTENTS (Continued)

Paragraph	Title	Page
<b>SECTION 9</b>		
<b>SERIAL I/O PORT (SIOP)</b>		
9.1	SIOP Operation.....	9-2
9.1.1	SIOP Pin Functions.....	9-2
9.1.2	Serial Clock.....	9-3
9.1.3	Data Movement.....	9-4
9.2	SIOP I/O Registers.....	9-4
9.2.1	SIOP Control Register (SCR).....	9-4
9.2.2	SIOP Status Register (SSR).....	9-5
9.2.3	SIOP Data Register (SDR).....	9-6
<b>SECTION 10</b>		
<b>ANALOG-TO-DIGITAL CONVERTER (ADC)</b>		
10.1	ADC Operation.....	10-1
10.1.1	Pin Functions.....	10-2
10.1.1.1	PC7/V <sub>RH</sub> .....	10-2
10.1.1.2	PC6/AN0, PC5/AN1, PC4/AN2, and PC3/AN3.....	10-2
10.1.2	Conversion Accuracy.....	10-2
10.1.3	Conversion Time.....	10-2
10.1.4	Internal RC Oscillator.....	10-3
10.2	ADC I/O Registers.....	10-3
10.2.1	ADC Status and Control Register (ADSCR).....	10-3
10.2.2	ADC Data Register (ADDR).....	10-6
<b>SECTION 11</b>		
<b>SELF-CHECK ROM</b>		
11.1	Self-Check Tests.....	11-1
11.2	Self-Check Results.....	11-1
11.3	Self-Check Circuit.....	11-2
<b>SECTION 12</b>		
<b>INSTRUCTION SET</b>		
12.1	Addressing Modes.....	12-1
12.1.1	Inherent.....	12-2
12.1.2	Immediate.....	12-3
12.1.3	Direct.....	12-4
12.1.4	Extended.....	12-5



## TABLE OF CONTENTS (Continued)

Paragraph	Title	Page
12.1.5	Indexed, No Offset .....	12-6
12.1.6	Indexed, 8-Bit Offset .....	12-6
12.1.7	Indexed, 16-Bit Offset.....	12-6
12.1.8	Relative.....	12-8
12.2	Instruction Types.....	12-9
12.2.1	Register/Memory Instructions .....	12-9
12.2.2	Read-Modify-Write Instructions .....	12-10
12.2.3	Jump/Branch Instructions.....	12-10
12.2.4	Bit Manipulation Instructions.....	12-12
12.2.5	Control Instructions.....	12-12
12.3	Instruction Set Summary.....	12-13
12.4	Opcode Map .....	12-18

### SECTION 13 ELECTRICAL SPECIFICATIONS

13.1	Maximum Ratings .....	13-1
13.2	Thermal Characteristics.....	13-1
13.3	Power Considerations .....	13-2
13.4	DC Electrical Characteristics ( $V_{DD} = 5.0 \text{ Vdc}$ ).....	13-3
13.5	DC Electrical Characteristics ( $V_{DD} = 3.3 \text{ Vdc}$ ).....	13-4
13.6	ADC Characteristics.....	13-7
13.7	Control Timing ( $V_{DD} = 5.0 \text{ Vdc}$ ).....	13-8
13.8	Control Timing ( $V_{DD} = 3.3 \text{ Vdc}$ ).....	13-10
13.9	SIOP Timing ( $V_{DD} = 5.0 \text{ Vdc}$ ).....	13-12
13.10	SIOP Timing ( $V_{DD} = 3.3 \text{ Vdc}$ ).....	13-13

### SECTION 14 MECHANICAL SPECIFICATIONS

14.1	PDIP .....	14-1
14.2	SOIC.....	14-2

# TABLE OF CONTENTS (Concluded)

Section	Title	Page
<b>SECTION 15</b>		
<b>ORDERING INFORMATION</b>		
15.1	MCU Ordering Forms.....	15-1
15.2	Application Program Media.....	15-2
15.2.1	Diskettes.....	15-2
15.2.2	EPROMs.....	15-3
15.3	ROM Program Verification.....	15-4
15.4	ROM Verification Units (RVUs).....	15-5
15.5	XC Status Letter.....	15-5

## LIST OF FIGURES

Figure	Title	Page
1-1	MC68HC05P6 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	RC Oscillator Connections .....	1-7
1-7	External Clock Connections .....	1-8
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 Logic .....	4-2
4-2	Interrupt Stacking Order .....	4-4
4-3	Interrupt Flowchart .....	4-6
5-1	Reset Sources .....	5-2
5-2	COP Register (COPR) .....	5-2
6-1	STOP/HALT/WAIT Flowchart .....	6-4
7-1	Port A Data Register (PORTA) .....	7-2
7-2	Data Direction Register A (DDRA) .....	7-2
7-3	Port A I/O Circuit .....	7-3
7-4	Port B Data Register (PORTB) .....	7-4
7-5	Data Direction Register B (DDRB) .....	7-5
7-6	Port B I/O Circuit .....	7-6

## LIST OF FIGURES (Continued)

Figure	Title	Page
7-7	Port C Data Register (PORTC) .....	7-7
7-8	Data Direction Register C (DDRC) .....	7-8
7-9	Port C I/O Circuit.....	7-9
7-10	Port D Data Register (PORTD) .....	7-10
7-11	Data Direction Register D (DDRD) .....	7-11
7-12	Port D I/O Circuit.....	7-12
8-1	Timer Block Diagram.....	8-1
8-2	Input Capture Operation.....	8-3
8-3	Output Compare Operation.....	8-4
8-4	Timer Control Register (TCR) .....	8-5
8-5	Timer Status Register (TSR).....	8-6
8-6	Timer Registers (TRH and TRL) .....	8-7
8-7	Timer Register Reads.....	8-8
8-8	Alternate Timer Registers (ATRH and ATRL).....	8-8
8-9	Alternate Timer Register Reads .....	8-9
8-10	Input Capture Register (ICRH/ICRL).....	8-9
8-11	Output Compare Registers (OCRH and OCRL) .....	8-10
9-1	SIOPI Block Diagram .....	9-1
9-2	SIOPI Data/Clock Timing.....	9-3
9-3	Master/Slave SIOPI Shift Register Operation.....	9-4
9-4	SIOPI Control Register (SCR) .....	9-5
9-5	SIOPI Status Register (SCR).....	9-5
9-6	SIOPI Data Register (SDR).....	9-6
10-1	ADC Block Diagram .....	10-1
10-2	ADC Status and Control Register (ADSCR).....	10-4
10-3	ADC Data Register (ADDR) .....	10-6
11-1	Self-Check Circuit.....	11-2
13-1	Test Load.....	13-2
13-2	I <sub>DD</sub> vs Internal Clock Frequency (T = 25 °C).....	13-5
13-3	I <sub>DD</sub> vs Internal Clock Frequency (T = -40 °C to +125 °C) .....	13-6

# LIST OF FIGURES (Concluded)

Figure	Title	Page
13-4	TCAP Timing.....	13-9
13-5	STOP Recovery Timing .....	13-9
13-6	External Interrupt Timing .....	13-10
13-7	Power-On Reset Timing.....	13-11
13-8	External Reset Timing.....	13-11
13-9	SLOP Timing.....	13-12
14-1	Case #710-02.....	14-1
14-2	Case #751F-02 .....	14-2



## LIST OF TABLES

Table	Title	Page
4-1	Reset/Interrupt Vector Addresses .....	4-5
7-1	Port A Pin Functions.....	7-4
7-2	Port B Pin Functions.....	7-6
7-3	Port C Pin Functions.....	7-9
7-4	Port D Pin Functions.....	7-12
10-1	ADC Input Channel Selection.....	10-5
11-1	Self-Check Circuit LED Codes.....	10-1
12-1	Inherent Addressing Instructions .....	12-2
12-2	Immediate Addressing Instructions .....	12-3
12-3	Direct Addressing Instructions.....	12-4
12-4	Extended Addressing Instructions.....	12-5
12-5	Indexed Addressing Instructions.....	12-7
12-6	Relative Addressing Instructions.....	12-8
12-7	Register/Memory Instructions .....	12-9
12-8	Read-Modify-Write Instructions .....	12-10
12-9	Jump and Branch Instructions .....	12-11
12-10	Bit Manipulation Instructions.....	12-12
12-11	Control Instructions.....	12-12
12-12	Instruction Set.....	12-14
13-1	Maximum Ratings .....	13-1
13-2	Thermal Resistance.....	13-1
13-3	DC Electrical Characteristics ( $V_{DD} = 5.0 \text{ Vdc}$ ).....	13-3
13-4	DC Electrical Characteristics ( $V_{DD} = 3.3 \text{ Vdc}$ ).....	13-4
13-5	A/D Converter Characteristics.....	13-7
13-6	Control Timing ( $V_{DD} = 5.0 \text{ Vdc}$ ).....	13-8
13-7	Control Timing ( $V_{DD} = 3.3 \text{ Vdc}$ ).....	13-10
13-8	SIOPI Timing ( $V_{DD} = 5.0 \text{ Vdc}$ ).....	13-12
13-9	SIOPI Timing ( $V_{DD} = 3.3 \text{ Vdc}$ ).....	13-13





## **SECTION 1 GENERAL DESCRIPTION**

The MC68HC05P6 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 MC68HC05P6 includes 176 bytes of user RAM and 4672 bytes of user ROM.

### **1.1 Features**

Features of the MC68HC05P6 MCU include the following:

- Popular M68HC05 Central Processor Unit
- Memory-Mapped Input/Output (I/O) Registers
- 4672 Bytes of User ROM Including 48 Bytes of Page Zero ROM and 16 User Vector Locations
- 176 Bytes of User RAM
- 20 I/O Port Pins and One Input-Only Port Pin
- On-Chip Oscillator with Crystal or Ceramic Resonator Connections or Resistor-Capacitor (RC) Connections
- 4-Input, 8-Bit Analog-to-Digital Converter (ADC)
- Synchronous Serial I/O Port (SIOP)
- 16-Bit Capture/Compare Timer
- Fully Static Operation with No Minimum Clock Speed
- Self-Check ROM
- Computer Operating Properly (COP) Watchdog
- Power-Saving Stop (or Halt), Wait, and Data-Retention Modes
- 8 × 8 Unsigned Multiply Instruction
- 28-Pin Plastic Dual In-Line Package (PDIP)
- 28-Pin Small Outline Integrated Circuit (SOIC)

## 1.2 Mask Options

The following MC68HC05P6 mask options are available:

- On-chip oscillator connections: crystal/ceramic resonator connections or resistor-capacitor (RC) network connections
- STOP instruction: enabled or disabled
- SIOP clock rate: oscillator frequency divided by 8, 16, 32, or 64
- SIOP data format: MSB-first or LSB-first
- External interrupt pin: edge-triggered or edge- and level-triggered
- COP watchdog: enabled or disabled

### 1.3 MCU Structure

Figure 1-1 shows the structure of the MC68HC05P6 MCU.

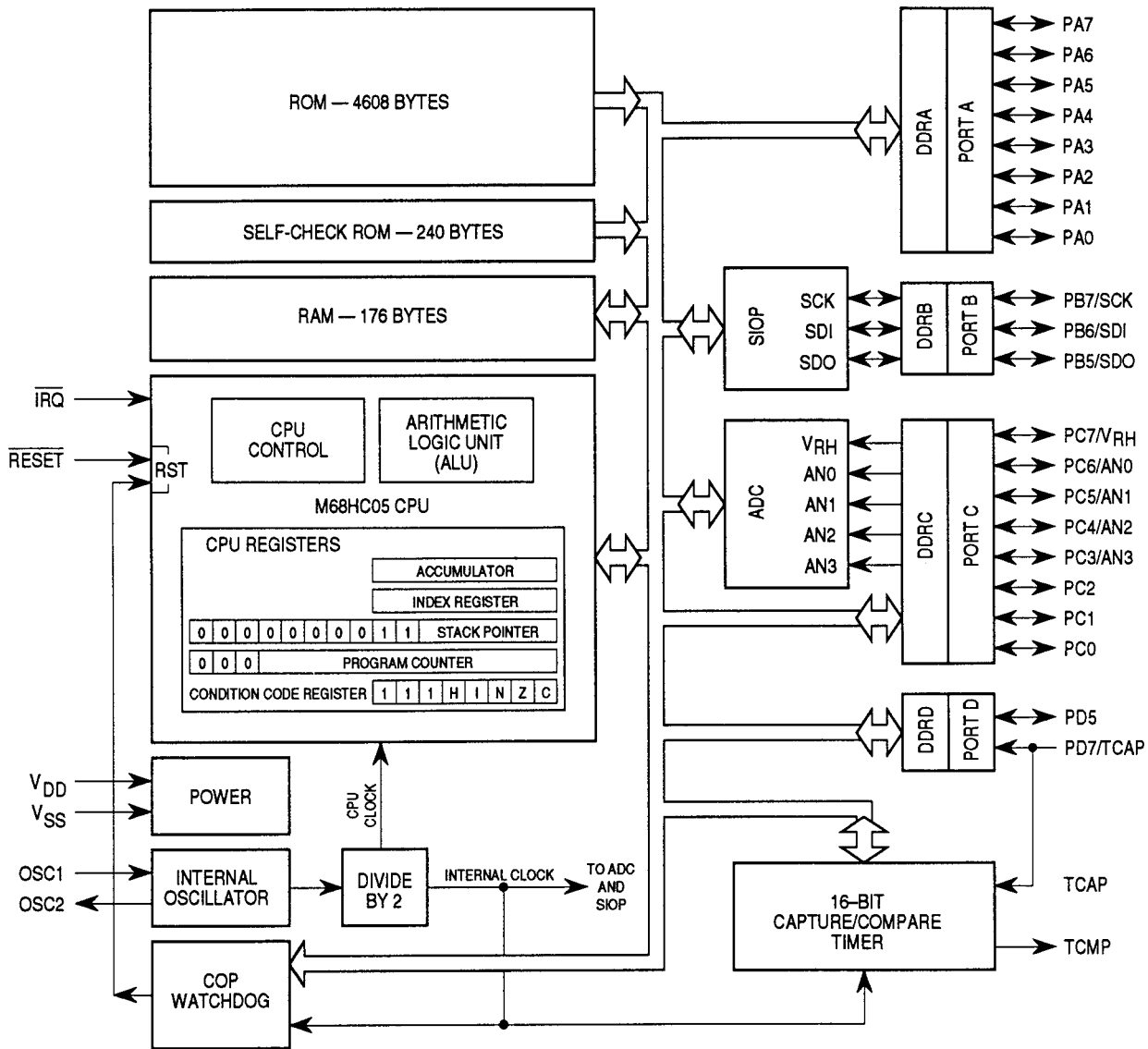
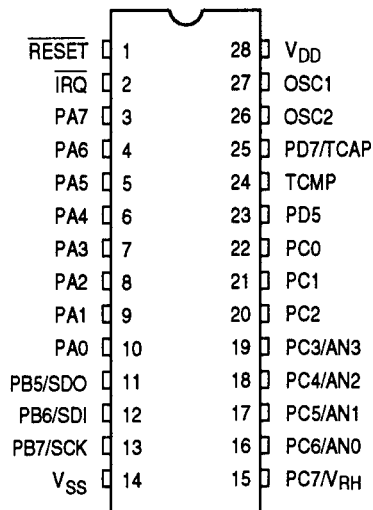


Figure 1-1. MC68HC05P6 Block Diagram

## 1.4 Pin Assignments

Figure 1-2 shows the MC68HC05P6 pin assignments.

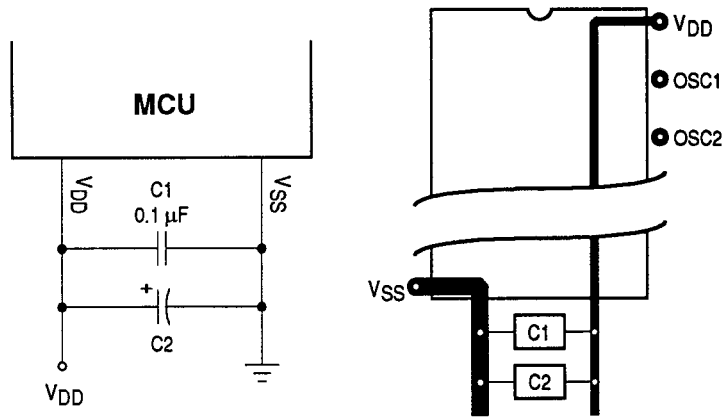


**Figure 1-2. Pin Assignments**

### 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. C2 is an optional bulk current bypass capacitor for use in applications that require the port pins to source high current levels.



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

### 1.4.2 OSC1 and OSC2

The OSC1 and OSC2 pins are the control connections for the on-chip oscillator. Depending on the mask option selected, the oscillator can be driven by any of the following:

- Crystal
- Ceramic resonator
- Resistor-capacitor network
- External clock signal

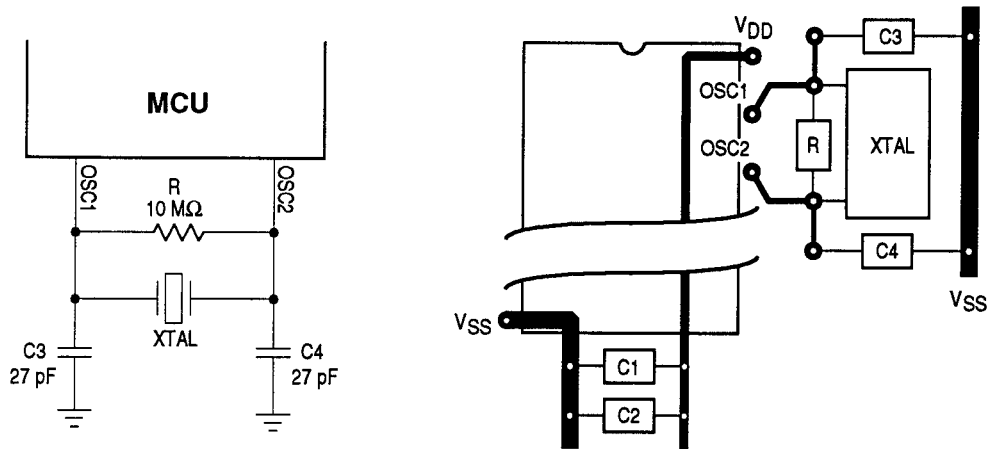
The frequency of the internal oscillator is  $f_{OSC}$ . The MCU divides the internal oscillator output by two to produce the internal clock. The frequency of the internal clock is  $f_{OP}$ .

#### 1.4.2.1 Crystal

With the crystal/ceramic resonator mask option, a crystal connected to the OSC1 and OSC2 pins can drive the on-chip oscillator. 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 reliable start-up and maximum stability. The load capacitance values used in the oscillator circuit design should include all stray layout capacitances. To minimize output distortion, mount the crystal and capacitors as close as possible to the pins.

## NOTE

Use an AT-cut crystal and not an AT-strip crystal. The MCU may overdrive an AT-strip crystal.



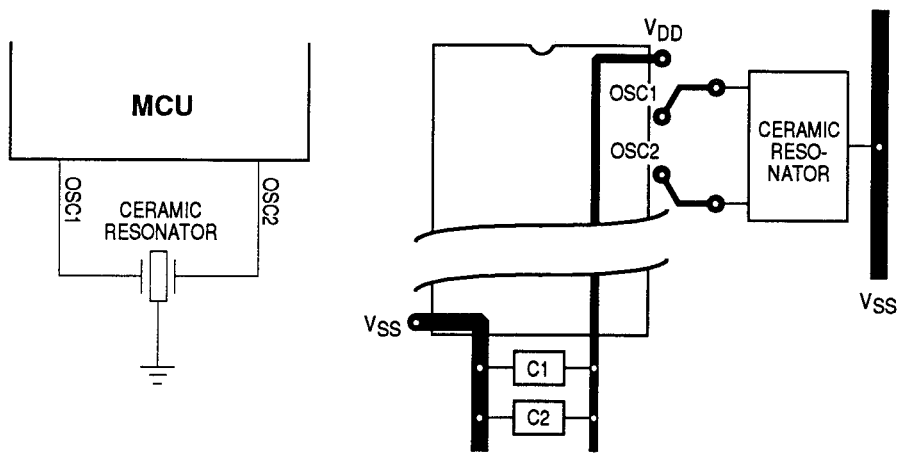
**Figure 1-4. Crystal Connections**

### 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 load capacitance values used in the oscillator circuit design should include all stray layout capacitances. To minimize output distortion, mount the resonator as close as possible to the pins.

## NOTE

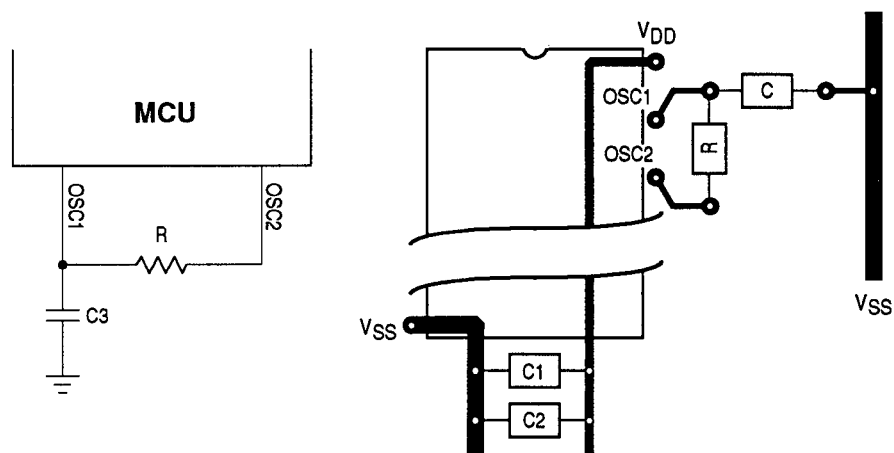
Because the frequency stability of ceramic resonators is not as high as that of crystal oscillators, using a ceramic resonator may degrade the performance of the analog-to-digital converter (ADC).



**Figure 1-5. Ceramic Resonator Connections**

### 1.4.2.3 RC Oscillator

For maximum cost reduction, the RC oscillator mask option allows the configuration shown in Figure 1-6 to drive the on-chip oscillator. 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. The optimum frequency for the RC oscillator configuration is 2 MHz. Mount the RC components as close as possible to the pins for start-up stabilization and to minimize output distortion.



**Figure 1-6. RC Oscillator Connections**

#### 1.4.2.4 External Clock

With the RC oscillator mask option, an external clock from another CMOS-compatible device can drive the OSC1 input. Leave the OSC2 pin unconnected, as Figure 1-7 shows.

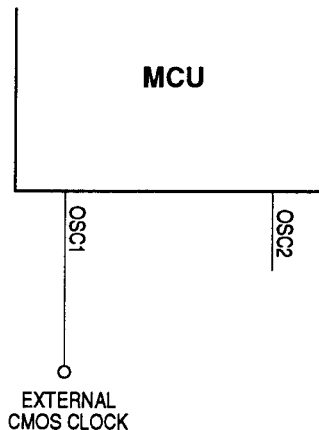


Figure 1-7. External Clock Connections

#### 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 $\overline{\text{IRQ}}$

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

- Applying asynchronous external interrupt signals (See **4.1.2 External Interrupt**.)
- Applying  $V_{TST}$  to put the MCU in self-check mode (See **SECTION 11 SELF-CHECK ROM**.)



#### **1.4.5 PA7–PA0**

PA7–PA0 are the pins of port A, a general-purpose bidirectional I/O port.

#### **1.4.6 PB7/SCK**

PB7/SCK is a general-purpose bidirectional port B I/O pin. When the serial I/O port (SIOP) is enabled, PB7/SCK is the serial clock input pin (slave mode) or the serial clock output pin (master mode).

#### **1.4.7 PB6/SDI**

PB6/SDI is a general-purpose bidirectional port B I/O pin. When the serial I/O port (SIOP) is enabled, PB6/SDI is the serial data input pin.

#### **1.4.8 PB5/SDO**

PB5/SDO is a general-purpose bidirectional port B I/O pin. When the serial I/O port (SIOP) is enabled, PB5/SDO is the serial data output pin.

#### **1.4.9 PC7/V<sub>RH</sub>**

PC7/V<sub>RH</sub> is a general-purpose bidirectional port C I/O pin. When the analog-to-digital converter (ADC) is enabled, PC7/V<sub>RH</sub> is the positive reference voltage pin for the ADC.

#### **1.4.10 PC6/AN0**

PC6/AN0 is a general-purpose bidirectional port C I/O pin. When the analog-to-digital converter (ADC) is enabled, PC6/AN0 is an analog input pin to the ADC.

#### **1.4.11 PC5/AN1**

PC5/AN1 is a general-purpose bidirectional port C I/O pin. When the analog-to-digital converter (ADC) is enabled, PC5/AN1 is an analog input pin to the ADC.

#### **1.4.12 PC4/AN2**

PC4/AN2 is a general-purpose bidirectional port C I/O pin. When the analog-to-digital converter (ADC) is enabled, PC4/AN2 is an analog input pin to the ADC.

#### **1.4.13 PC3/AN3**

PC3/AN3 is a general-purpose bidirectional port C I/O pin. When the analog-to-digital converter (ADC) is enabled, PC3/AN3 is an analog input pin to the ADC.

#### **1.4.14 PC2–PC0**

PC2–PC0 are general-purpose bidirectional port C I/O pins.

#### **1.4.15 PD7/TCAP**

PD7/TCAP serves as both a general-purpose input-only port D I/O pin and as the input capture line for the capture/compare timer.

#### **1.4.16 PD5**

PD5 is a general-purpose bidirectional port D I/O pin.

#### **1.4.17 TCMP**

TCMP is the output pin for the output compare function of the capture/compare timer.

## SECTION 2 MEMORY

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

### 2.1 Memory Map

The CPU can address 8 Kbytes of memory space. The ROM 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.

### 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. (See Figure 2-2.)

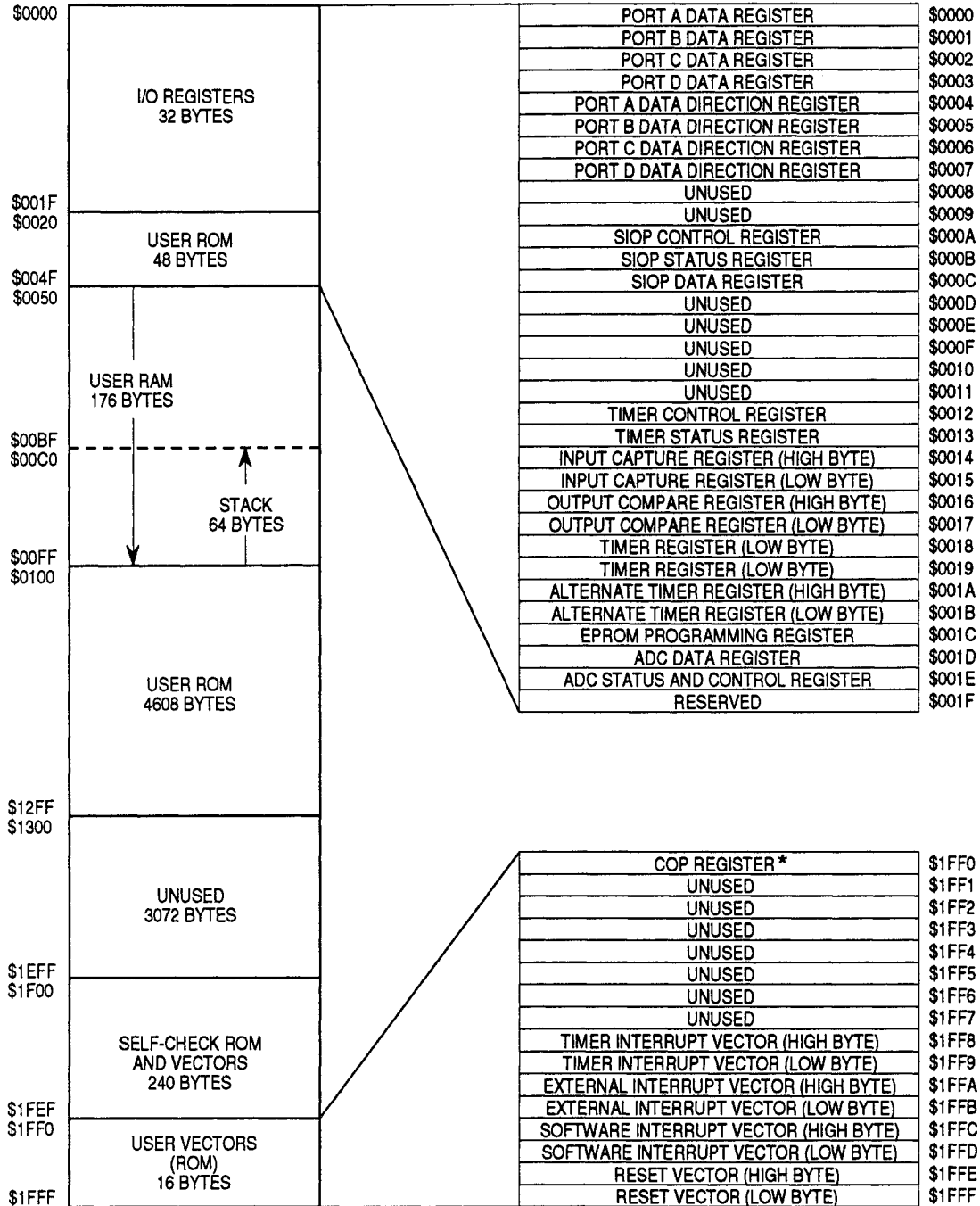
One I/O register shown in Figure 2-2 is located outside the 32-byte I/O section: the computer operating properly (COP) register is mapped at \$1FF0.

### 2.3 RAM

The 176 addresses from \$0050 to \$00FF are RAM locations. The CPU uses the top 64 RAM addresses, \$00C0–\$00FF, as the stack. Before processing an interrupt, the CPU uses five bytes of the stack to save the contents of the CPU registers. During a subroutine call, the CPU uses two stack 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 \$1FF0 clears the COP watchdog timer. Reading \$1FF0 returns user ROM data.

**Figure 2-1. Memory Map**

\$0000	Bit 7	6	5	4	3	2	1	Bit 0	PORTA
	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0	
\$0001	PB7	PB6	PB5	0	0	0	0	0	PORTB
\$0002	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0	PORTC
\$0003	PD7	0	PD5	1	0	0	0	0	PORTD
\$0004	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0	DDRA
\$0005	DDRB7	DDRB6	DDRB5	1	1	1	1	1	DDRB
\$0006	DDRC7	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0	DDRC
\$0007	0	0	DDR5	0	0	0	0	0	DDR
\$0008	—	—	—	—	—	—	—	—	UNUSED
\$0009	—	—	—	—	—	—	—	—	UNUSED
\$000A	0	SPE	0	MSTR	0	0	0	0	SCR
\$000B	SPIF	DCOL	0	0	0	0	0	0	SSR
\$000C	Bit 7	6	5	4	3	2	1	Bit 0	SDR
\$000D									RESERVED
\$000E	—	—	—	—	—	—	—	—	UNUSED
\$000F	—	—	—	—	—	—	—	—	UNUSED
\$0010	—	—	—	—	—	—	—	—	UNUSED
\$0011	—	—	—	—	—	—	—	—	UNUSED
\$0012	ICIE	OCIE	TOIE	0	0	0	IEDG	OLVL	TCR
\$0013	ICF	OCF	TOF	0	0	0	0	0	TSR
\$0014	Bit 15	14	13	12	11	10	9	Bit 8	ICRH
\$0015	Bit 7	6	5	4	3	2	1	Bit 0	ICRL
\$0016	Bit 15	14	13	12	11	10	9	Bit 8	OCRH
\$0017	Bit 7	6	5	4	3	2	1	Bit 0	OCRL
\$0018	Bit 15	14	13	12	11	10	9	Bit 8	TCRH
\$0019	Bit 7	6	5	4	3	2	1	Bit 0	TCRL
\$001A	Bit 15	14	13	12	11	10	9	Bit 8	ACRH
\$001B	Bit 7	6	5	4	3	2	1	Bit 0	ACRL
\$001C	—	—	—	—	—	—	—	—	UNUSED
\$001D	Bit 7	6	5	4	3	2	1	Bit 0	ADDR
\$001E	CC	ADRC	ADON	0	0	CH2	CH1	CH0	ADSCR
\$001F	—	—	—	—	—	—	—	—	RESERVED
\$1FF0								COPC	COPR

Figure 2-2. I/O Registers

## 2.4 ROM

The ROM is located in three areas of the memory map:

- Addresses \$0020–\$004F contain 48 bytes of page zero ROM.
- Addresses \$0100–\$12FF contain 4608 bytes of ROM.
- Addresses \$1FF0–\$1FFF contain 16 bytes of ROM reserved for vectors.

## 2.5 Self-Check ROM

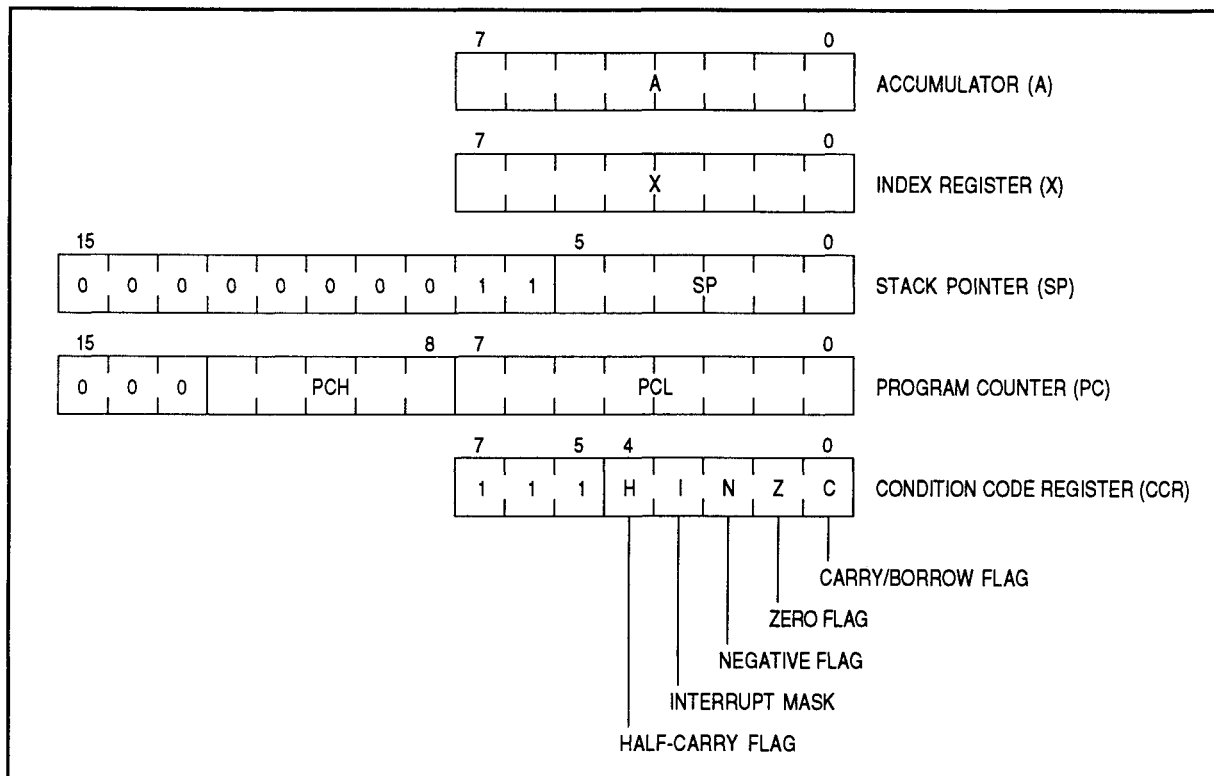
Addresses \$1F00–\$1FEF contain the self-check ROM. When activated, the self-check program performs a series of MCU functional tests. (See **SECTION 11 SELF-CHECK ROM.**)

## SECTION 3 CENTRAL PROCESSOR UNIT

This section describes the CPU registers.

### 3.1 CPU Registers

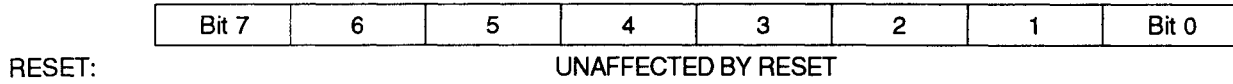
Figure 3-1 shows the five CPU registers. CPU registers are not part of the memory map.



**Figure 3-1. Programming Model**

### 3.1.1 Accumulator

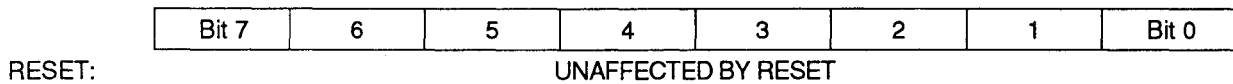
The accumulator is a general-purpose 8-bit register. The CPU uses the accumulator to hold operands and results of arithmetic and nonarithmetic operations.



**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 **12.1 Addressing Modes.**)



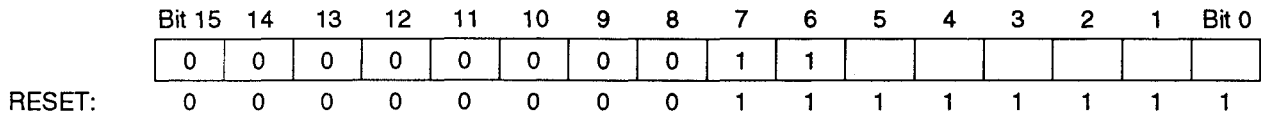
**Figure 3-3. Index Register**

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 preset 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.



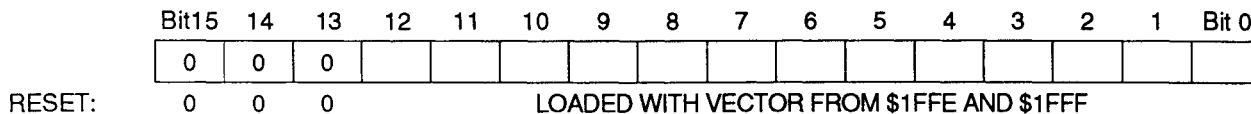
**Figure 3-4. Stack Pointer**

The ten most significant bits of the stack pointer are permanently fixed at 0000000011, so the stack pointer produces addresses from \$00C0 to \$00FF. If subroutines and interrupts use more than 64 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 three most significant bits of the program counter are ignored internally and appear as 000.

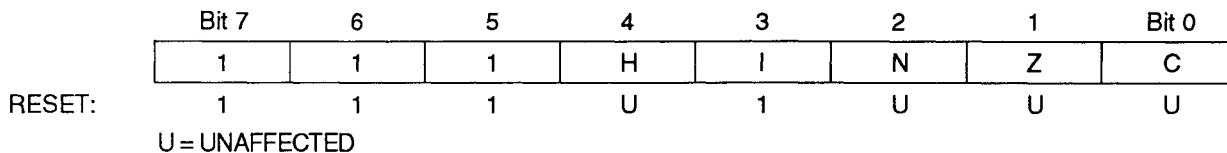
Normally, the address in the program counter automatically increments to the next sequential memory location every time an instruction or operand is fetched. Jump, branch, and interrupt operations load the program counter with an address other than that of the next sequential location.



**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 software 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.

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

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

### **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 binary arithmetic circuits decode instructions and 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 Sources

The following sources can generate interrupt requests:

- SWI instruction
- $\overline{\text{IRQ}}$  pin
- Capture/compare timer

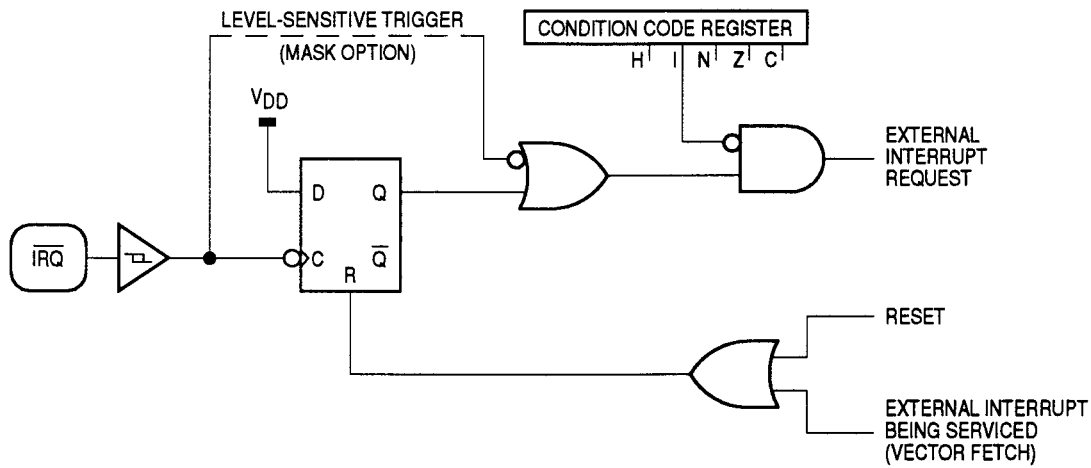
An interrupt temporarily stops normal program execution to process a particular event. An interrupt does not stop the execution of the instruction in progress, but takes effect when the current instruction completes its execution. Interrupt processing automatically saves the CPU registers on the stack and loads the program counter with a user-defined interrupt vector address.

#### 4.1.1 Software Interrupt

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

#### 4.1.2 External Interrupt

An interrupt signal on the  $\overline{\text{IRQ}}$  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. If the I bit is clear, 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.



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

Interrupt triggering sensitivity of the  $\overline{\text{IRQ}}$  pin is a mask option. The  $\overline{\text{IRQ}}$  pin can be negative edge-triggered or negative edge- and low-level-triggered. The low-level-sensitive triggering option allows the wired-OR use of multiple external interrupt sources. An external interrupt request is latched as long as any source is holding the  $\overline{\text{IRQ}}$  pin low.

### 4.1.3 Timer Interrupts

The capture/compare timer can generate the following interrupts:

- Input capture interrupt
- Output compare interrupt
- Timer overflow interrupt

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

#### 4.1.3.1 Input Capture Interrupt

An input capture interrupt request occurs if the input capture flag, ICF, becomes set while the input capture interrupt enable bit, ICIE, is also set. ICF is in the timer status register, and ICIE is in the timer control register. (See **SECTION 8 CAPTURE/COMPARE TIMER.**)

#### 4.1.3.2 Output Compare Interrupt

An output compare interrupt request occurs if the output compare flag, OCF, becomes set while the output compare interrupt enable bit, OCIE, is also set. OCF is in the timer status register, and OCIE is in the timer control register. (See **SECTION 8 CAPTURE/COMPARE TIMER.**)

#### 4.1.3.3 Timer Overflow Interrupt

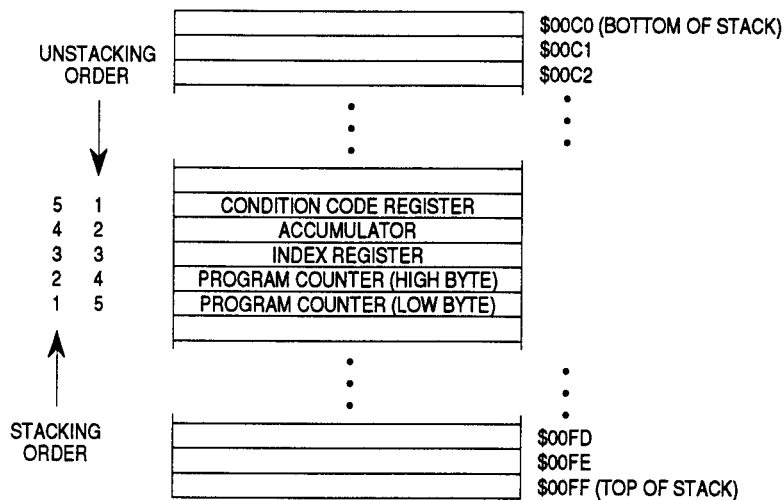
A timer overflow interrupt request occurs if the timer overflow flag, TOF, becomes set while the timer overflow interrupt enable bit, TOIE, is also set. TOF is in the timer status register, and TOIE is in the timer control register. (See **SECTION 8 CAPTURE/COMPARE TIMER.**)

## 4.2 Interrupt Processing

The CPU begins servicing an interrupt by taking the following actions:

- Stores the CPU registers on the stack in the order shown in Figure 4-2
- 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:
  - \$1FFC and \$1FFD (software interrupt vector)
  - \$1FFA and \$1FFB (external interrupt vector)
  - \$1FF8 and \$1FF9 (timer interrupt vector)

The return from interrupt (RTI) instruction causes the CPU to recover the CPU registers from the stack as shown in Figure 4-2.



**Figure 4-2. 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 = Highest)	Vector Address
Reset	Power-On RESET Pin COP Watchdog*	None	None	1 1 1	\$1FFE-\$1FFF
Software Interrupt (SWI)	User Code	None	None	Same Priority As Instruction	\$1FFC-\$1FFD
External Interrupt	$\overline{\text{IRQ}}$ Pin	None	1 Bit	2	\$1FFA-\$1FFB
Timer Interrupts	ICF Bit OCF Bit TOF Bit	ICIE Bit OCIE Bit TOIE Bit	1 Bit	3	\$1FF8-\$1FF9

\*The COP watchdog is a mask option.

Figure 4-3 shows the sequence of events caused by an interrupt.

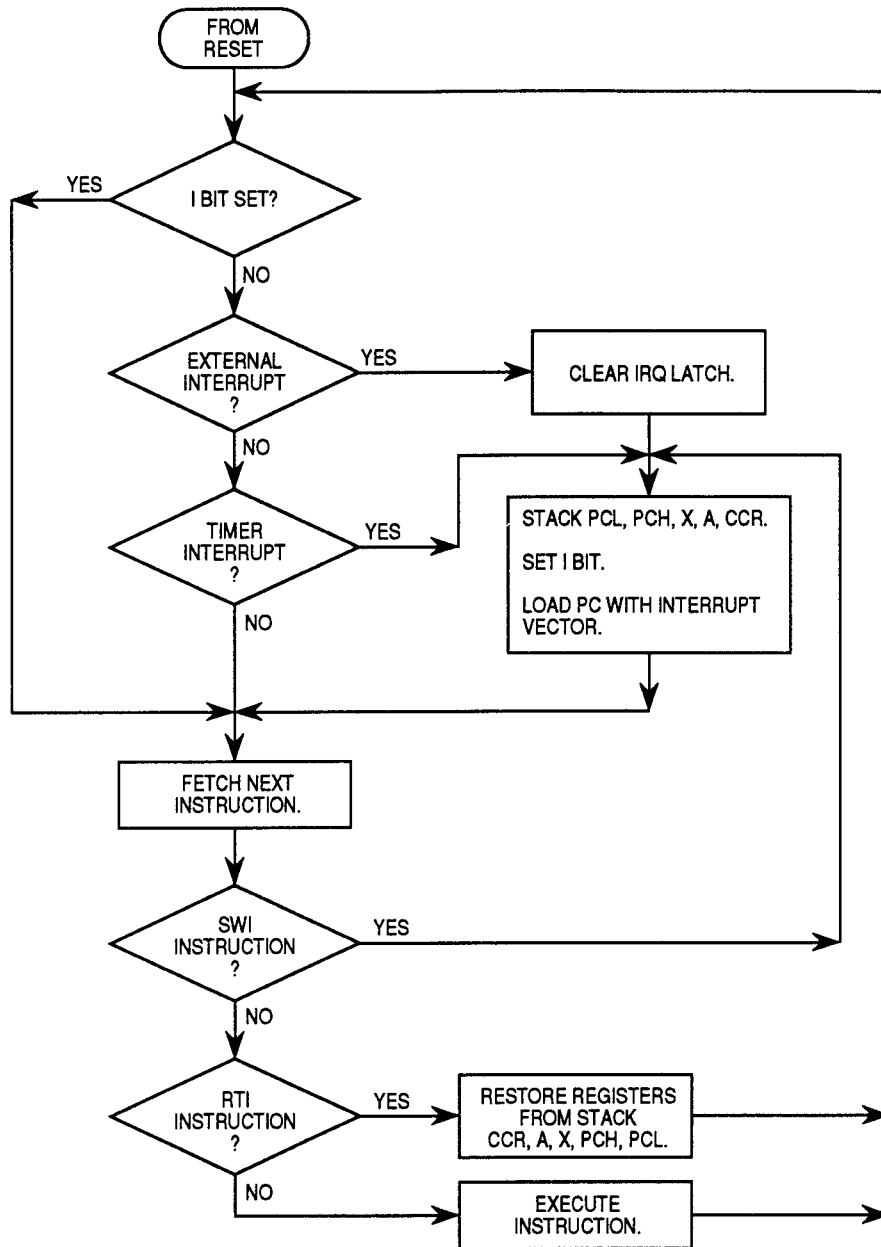


Figure 4-3. Interrupt Flowchart

## SECTION 5 RESETS

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

### 5.1 Reset Sources

The following sources can generate resets:

- Power-on reset (POR) circuit
- $\overline{\text{RESET}}$  pin
- COP watchdog

A reset immediately stops the execution of the instruction in progress, initializes certain control bits, and loads the program counter with a user-defined reset vector address. Figure 5-1 is a block diagram of the reset sources.

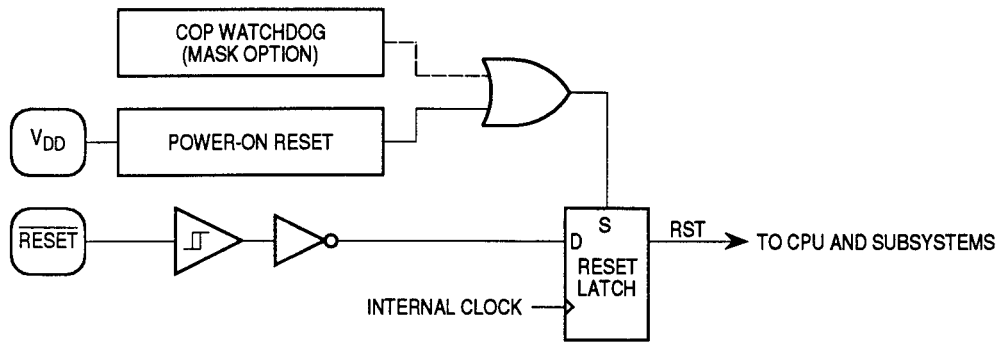
#### 5.1.1 Power-On Reset

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. (See Figure 5-1.)



**Figure 5-1. Reset Sources**

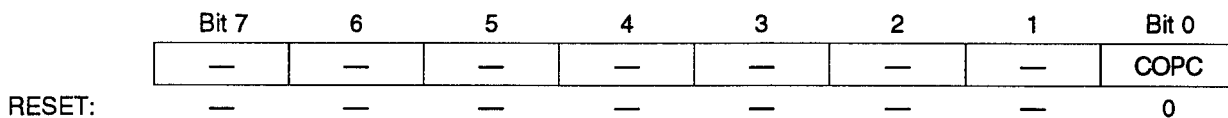
### 5.1.3 Computer Operating Properly (COP) Watchdog Reset

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. To clear the COP watchdog and prevent a COP reset, write a logic zero to bit 0 (COPC) of the COP register at location \$1FF0. The COP register, shown in Figure 5-2, is a write-only register that returns the contents of a ROM location when read.

The COP watchdog function is a mask option.

#### COPR — COP Register

**\$1FF0**



**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.

## 5.2 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
- Loads the program counter with the user-defined reset vector from locations \$1FFE and \$1FFF
- 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 data direction registers A, B, C, and D so that all I/O port pins are inputs (PD7/TCAP remains an input-only pin.)
- Has no effect on port A, port B, port C, or port D data registers

### 5.2.3 Capture/Compare Timer

A reset has the following effects on the capture/compare timer:

- Loads the timer counter with \$FFFC
- Clears the timer control register, except for the IEDG bit, with the following results:

- Clears the ICIE bit, inhibiting input capture interrupts
- Clears the OCIE bit, inhibiting output compare interrupts
- Clears the TOIE bit, inhibiting timer overflow interrupts
- Clears OLVL, the output compare bit
- Clears the TCMP pin
- Has no effect on the ICF, OCF, and TOF flags in the timer status register

#### **5.2.4 Serial I/O Port (SIOP)**

A reset clears the SIOP control and status registers and produces the following results:

- Clears the SPE bit, disabling the SIOP
- Clears the MSTR bit, configuring the disabled SIOP for slave mode operation
- Clears the SPIF and DCOL flags

#### **5.2.5 COP Watchdog**

A reset clears the COP watchdog timer.

#### **5.2.6 Analog-to-Digital Converter (ADC)**

A reset clears the ADC status and control register and produces the following results:

- Clears the ADON bit, disabling the ADC
- Clears the CCF flag
- Clears the ADRC bit, configuring the disabled ADC for operation at internal clock frequency
- Clears bits CH2–CH0, selecting channel 0 as the analog input

## **SECTION 6 LOW POWER MODES**

This section describes the four low-power modes:

- Stop mode
- Wait mode
- Halt mode (mask option)
- 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:

- Stops the internal oscillator, the CPU clock, and the internal clock, turning off the capture/compare timer, the COP watchdog, the SIO, and the ADC
- Clears the I bit in the condition code register, enabling external interrupts
- Clears the ICF, OCF, and TOF interrupt flags in the timer status register, removing any pending timer interrupts
- Clears the ICIE, OCIE, and TOIE bits in the timer control register, disabling further timer interrupts

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

The following events bring the MCU out of stop mode:

- External interrupt — A high-to-low transition on the  $\overline{\text{IRQ}}$  pin loads the program counter with the contents of locations \$1FFA and \$1FFB.
- External reset — A logic zero on the  $\overline{\text{RESET}}$  pin resets the MCU and loads the program counter with the contents of locations \$1FFE and \$1FFF.

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

If an external interrupt brings the MCU out of stop mode after an active edge occurred on the PD7/TCAP pin during stop mode, the ICF flag becomes set. An external interrupt also latches the value in the timer registers into the input capture registers.

If an external reset brings the MCU out of stop mode after an active edge occurred on the PD7/TCAP pin during stop mode, the ICF flag does not become set. An external reset has no effect on the input capture registers.

## 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.
- Stops the CPU clock, but allows the internal oscillator and internal clock to continue to run.

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

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

- External interrupt — A high-to-low transition on the  $\overline{\text{IRQ}}$  pin loads the program counter with the contents of locations \$1FFA and \$1FFB.
- Timer interrupt — Input capture, output compare, and timer overflow interrupts load the program counter with the contents of locations \$1FF8 and \$1FF9.



- COP watchdog reset — A timeout of the COP watchdog resets the MCU and loads the program counter with the contents of locations \$1FFE and \$1FFF. Software can enable timer 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 \$1FFE and \$1FFF.

### 6.3 Halt Mode

If the mask option to disable the STOP instruction is selected, a STOP instruction puts the MCU in halt mode. 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. If the mask option to disable the STOP instruction is selected, 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.



## **SECTION 7 PARALLEL I/O**

This section describes the four bidirectional I/O ports.

### **7.1 I/O Port Function**

Twenty bidirectional I/O pins and one input-only pin form four parallel I/O ports. The 20 bidirectional I/O pins are programmable as inputs or outputs through the four data direction registers.

#### **NOTE**

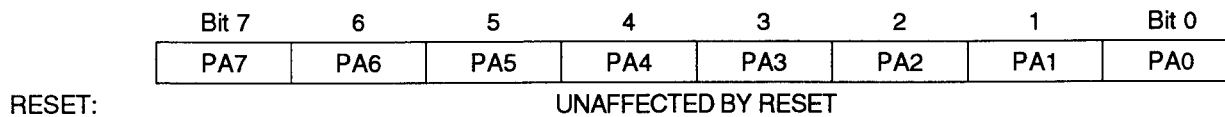
Connect any unused inputs and I/O pins to an appropriate logic level, either  $V_{DD}$  or  $V_{SS}$ . Although the I/O ports do not require termination for proper operation, termination reduces excess current consumption and the possibility of electrostatic damage.

### **7.2 Port A**

Port A is an 8-bit general-purpose bidirectional I/O port.

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

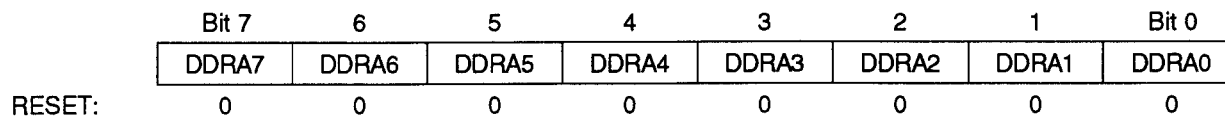
The port A data register, shown in Figure 7-1, contains a data latch for each of the eight port A pins.

**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 bit in data direction register A. Resets have no effect on port A data.

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

Data direction register A, shown in Figure 7-2, determines whether each port A pin is an input or an output. Writing a logic one to a DDRA bit enables the output buffer for the corresponding port A pin; a logic zero disables the output buffer.

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

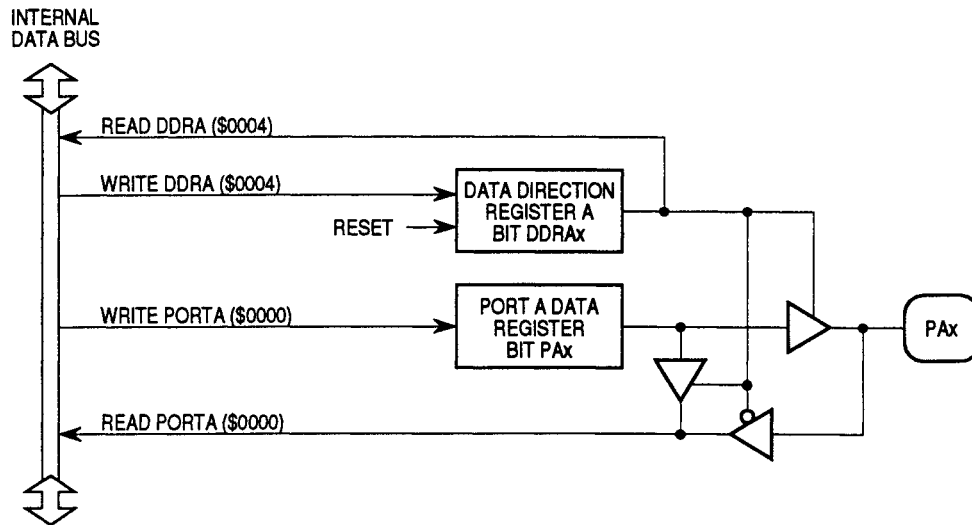
These read/write bits control port A data direction. A reset clears all DDRA bits, configuring all port A pins as inputs.

- 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 data direction register A bits from 0 to 1.

Figure 7-3 shows the port A I/O logic.



**Figure 7-3. Port A I/O Circuit**

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

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

DDRA Bit	PORTA Bit	I/O Pin Mode	Accesses to DDRA	Accesses to PORTA	
			Read/Write	Read	Write
0	X	Input, Hi-Z	DDRA7-0	Pin	NOTE 2
1	X	Output	DDRA7-0	PA7-0	PA7-0

**NOTES:**

1. X = don't care
2. Writing affects data register, but does not affect input
3. Hi-Z = high impedance

### 7.3 Port B

Port B is a 3-bit general-purpose bidirectional I/O port that shares its pins with the serial I/O port (SIOP) subsystem. Port B is available for general-purpose I/O functions when the SIOP is disabled. While the SIOP is enabled and a SIOP data transfer is in progress, writing to the port B data register or to bits DDRB7–DDR5 of data direction register B can corrupt the SIOP data. (See **9.2.1 SIOP Control Register (SCR)**.)

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

The port B data register, shown in Figure 7-4, contains a data latch for each of the three port B pins.

#### PORTB — Port B Data Register

**\$0001**

Bit 7	6	5	4	3	2	1	Bit 0
PB7	PB6	PB5	0	0	0	0	0

RESET: UNAFFECTED BY RESET

ALTERNATE FUNCTION: SCK SDI SDO

**Figure 7-4. Port B Data Register (PORTB)**

#### PB7–PB5 — Port B Data Bits

These read/write bits are software-programmable. Data direction of each bit is under the control of the corresponding bit in data direction register B. Resets have no effect on port B data.

#### Bits 4–0 — Not used

Bits 4–0 always read as logic zeros. Writes to these bits have no effect.

### 7.3.2 Data Direction Register B (DDRB)

Data direction register B determines whether each port B pin is an input or an output. (See Figure 7-5.) Writing a logic one to a DDRB bit enables the output buffer for the corresponding port B pin; a logic zero disables the output buffer.

**DDRC — Data Direction Register B**

**\$0005**

	Bit 7	6	5	4	3	2	1	Bit 0
	DDRB7	DDRB6	DDRB5	0	0	0	0	0
RESET:	0	0	0	0	0	0	0	0

**Figure 7-5. Data Direction Register B (DDRB)**

#### DDRB7–DDRB5 — Port B Data Direction Bits

These read/write bits control port B data direction. A reset clears DDRB7–DDRB5, configuring pins PB7–PB5 as inputs.

1 = Corresponding port B pin configured as output

0 = Corresponding port B pin configured as input

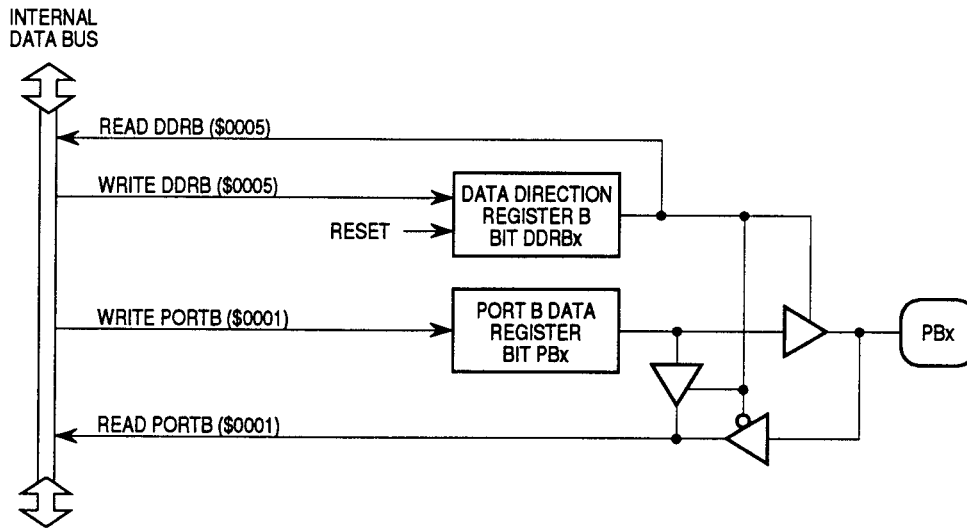
#### Bits 4–0 — Not used

Bits 4–0 always read as logic zeros. Writes to these bits have no effect.

#### NOTE

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

Figure 7-6 shows the port B I/O logic.



**Figure 7-6. Port B I/O Circuit**

When a port B 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. 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 DDRB bit. Table 7-2 summarizes the operation of the port B pins.

**Table 7-2. Port B Pin Functions**

DDRB Bit	PORTB Bit	I/O Pin Mode	Accesses to DDRB	Accesses to PORTB	
			Read/Write	Read	Write
0	X	Input, Hi-Z	DDRB7-5	Pin	NOTE 2
1	X	Output	DDRB7-5	PB7-5	PB7-5

**NOTES:**

1. X = don't care
2. Writing affects data register, but does not affect input
3. Hi-Z = high impedance



## 7.4 Port C

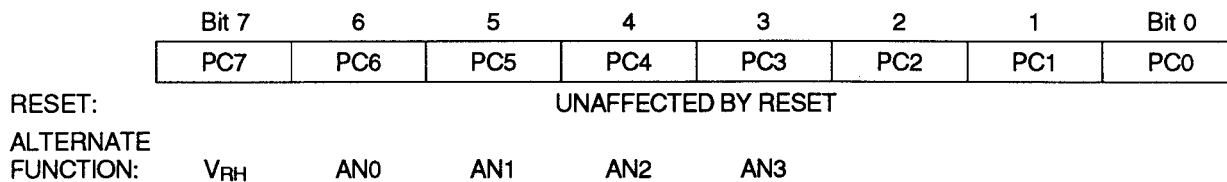
Port C is an 8-bit general-purpose bidirectional I/O port that shares five of its pins with the analog-to-digital converter (ADC) subsystem. The five shared pins are available for general-purpose I/O functions when the ADC is disabled. While the ADC is enabled, writing to bits PC7–PC5 of the port C data register or to bits DDRC7–DDRC5 of data direction register C can produce unpredictable ADC results. (See **10.2.1 ADC Status and Control Register (ADSCR)**.)

### 7.4.1 Port C Data Register (PORTC)

The port C data register, shown in Figure 7-7, contains a data latch for each of the eight port C pins.

**PORTC** — Port C Data Register

**\$0002**



**Figure 7-7. Port C Data Register (PORTC)**

#### PC7–PC3 — Port C Data Bits

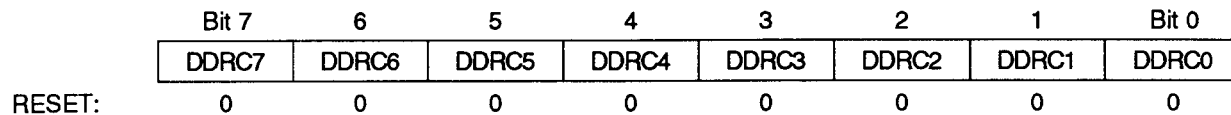
These read/write bits are software-programmable. Data direction of each bit is under the control of the corresponding bit in data direction register C. Resets have no effect on port C data.

## 7.4.2 Data Direction Register C (DDRC)

Data direction register C determines whether each port C pin is an input or an output. (See Figure 7-8.) Writing a logic one to a DDRC bit enables the output buffer for the corresponding port C pin; a logic zero disables the output buffer.

**DDRC** — Data Direction Register C

**\$0006**



**Figure 7-8. Data Direction Register C (DDRC)**

### DDRC7–DDRC0 — Port C Data Direction Bits

These read/write bits control port C data direction. A reset clears all DDRC bits, configuring all port C pins as inputs.

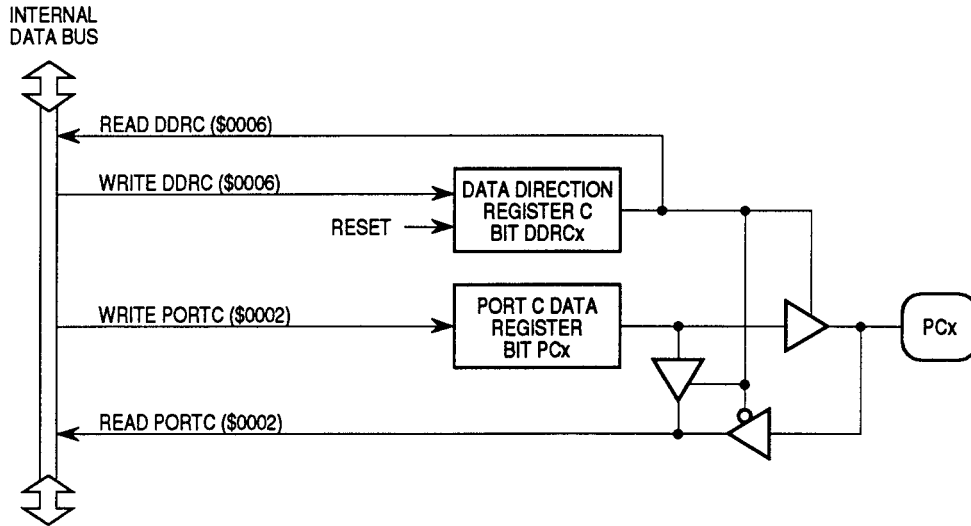
1 = Corresponding port C pin configured as output

0 = Corresponding port C pin configured as input

### NOTE

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

Figure 7-9 shows the port C I/O logic.



**Figure 7-9. Port C I/O Circuit**

When a port C 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. When a port C pin is programmed as an input, reading the port bit reads the voltage level on the pin. The data register can always be written, regardless of the state of its DDRC bit. Table 7-3 summarizes the operation of the port C pins.

**Table 7-3. Port C Pin Functions**

DDRC Bit	PORTC Bit	I/O Pin Mode	Accesses to DDRC		Accesses to PORTC	
			Read/Write	Read	Write	
0	X	Input, Hi-Z	DDRC7-0	Pin	NOTE 2	
1	X	Output	DDRC7-0	PC7-0	PC7-0	

**NOTES:**

1. X = don't care
2. Writing affects data register, but does not affect input
3. Hi-Z = high impedance

## 7.5 Port D

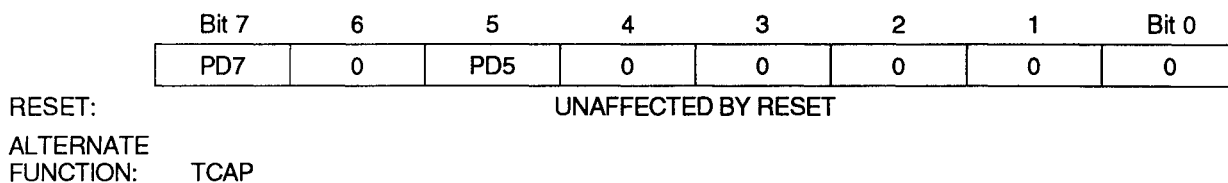
Port D is a 2-bit general-purpose I/O port with one bidirectional pin, PD5, and one input-only pin, PD7/TCAP. Port D shares pin PD7/TCAP with the capture/compare timer. PD7/TCAP is always available for general-purpose I/O functions, and the state of the pin can be read from the port D data register at any time.

### 7.5.1 Port D Data Register (PORTD)

The port D data register, shown in Figure 7-10, contains a data latch for each of the two port D pins.

#### PORTD — Port D Data Register

**\$0003**



**Figure 7-10. Port D Data Register (PORTD)**

#### PD7 — Port D Data Bit 7

This read/write bit is software-programmable. The PD7/TCAP pin can be a general-purpose input even when the timer is using it as the input capture pin. Resets have no effect on PD7.

#### PD5 — Port D Data Bit 5

This read/write bit is software-programmable. Data direction of PD5 is under the control of bit DDRD5 in data direction register D. Resets have no effect on PD5.

#### Bit 6 and bits 4–0 — Not used

Bit 6 and bits 4–0 always read as logic zeros. Writes to these bits have no effect.

## 7.5.2 Data Direction Register D (DDRD)

Bit DDRD5 in data direction register D, shown in Figure 7-11, determines whether pin PD5 is an input or an output. Writing a logic one to bit DDRD5 enables the output buffer for pin PD5; a logic zero disables the output buffer.

**DDRD** — Data Direction Register D

**\$0007**

	Bit 7	6	5	4	3	2	1	Bit 0
	0	0	DDRD5	0	0	0	0	0
RESET:	0	0	0	0	0	0	0	0

**Figure 7-11. Data Direction Register D (DDRD)**

**DDRD5** — PD5 Data Direction Bit

This read/write bit controls pin PD5 data direction. A reset clears DDRD5, configuring pin PD5 as an input.

1 = Pin PD5 configured as output

0 = Pin PD5 configured as input

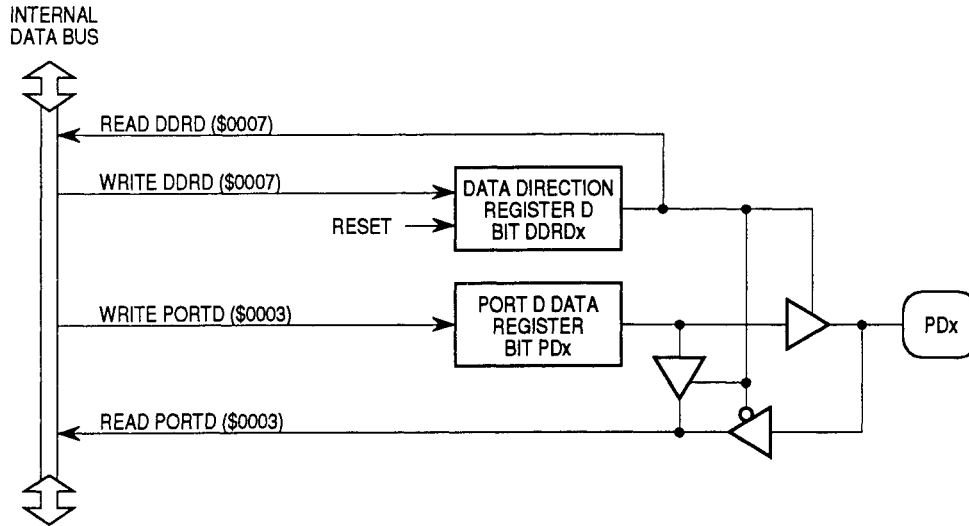
**Bits 7–6 and bits 4–0** — Not used

Bits 7–6 and 4–0 always read as logic zeros. Writes to these bits have no effect.

### **NOTE**

Avoid glitches on PD5 by writing to PD5 before changing DDRD5 from 0 to 1.

Figure 7-12 shows the port D I/O logic.



**Figure 7-12. Port D I/O Circuit**

When pin PD5 is programmed as an output, reading the port bit actually reads the value of the data latch and not the voltage on the pin. When pin PD5 is programmed as an input, reading bit PD5 reads the voltage level on the pin. Bit PD5 can always be written, regardless of the state of bit DDRD5 in data direction register D. Table 7-4 summarizes the operation of the port D pins.

**Table 7-4. Port D Pin Functions**

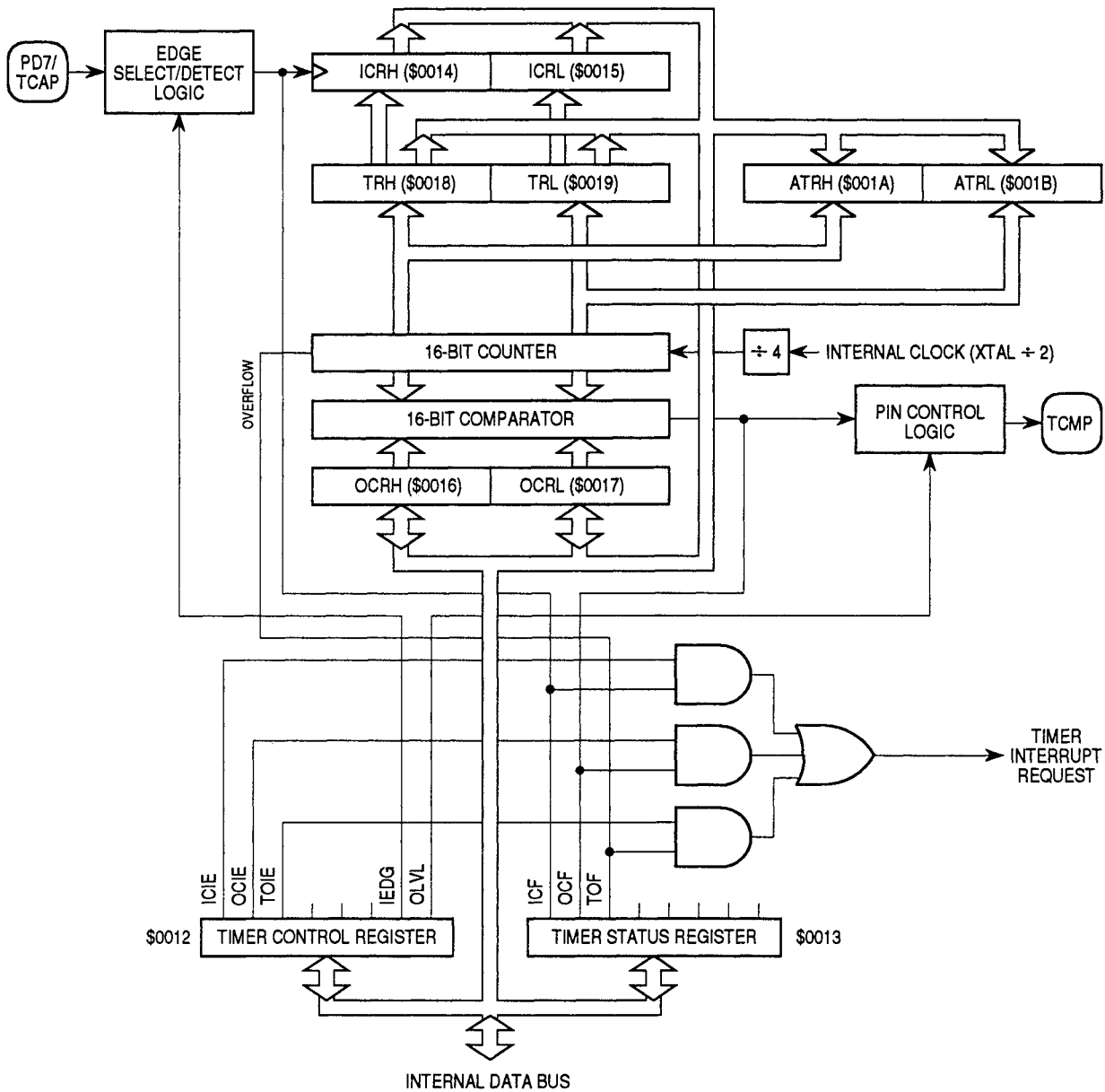
DDRD Bit	PORTD Bit	I/O Pin Mode	Accesses to DDRD	Accesses to PORTD	
			Read/Write	Read	Write
0	X	Input, hi-Z	DDRD5	Pin	NOTE 2
1	X	Output	DDRD5	PD5	PD5

NOTES:

1. X = don't care.
2. Writing affects data register, but does not affect input.

## SECTION 8 CAPTURE/COMPARE TIMER

This section describes the operation of the 16-bit capture/compare timer. Figure 8-1 shows the organization of the capture/compare timer subsystem.



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

## 8.1 Timer Operation

The core of the capture/compare timer is a 16-bit free-running counter. The counter is the timing reference for the input capture and output compare functions. The input capture and output compare functions provide a means to latch the times at which external events occur, to measure input waveforms, and to generate output waveforms and timing delays. Software can read the value in the 16-bit free-running counter at any time without affecting the counter sequence.

Because of the 16-bit timer architecture, the I/O registers for the input capture and output compare functions are pairs of 8-bit registers.

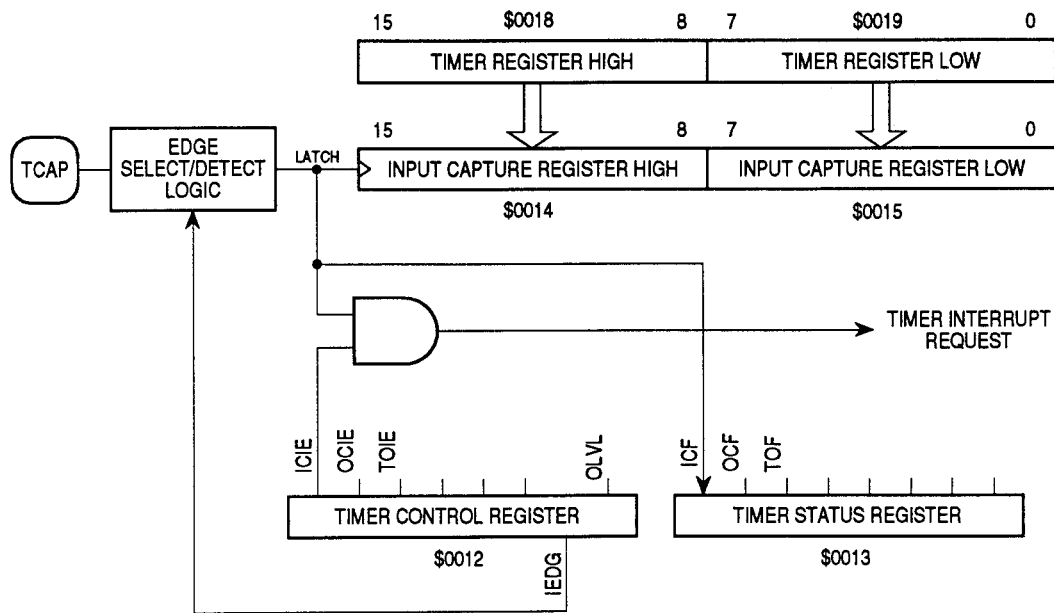
Because the counter is 16 bits long and preceded by a fixed divide-by-four prescaler, the counter rolls over every 262,144 internal clock cycles. Timer resolution with a 4 MHz crystal is 2  $\mu$ s.

### 8.1.1 Input Capture

The input capture function is a means to record the time at which an external event occurs. When the input capture circuitry detects an active edge on the PD7/TCAP pin, it latches the contents of the timer registers into the input capture registers. The polarity of the active edge is a mask option.

Latching values into the input capture registers at successive edges of the same polarity measures the period of the input signal on the PD7/TCAP pin. Latching the counter values at successive edges of opposite polarity measures the pulse width of the signal. Figure 8-2 shows the logic of the input capture function.



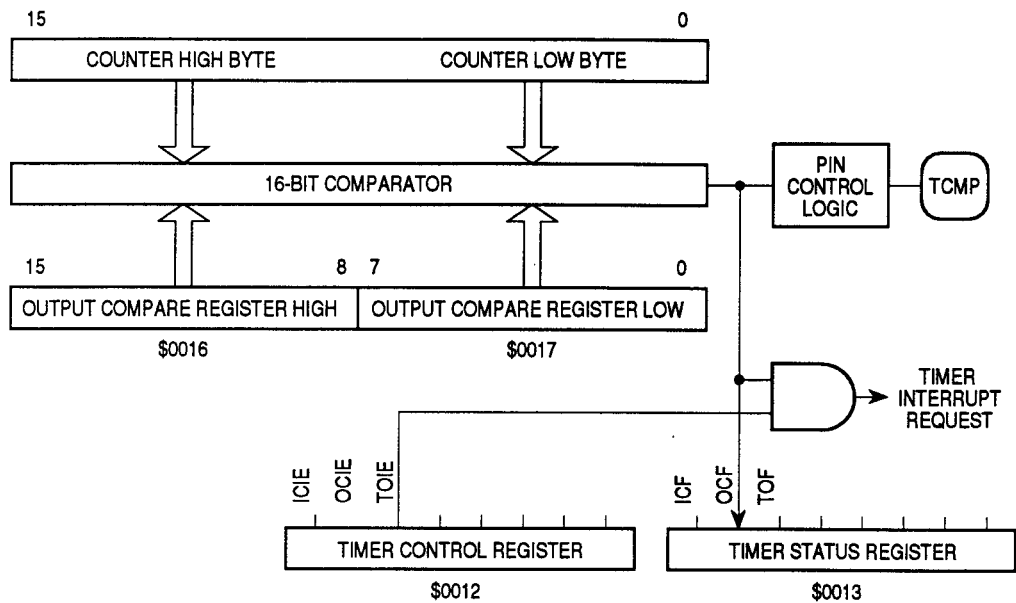


**Figure 8-2. Input Capture Operation**

### 8.1.2 Output Compare

The output compare function is a means of generating an output signal when the 16-bit counter reaches a selected value. Software writes the selected value into the output compare registers. On every fourth internal clock cycle the output compare circuitry compares the value of the counter to the value written in the output compare registers. When a match occurs, the timer transfers the programmable output level bit (OLVL) from the timer control register to the TCMP pin.

Software can use the output compare register to measure time periods, to generate timing delays, or to generate a pulse of specific duration or a pulse train of specific frequency and duty cycle on the TCMP pin. Figure 8-3 shows the logic of the output compare function.



**Figure 8-3. Output Compare Operation**

## 8.2 Timer I/O Registers

The following registers control and monitor the operation of the timer:

- Timer control register (TCR)
- Timer status register (TSR)
- Timer registers (TRH and TRL)
- Alternate timer registers (ATRH and ATRL)
- Input capture registers (ICRH and ICRL)
- Output compare registers (OCRH and OCRL)

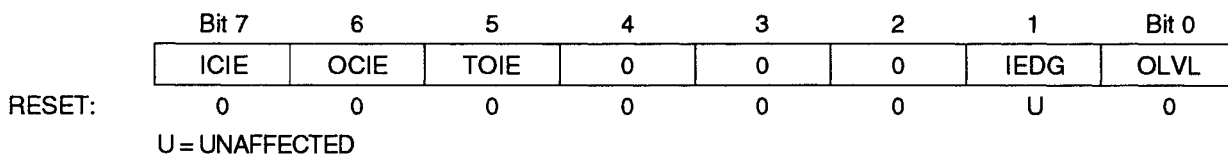
### 8.2.1 Timer Control Register (TCR)

The timer control register, shown in Figure 8-4, performs the following functions:

- Enables input capture interrupts
- Enables output compare interrupts
- Enables timer overflow interrupts
- Controls the active edge polarity of the TCAP signal
- Controls the active level of the TCMP output

#### TCR — Timer Control Register

**\$0012**



**Figure 8-4. Timer Control Register (TCR)**

#### ICIE — Input Capture Interrupt Enable

This read/write bit enables interrupts caused by active signal on the TCAP pin. Resets clear the ICIE bit.

- 1 = Input capture interrupts enabled
- 0 = Input capture interrupts disabled

#### OCIE — Output Compare Interrupt Enable

This read/write bit enables interrupts caused by an active signal on the TCMP pin. Resets clear the OCIE bit.

- 1 = Output compare interrupts enabled
- 0 = Output compare interrupts disabled

#### TOIE — Timer Overflow Interrupt Enable

This read/write bit enables interrupts caused by a timer overflow. Resets clear the TOIE bit.

- 1 = Timer overflow interrupts enabled
- 0 = Timer overflow interrupts disabled

## IEDG — Input Edge

The state of this read/write bit determines whether a positive or negative transition on the TCAP pin triggers a transfer of the contents of the timer register to the input capture register. Resets have no effect on the IEDG bit.

1 = Positive edge (low to high transition) triggers input capture

0 = Negative edge (high to low transition) triggers input capture

## OLVL — Output Level

The state of this read/write bit determines whether a logic one or a logic zero appears on the TCMP pin when a successful output compare occurs. Resets clear the OLVL bit.

1 = TCMP goes high on output compare

0 = TCMP goes low on output compare

## 8.2.2 Timer Status Register (TSR)

The timer status register, shown in Figure 8-5, contains flags for the following events:

- An active signal on the TCAP pin, transferring the contents of the timer registers to the input capture registers
- A match between the 16-bit counter and the output compare registers, transferring the OLVL bit to the TCMP pin
- A timer rollover from \$FFFF to \$0000

## TSR — Timer Status Register

**\$0012**

	Bit 7	6	5	4	3	2	1	Bit 0
	ICF	OCF	TOF	0	0	0	0	0
RESET:	U	U	U	0	0	0	0	0

U = UNAFFECTED

**Figure 8-5. Timer Status Register (TSR)**

## ICF — Input Capture Flag

The ICF bit is automatically set when an edge of the selected polarity occurs on the TCAP pin. Clear the ICF bit by reading the timer status register with ICF set, and then reading the low byte (\$0015) of the input capture registers. Resets have no effect on ICF.

### OCF — Output Compare Flag

The OCF bit is automatically set when the value of the timer registers matches the contents of the output compare registers. Clear the OCF bit by reading the timer status register with OCF set, and then accessing the low byte (\$0017) of the output compare registers. Resets have no effect on OCF.

### TOF — Timer Overflow Flag

The TOF bit is automatically set when the 16-bit counter rolls over from \$FFFF to \$0000. Clear the TOF bit by reading the timer status register with TOF set, and then accessing the low byte (\$0019) of the timer registers. Resets have no effect on TOF.

### 8.2.3 Timer Registers (TRH and TRL)

The timer registers, shown in Figure 8-6, contain the current high and low bytes of the 16-bit counter. Reading TRH before reading TRL causes TRL to be latched until TRL is read. Reading TRL after reading the timer status register clears the timer overflow flag (TOF). Writing to the timer registers has no effect.

#### TRH and TRL — Timer Register High/Low

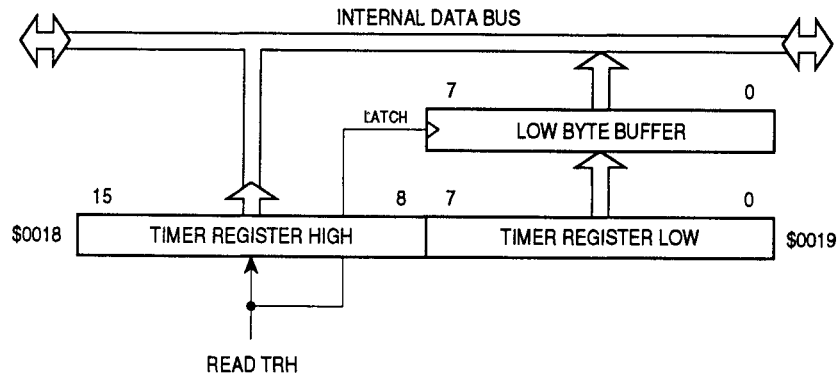
**\$0018 and \$0019**

\$0018	Bit 15	14	13	12	11	10	9	Bit 8
\$0019	Bit 7	6	5	4	3	2	1	Bit 0

Reset initializes the timer registers to \$FFFC.

**Figure 8-6. Timer Registers (TRH and TRL)**

Reading TRH returns the current value of the high byte of the counter and causes the low byte to be latched into a buffer, as shown in Figure 8-7. The buffer value remains fixed even if the high byte is read more than once. Reading TRL reads the transparent low byte buffer and completes the read sequence of the timer registers.



**Figure 8-7. Timer Register Reads**

**NOTE**

To prevent interrupts from occurring between readings of TRH and TRL, set the interrupt flag in the condition code register before reading TRH, and clear the flag after reading TRL.

**8.2.4 Alternate Timer Registers (ATRH and ATRL)**

The alternate timer registers, shown in Figure 8-8, contain the current high and low bytes of the 16-bit counter. Reading ATRH before reading ATRL causes ATRL to be latched until ATRL is read. Reading does not affect the timer overflow flag (TOF). Writing to the alternate timer registers has no effect.

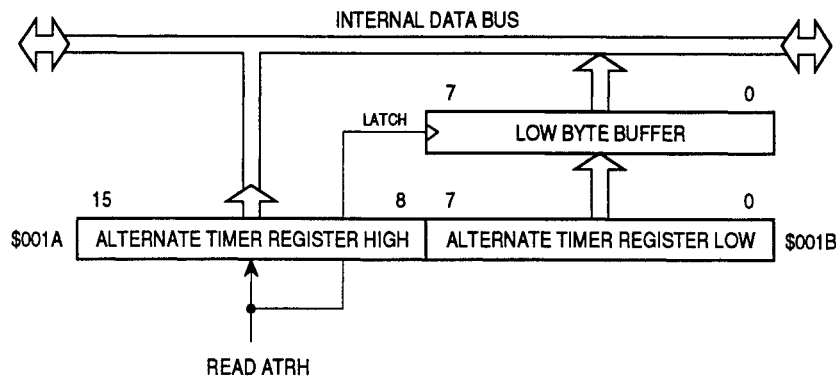
**ATRH and ATRL — Alternate Timer Register High/Low                      \$001A and \$001B**

\$001A	Bit 15	14	13	12	11	10	9	Bit 8
\$001B	Bit 7	6	5	4	3	2	1	Bit 0

Reset initializes the alternate timer registers to \$FFFC.

**Figure 8-8. Alternate Timer Registers (ATRH and ATRL)**

Reading ATRH returns the current value of the high byte of the counter and causes the low byte to be latched into a buffer, as shown in Figure 8-9. The buffer value remains fixed even if the high byte is read more than once. Reading ATRL reads the transparent low byte buffer and completes the read sequence of the alternate timer registers.



**Figure 8-9. Alternate Timer Register Reads**

**NOTE**

To prevent interrupts from occurring between readings of ATRH and ATRL, set the interrupt flag in the condition code register before reading ATRH, and clear the flag after reading ATRL.

**8.2.5 Input Capture Registers (ICRH and ICRL)**

When a selected edge occurs on the TCAP pin, the current high and low bytes of the 16-bit counter are latched into the input capture registers. Reading ICRH before reading ICRL inhibits further captures until ICRL is read. Reading ICRL after reading the timer status register clears the input capture flag (ICF). Writing to the input capture registers has no effect.

**ICRH and ICRL — Input Capture Register High/Low**

**\$0014 and \$0015**

\$0014	Bit 15	14	13	12	11	10	9	Bit 8
\$0015	Bit 7	6	5	4	3	2	1	Bit 0

Reset does not affect the input capture registers.

**Figure 8-10. Input Capture Registers (ICRH and ICRL)**

## NOTE

To prevent interrupts from occurring between readings of ICRH and ICRL, set the interrupt flag in the condition code register before reading ICRH, and clear the flag after reading ICRL.

### 8.2.6 Output Compare Registers (OCRH and OCRL)

When the value of the 16-bit counter matches the value in the output compare registers, the planned TCMP pin action takes place. Writing to OCRH before writing to OCRL inhibits timer compares until OCRL is written. Reading or writing to OCRL after reading the timer status register clears the output compare flag (OCF).

#### OCRH and OCRL — Output Compare Register High/Low **\$0016 and \$0017**

\$0016	Bit 15	14	13	12	11	10	9	Bit 8
\$0017	Bit 7	6	5	4	3	2	1	Bit 0

Reset does not affect the output compare registers.

#### Figure 8-11. Output Compare Registers (OCRH and OCRL)

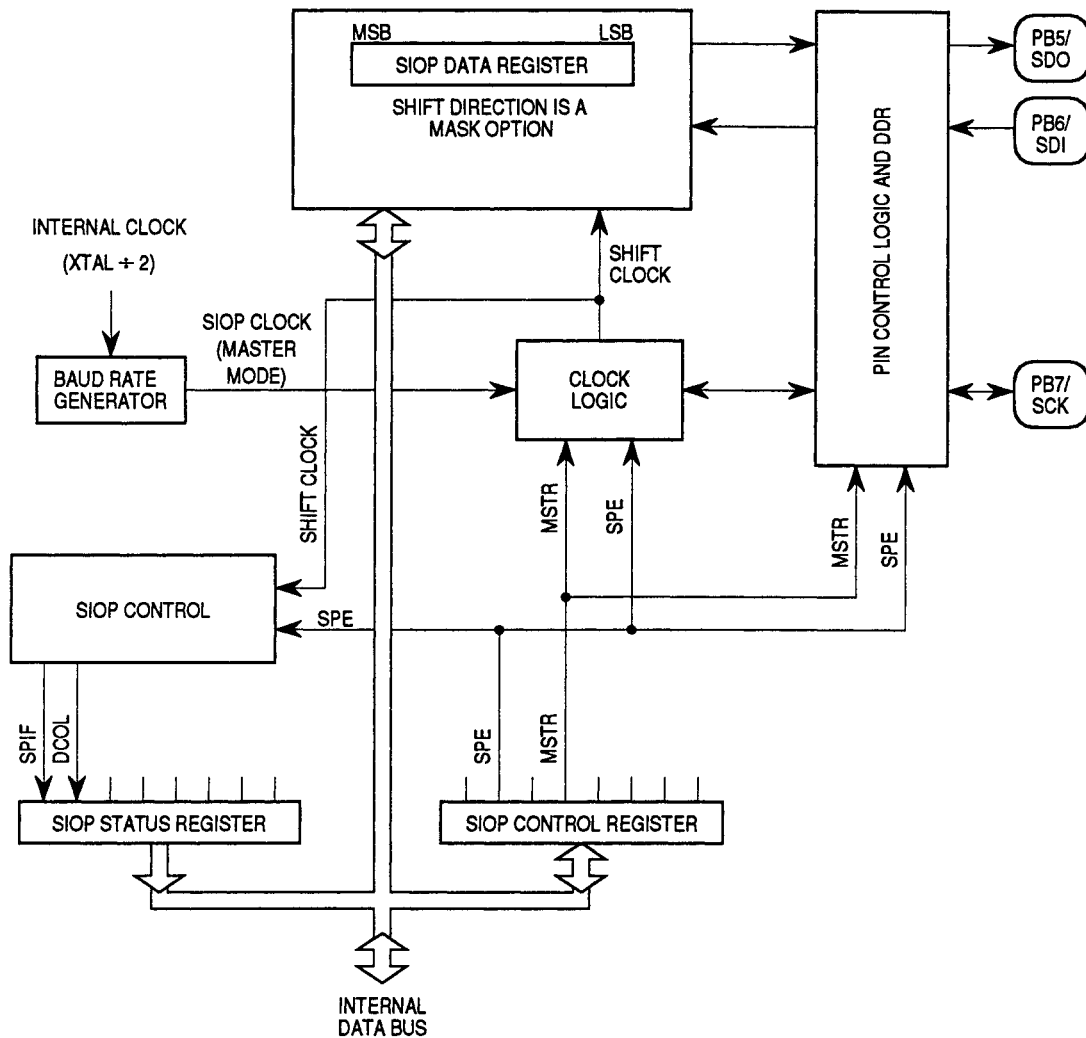
To prevent OCF from being set between the time it is read and the time the output compare registers are updated, use the following procedure:

1. Disable interrupts by setting the I bit in the condition code register.
2. Write to OCRH. Compares are now inhibited until OCRL is written.
3. Clear bit OCF by reading the timer status register (TSR).
4. Enable the output compare function by writing to OCRL.
5. Enable interrupts by clearing the I bit in the condition code register.



## SECTION 9 SERIAL I/O PORT (SIOP)

This section describes the operation of the serial I/O port (SIOP). Figure 9-1 shows the structure of the SIOP subsystem.



**Figure 9-1. SIOP Block Diagram**

## 9.1 SIOP Operation

The SIOP enables high-speed synchronous serial data transmission between the MCU and peripheral devices. Shift registers used with the SIOP can increase the number of parallel I/O pins controlled by the MCU. More powerful peripherals such as analog-to-digital converters and real-time clocks are also compatible with the SIOP. A mask option determines whether the SIOP transmits data MSB-first or LSB-first.

### 9.1.1 SIOP Pin Functions

The SIOP uses the three port B I/O pins. Setting the SPE bit in the SIOP control register enables the SIOP. When the SPE bit is set, the PB7/SCK, PB6/SDI, and PB5/SDO pins are dedicated to SIOP functions. When the SPE bit is clear, the PB7/SCK, PB6/SDI, and PB5/SDO pins are bidirectional port B I/O pins.

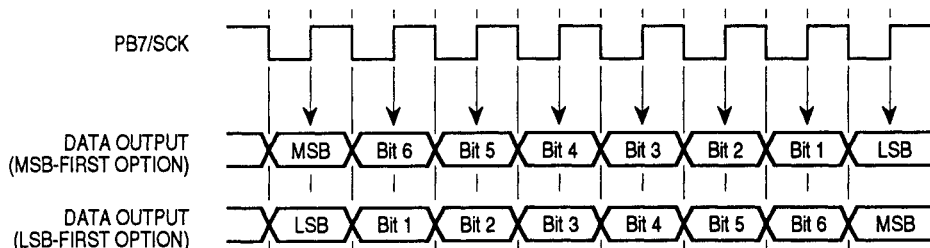
Setting the MSTR bit in the SIOP control register configures the SIOP for master mode. In master mode, the PB7/SCK pin is the serial clock output. PB6/SDI is the serial data input pin, and PB5/SDO is the serial data output pin. The master MCU initiates and controls the transmission of data to and from one or more slave peripheral devices. In master mode, a transmission is initiated by writing to the SIOP data register. Data written to the SIOP data register is parallel-loaded and shifted out serially to the slave device(s).

## 9.1.2 Serial Clock

The PB7/SCK pin synchronizes the movement of data into and out of the MCU through the PB6/SDI and PB5/SDO pins. In master mode, the PB7/SCK pin is an output. The transmission rate for master mode is a mask option.

In slave mode, the PB7/SCK pin is an input. The maximum serial clock rate for slave mode is the maximum internal clock rate divided by four. There is no minimum serial clock frequency for slave mode.

Figure 9-2 shows the timing relationships between the serial clock, data input, and data output. The state of the serial clock between transmissions is a logic one. The first falling edge on the PB7/SCK pin signals the beginning of a transmission, and data appears at the PB5/SDO pin. Data is captured at the PB6/SDI pin on the rising edge of the serial clock, and the transmission ends on the eighth rising edge of the serial clock.



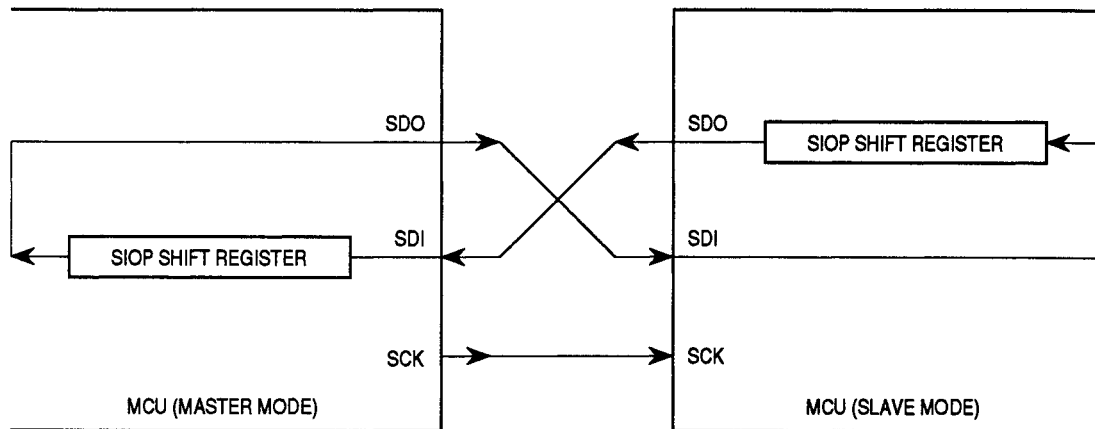
**Figure 9-2. SIOP Data/Clock Timing**

Valid SDI data must be present for an SDI setup time before the rising edge of the serial clock and must remain valid for an SDI hold time after the rising edge of the serial clock. (See **13.9 SIOP Timing ( $V_{DD} = 5.0$  Vdc)** and **13.10 SIOP Timing ( $V_{DD} = 3.3$  Vdc).**)

Between transmissions, the state of the PB5/SDO pin reflects the value of the last bit received on the previous transmission. On the first falling edge on the PB7/SCK pin, the first data bit to be shifted out appears at the PB5/SDO pin.

### 9.1.3 Data Movement

Connecting the SIOP data register of a master MCU with the SIOP of a slave MCU forms a 16-bit circular shift register. During a SIOP transmission, the master shifts out the contents of its SIOP data register on its PB5/SDO pin. At the same time, the slave MCU shifts out the contents of its SIOP data register on its SDO pin. Figure 9-3 shows how the master and slave exchange the contents of their data registers.



NOTE: Both MCUs shown are programmed for MSB-first data format.

**Figure 9-3. Master/Slave SIOP Shift Register Operation**

## 9.2 SIOP I/O Registers

The following registers control and monitor SIOP operation:

- SIOP control register (SCR)
- SIOP status register (SSR)
- SIOP data register (SDR)

### 9.2.1 SIOP Control Register (SCR)

The read/write SIOP control register, shown in Figure 9-4, contains two bits. One bit enables the SIOP, and the other configures the SIOP for master mode or for slave mode.

	Bit 7	6	5	4	3	2	1	Bit 0
	0	SPE	0	MSTR	0	0	0	0
RESET:	0	0	0	0	0	0	0	0

Figure 9-4. SIOP Control Register (SCR)

SPE — SIOP Enable

This read/write bit enables the SIOP. Clearing the SPE bit during a transmission aborts the transmission and returns port B to its normal I/O function. After clearing the SPE bit, be sure to initialize the port B data direction register for the intended port B I/O use. Resets clear SPE.

- 1 = SIOP enabled
- 0 = SIOP disabled

MSTR — Master Mode Select

This read/write bit configures the SIOP for master mode. Setting MSTR initializes the PB7/SCK pin as the serial clock output. Clearing MSTR initializes the PB7/SCK pin as the serial clock input. Resets clear MSTR.

- 1 = Master mode selected
- 0 = Slave mode selected

9.2.2 SIOP Status Register (SSR)

The read/write SIOP status register, shown in Figure 9-5, contains two bits. One bit indicates that a SIOP transmission is complete, and the other indicates that an access of the SIOP status register occurred while a transmission was in progress.

SSR — SIOP Status Register

	Bit 7	6	5	4	3	2	1	Bit 0
	SPIF	DCOL	0	0	0	0	0	0
RESET:	0	0	0	0	0	0	0	0

Figure 9-5. SIOP Status Register (SCR)

### SPIF — SIOPI Interrupt Flag

This clearable, read-only bit is set automatically at the end of a transmission. Clear the SPIF bit by reading the SIOPI status register with SPIF set, and then reading or writing the SIOPI data register. Resets clear the SPIF bit.

- 1 = Serial transmission complete
- 0 = Serial transmission not complete

### DCOL — Data Collision

This clearable, read-only bit is set if the SIOPI data register is read or written during a transmission. Clear the DCOL bit by reading the SIOPI status register with the SPIF bit set, and then reading or writing the SIOPI data register. Resets clear the DCOL bit.

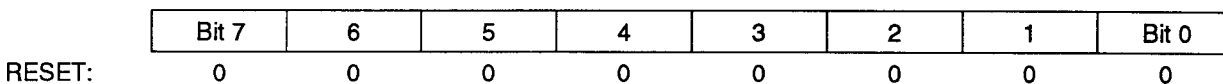
- 1 = Invalid access of SIOPI status register
- 0 = No invalid access of SIOPI status register

### 9.2.3 SIOPI Data Register (SDR)

The SIOPI data register, shown in Figure 9-6, is both the transmit data register and the receive data register. To read or write the SIOPI data register, the SPE bit in the SIOPI control register must be set.

### SDR — SIOPI Data Register

**\$000C**



**Figure 9-6. SIOPI Data Register (SDR)**

With the SIOPI configured as master, writing to the SIOPI data register initiates a serial transmission. This register is not buffered. Writing to the SIOPI data register overwrites the previous contents. Reading or writing to the SIOPI data register while a transmission is in progress can cause invalid data to be transmitted or received.

## SECTION 10 ANALOG-TO-DIGITAL CONVERTER (ADC)

This section describes the four-channel, 8-bit, successive approximation ADC.

### 10.1 ADC Operation

Figure 10-1 shows the structure of the ADC.

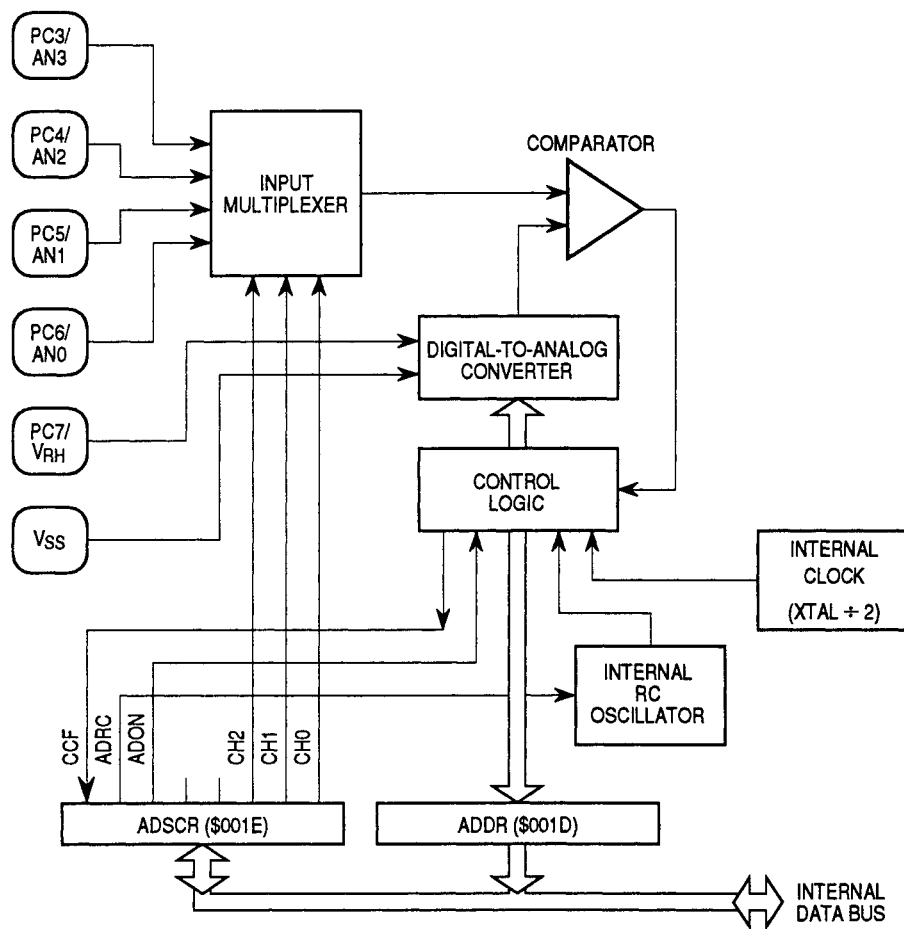


Figure 10-1. ADC Block Diagram

A multiplexer selects one of four external analog signals for sampling. The CH2–CH0 bits in the ADC status and control register (ADSCR) control input selection.

A comparator makes successive comparisons of the selected analog input and the output of a precision digital-to-analog converter (DAC). Control logic changes the input to the DAC one bit at a time, starting with the MSB, until the DAC output matches the selected analog input. The conversion is monotonic and has no missing codes. At the end of 32 internal clock cycles, the conversion complete (CC) flag becomes set, and the digital conversion is readable in the ADC data register (ADDR).

For ratiometric conversion, the supply voltage of the analog source should be the same as  $V_{RH}$  and be referenced to  $V_{SS}$ . An analog input voltage equal to  $V_{RH}$  converts to digital \$FF; an input voltage greater than  $V_{RH}$  converts to \$FF with no overflow. An analog input voltage equal to  $V_{SS}$  converts to digital \$00.

### 10.1.1 Pin Functions

Port C shares five of its pins with the ADC. The following paragraphs describe the ADC pin functions.

#### 10.1.1.1 PC7/ $V_{RH}$

The PC7/ $V_{RH}$  pin supplies the high reference voltage for the ratiometric conversion process. The low reference is connected internally to  $V_{SS}$ .

#### 10.1.1.2 PC6/AN0, PC5/AN1, PC4/AN2, and PC3/AN3

PC6/AN0, PC5/AN1, PC4/AN2, and PC3/AN3 are the analog inputs to the ADC.

### 10.1.2 Conversion Accuracy

Conversion accuracy of  $\pm 1.5$  LSB is guaranteed when  $V_{RH} = V_{DD}$ .

### 10.1.3 Conversion Time

Each input conversion takes 32 internal clock cycles. The internal clock frequency must be equal to or greater than 1 MHz.



#### 10.1.4 Internal RC Oscillator

If the internal clock frequency is less than 1 MHz, the internal RC oscillator must be used for the ADC clock instead of the internal clock. The nominal frequency of the RC oscillator is 1.5 MHz. Select the internal RC oscillator by setting the ADRC bit in the ADSCR.

Since the RC oscillator is not synchronous with the internal clock, software must rely on the CC bit to determine when each conversion process is complete.

Using the RC oscillator may slightly degrade ADC accuracy. The ADC reads voltages only during the periods when the clock driving it is not changing. But since the RC oscillator and the internal clock are not synchronous, the ADC occasionally reads voltages during internal clock transitions.

When the internal clock frequency is 1 MHz or greater, the RC oscillator must be turned off.

### 10.2 ADC I/O Registers

The following registers control and monitor operation of the ADC:

- ADC status and control register (ADSCR)
- ADC data register (ADDR)

#### 10.2.1 ADC Status and Control Register (ADSCR)

The ADC status and control register, shown in Figure 10-2, contains a conversion complete flag and performs the following control functions:

- Turns on the ADC
- Turns on the internal RC oscillator
- Selects the analog inputs

Writing to ADSCR clears the conversion complete flag and starts a new conversion sequence.

	Bit 7	6	5	4	3	2	1	Bit 0
	CC	ADRC	ADON	0	0	CH2	CH1	CH0
RESET:	0	0	0	0	0	0	0	0

**Figure 10-2. ADC Status and Control Register (ADSCR)**

#### CC — Conversion Complete

This read-only bit is automatically set at the end of each conversion process. When CC is set, the result of the conversion is readable in the ADDR. Clear the CC bit by writing to the CH2–CH0 bits or by clearing the ADON bit or by reading the ADDR. Resets clear CC.

- 1 = Conversion complete
- 0 = Conversion not complete

#### ADRC — ADC RC (Oscillator)

This read/write bit turns on the internal RC oscillator to drive the ADC. If the internal clock frequency ( $f_{OP}$ ) is less than 1 MHz, ADRC must be set. When the RC oscillator is turned on, it requires a time,  $t_{ADRC}$ , to stabilize, and results can be inaccurate during this time. Resets clear ADRC.

- 1 = Internal RC oscillator drives ADC
- 0 = Internal clock drives ADC

#### ADON — ADC On

This read/write bit turns on the ADC. When the ADC is on, it requires a time,  $t_{ADON}$ , for the current sources to stabilize. During this time, results can be inaccurate. Resets clear ADON.

- 1 = ADC turned on
- 0 = ADC turned off

#### Bits 4–2 — Not used

Bits 4–2 always read as logic zeros.

## CH2-CH0 — Channel 2-Channel 0

These read/write bits select one of eight ADC input channels as shown in Table 10-1. Channels 0-3 are the port C input pins, PC3/AN3, PC4/AN2, PC5/AN1, and PC6/AN0. Channels 4-6 can be used for reference measurements. Channel 7 is reserved for factory testing.

**Table 10-1. ADC Input Channel Selection**

CH[2:1:0]	Channel	Signal
000	0	AN0 Port C Bit 6
001	1	AN1 Port C Bit 5
010	2	AN2 Port C Bit 4
011	3	AN3 Port C Bit 3
100	4	V <sub>RH</sub> Port C Bit 7
101	5	$(V_{RH} + V_{SS}) \div 2$
110	6	V <sub>SS</sub>
111	7	Reserved

To prevent excess power dissipation, do not use a port C pin as an analog input and a digital input at the same time.

Using one of the port C pins as the ADC input does not affect the ability to use the remaining port C pins as digital inputs.

Reading a port C pin that is selected as an analog input returns a logic zero.



## SECTION 11 SELF-CHECK ROM

This section describes how to use the self-check ROM to test MCU operation.

### 11.1 Self-Check Tests

To check for proper MCU operation, the self-check ROM performs the following tests:

- I/O test — A functional exercise of ports A, B, and C
- RAM test — A counter test for each page zero RAM byte
- Capture/compare timer test — A test of the counter register and the output compare function
- ROM test — An exclusive OR odd parity check
- ADC test — A test of reference voltage levels
- Interrupt test — A test of internal interrupts and the RTI instruction

Figure 11-1 shows the circuit required to execute the self-check tests.

### 11.2 Self-Check Results

Table 11-1 shows the LED codes that indicate self-check test results.

**Table 11-1. Self-Check Circuit LED Codes**

PC3	PC2	PC1	PC0	Problem
ON	OFF	OFF	ON	I/O Failure
ON	OFF	OFF	OFF	RAM Failure
OFF	ON	ON	ON	Timer Failure
OFF	ON	ON	OFF	ROM Failure
OFF	ON	OFF	ON	ADC Failure
OFF	ON	OFF	OFF	Interrupt Failure
Flashing				No Failures
All Other Patterns				Device Failure

### 11.3 Self-Check Circuit

Figure 11-1 shows the self-check circuit.

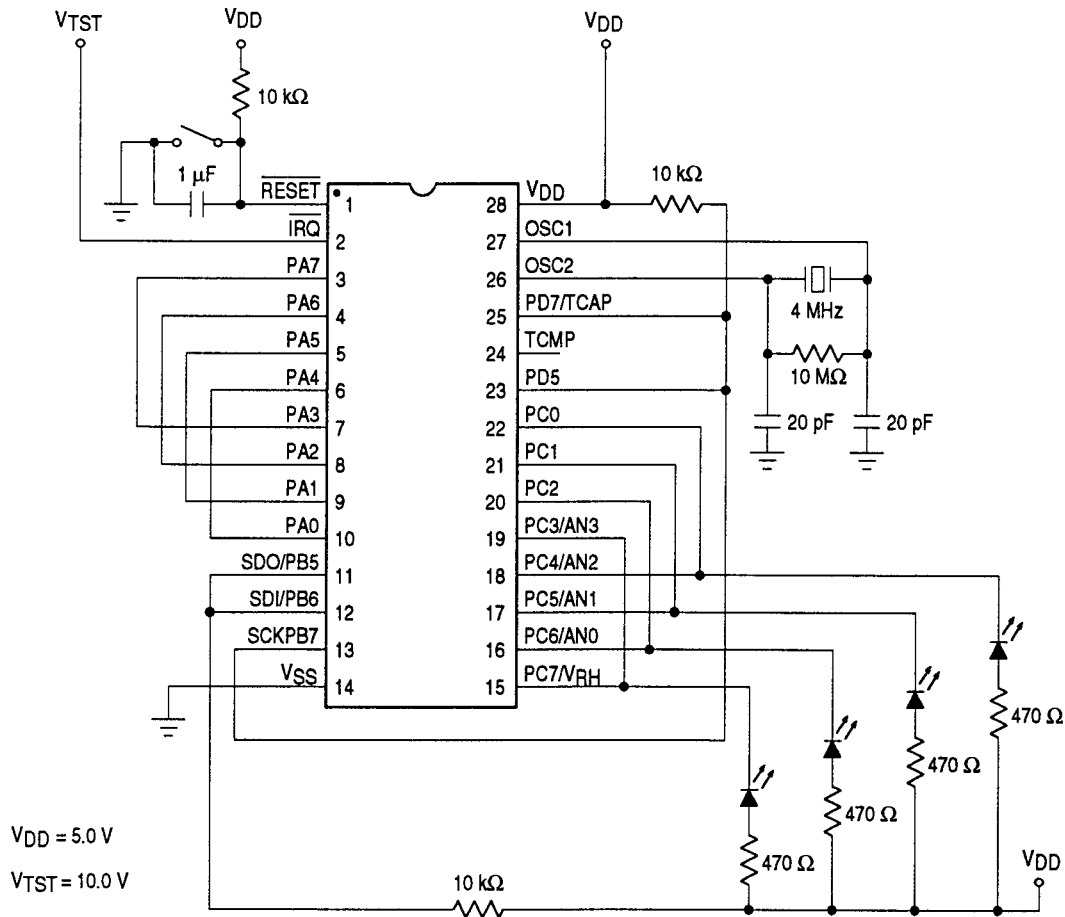


Figure 11-1. Self-Check Circuit

Perform the following steps to activate the self-check tests:

1. Apply  $V_{TST}$  to the  $\overline{IRQ}$  pin.
2. Apply a logic zero to the  $\overline{RESET}$  pin.
3. Apply a logic one to the PD1 pin.
4. Apply a logic one to the PB2 pin.
5. Apply a logic one to the  $\overline{RESET}$  pin.

The self-check tests begin on the rising edge of the  $\overline{RESET}$  pin.

## SECTION 12 INSTRUCTION SET

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

### 12.1 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

## 12.1.1 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 12-1 lists the instructions that use inherent addressing.

**Table 12-1. 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 Index Register by Accumulator (Unsigned)	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 $\overline{IRQ}$ 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 Interrupts and Halt CPU	WAIT



## 12.1.2 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 12-2 lists the instructions that use immediate addressing.

**Table 12-2. Immediate Addressing Instructions**

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

### 12.1.3 Direct

Direct instructions can access any of the first 256 memory addresses with two bytes. The first byte is the opcode, and the second is the low byte of the operand address. In direct addressing, 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 12-3 lists the instructions that use direct addressing.

**Table 12-3. Direct Addressing Instructions**

<b>Instruction</b>	<b>Mnemonic</b>
Add Memory and Carry to Accumulator	ADC
Add Memory to Accumulator	ADD
Logical AND Memory with Accumulator	AND
Arithmetic Shift Left	ASL
Arithmetic Shift Right	ASR
Clear Bit	BCLR
Bit Test Memory with Accumulator (Logical Compare)	BIT
Branch if Bit Clear	BRCLR
Branch if Bit Set	BRSET
Set Bit	BSET
Clear	CLR
Arithmetic Compare Accumulator with Memory	CMP
Complement	COM
Arithmetic 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
Logical Inclusive OR Memory with Accumulator	ORA
Rotate Left through Carry	ROL
Rotate Right through Carry	ROR
Subtract Memory and Carry from Accumulator	SBC
Store Accumulator in Memory	STA
Store Index Register in Memory	STX
Subtract Memory from Accumulator	SUB
Test for Negative or Zero	TST

#### 12.1.4 Extended

Extended instructions use only three bytes to access any address in memory. 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 12-4 lists the instructions that use extended addressing.

**Table 12-4. Extended Addressing Instructions**

<b>Instruction</b>	<b>Mnemonic</b>
Add Memory and Carry to Accumulator	ADC
Add Memory to Accumulator	ADD
Logical AND Memory with Accumulator	AND
Bit Test Memory with Accumulator (Logical Compare)	BIT
Arithmetic Compare Accumulator with Memory	CMP
Arithmetic 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
Logical Inclusive OR Memory with Accumulator	ORA
Subtract Memory and Carry from Accumulator	SBC
Store Accumulator in Memory	STA
Store Index Register in Memory	STX
Subtract Memory from Accumulator	SUB

### 12.1.5 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 conditional address of the operand. The CPU automatically uses \$00 as the high byte, 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 12-5 lists the instructions that use indexed, no offset addressing.

### 12.1.6 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 access locations \$0000–\$01FE.

Indexed, 8-bit offset instructions are useful for selecting the kth 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 12-5 lists the instructions that use indexed, 8-bit offset addressing.

### 12.1.7 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 kth 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 12-5 lists the instructions that use indexed, 16-bit offset addressing.

**Table 12-5. Indexed Addressing Instructions**

Instruction	Mnemonic	No Offset	8-Bit Offset	16-Bit Offset
Add Memory and Carry to Accumulator	ADC	√	√	√
Add Memory to Accumulator	ADD	√	√	√
Logical AND Memory with Accumulator	AND	√	√	√
Arithmetic Shift Left	ASL	√	√	
Arithmetic Shift Right	ASR	√	√	
Bit Test Memory with Accumulator (Logical Compare)	BIT	√	√	√
Clear	CLR	√	√	
Arithmetic Compare Accumulator with Memory	CMP	√	√	√
Complement	COM	√	√	
Arithmetic 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	√	√	
Logical Inclusive OR Memory with Accumulator	ORA	√	√	√
Rotate Left through Carry	ROL	√	√	
Rotate Right through Carry	ROR	√	√	
Subtract Memory and Carry from Accumulator	SBC	√	√	√
Store Accumulator in Memory	STA	√	√	√
Store Index Register in Memory	STX	√	√	√
Subtract Memory from Accumulator	SUB	√	√	√
Test for Negative or Zero	TST	√	√	

### 12.1.8 Relative

Relative addressing is only for branch instructions and bit test and branch instructions. If the branch condition is true, 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 not true, the CPU goes to the next instruction. The offset is a signed, two's complement byte that gives a branching range of  $-128$  to  $+127$  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 12-6 lists the instructions that use relative addressing.

**Table 12-6. 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 Clear	BRCLR
Branch if Bit Set	BRSET
Branch Never	BRN
Branch to Subroutine	BSR

## 12.2 Instruction Types

The MCU instructions fall into the following five categories:

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

### 12.2.1 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. Register/memory instructions use all the addressing modes except relative.

Table 12-7 lists the register/memory instructions.

**Table 12-7. 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 from Accumulator	SUB
Subtract Memory and Carry from Accumulator	SBC
Logical AND Memory with Accumulator	AND
Logical Inclusive OR Memory with Accumulator	ORA
Logical Exclusive OR Memory with Accumulator	EOR
Arithmetic Compare Accumulator with Memory	CMP
Arithmetic Compare Index Register with Memory	CPX
Bit Test Memory with Accumulator (Logical Compare)	BIT
Multiply Index Register by Accumulator (Unsigned)	MUL

### 12.2.2 Read-Modify-Write Instructions

These instructions read a memory location or a register, modify its contents, and write the modified value back to the memory location 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 12-8 lists the read-modify-write instructions.

**Table 12-8. 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 (Same as ASL)	LSL
Logical Shift Right	LSR
Arithmetic Shift Left (Same as LSL)	ASL
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST

### 12.2.3 Jump/Branch Instructions

Jump instructions allow the CPU to interrupt the normal sequence of the program counter. The unconditional jump (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 use relative addressing.

Bit test and branch instructions cause a branch based on the state 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 12-9 lists the jump and branch instructions.

**Table 12-9. Jump and Branch Instructions**

Instruction	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Bit Clear	BRCLR
Branch if Bit Set	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 (Same as BLO)	BCS
Branch if Lower (Same as BCS)	BLO
Branch if Not Equal	BNE
Branch if Equal	BEQ
Branch if Half-Carry Bit Clear	BHCC
Branch if Half-Carry Bit 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	JMP
Jump to Subroutine	JSR

## 12.2.4 Bit Manipulation Instructions

The CPU can set or clear any writable bit in the first 256 bytes of memory. Port registers, 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 direct addressing. Table 12-10 lists these instructions.

**Table 12-10. Bit Manipulation Instructions**

Instruction	Mnemonic
Set Bit	BSET
Clear Bit	BCLR
Branch if Bit Clear	BRCLR
Branch if Bit Set	BRSET

## 12.2.5 Control Instructions

These register reference instructions control CPU operation during program execution. Control instructions, listed in Table 12-11, use inherent addressing.

**Table 12-11. 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

## 12.3 Instruction Set Summary

Table 12-12 shows all MC68HC05P6 instructions in all possible addressing modes. The table shows the operand construction and the execution time in internal clock cycles ( $t_{CYC}$ ) of each instruction. One internal clock cycle equals two oscillator input cycles. The following legend summarizes the symbols and abbreviations used in Table 12-12.

### 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
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	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	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	←	Is loaded with
INH	Inherent addressing	:	Concatenated with
IX	Indexed, no offset addressing	×	Multiplication
IX1	Indexed, 8-bit offset addressing	-( )	Negation (two's complement)
IX2	Indexed, 16-bit offset addressing	+	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 12-12. 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	A9	ii	2	‡	-	‡	‡	‡
			DIR	B9	dd	3					
			EXT	C9	hh ll	4					
			IX2	D9	ee ff	5					
			IX1	E9	ff	4					
			IX	F9		3					
ADD opr	Add without carry	$A \leftarrow (A) + (M)$	IMM	AB	ii	2	‡	-	‡	‡	‡
			DIR	BB	dd	3					
			EXT	CB	hh ll	4					
			IX2	DB	ee ff	5					
			IX1	EB	ff	4					
			IX	FB		3					
AND opr	Logical AND	$A \leftarrow (A) \cdot (M)$	IMM	A4	ii	2	-	-	‡	‡	-
			DIR	B4	dd	3					
			EXT	C4	hh ll	4					
			IX2	D4	ee ff	5					
			IX1	E4	ff	4					
			IX	F4		3					
ASL opr ASLA ASLX ASL opr ASL opr	Arithmetic shift left (Same as LSL)		DIR	38	dd	5	-	-	‡	‡	‡
			INH	48		3					
			INH	58		3					
			IX1	68	ff	6					
			IX	78		5					
ASR opr ASRA ASRX ASR opr ASR opr	Arithmetic shift right		DIR	37	dd	5	-	-	‡	‡	‡
			INH	47		3					
			INH	57		3					
			IX1	67	ff	6					
			IX	77		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)	11	dd	5	-	-	-	-	-
			DIR (b1)	13	dd	5					
			DIR (b2)	15	dd	5					
			DIR (b3)	17	dd	5					
			DIR (b4)	19	dd	5					
			DIR (b5)	1B	dd	5					
			DIR (b6)	1D	dd	5					
			DIR (b7)	1F	dd	5					
BCS rel	Branch if carry bit set (Same as BLO)	? 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 $\overline{IRQ}$ pin high	? $\overline{IRQ} = 1$	REL	2F	rr	3	-	-	-	-	
BIL rel	Branch if $\overline{IRQ}$ pin low	? $\overline{IRQ} = 0$	REL	2E	rr	3	-	-	-	-	
BIT rel	Bit test accumulator with memory	$(A) \cdot (M)$	IMM	A5	ii	2	-	-	‡	‡	-
			DIR	B5	dd	3					
			EXT	C5	hh ll	4					
			IX2	D5	ee ff	5					
			IX1	E5	ff	4					
			IX	F5		3					
BLO rel	Branch if lower (Same as BCS)	? 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 12-12. 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	Clear register	M ← \$00 A ← \$00 X ← \$00 M ← \$00 M ← \$00	DIR	3F	dd	5	-	-	0	1	-
CLRA			INH	4F		3	-	-	-	-	-
CLR X			INH	5F		3	-	-	-	-	-
CLR opr			IX1	6F	ff	6	-	-	-	-	-
CLR opr			IX	7F		5	-	-	-	-	-
CMP opr	Compare accumulator with memory	(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	Complement memory or register (one's 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
COMA			INH	43		3	-	-	-	-	-
COM X			INH	53		3	-	-	-	-	-
COM opr			IX1	63	ff	6	-	-	-	-	-
COM opr			IX	73		5	-	-	-	-	-
CPX opr	Compare index register with memory	(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	Decrement	M ← (M) - 1 A ← (A) - 1 X ← (X) - 1 M ← (M) - 1 M ← (M) - 1	DIR	3A	dd	5	-	-	‡	‡	-
DECA			INH	4A		3	-	-	-	-	-
DEC X			INH	5A		3	-	-	-	-	-
DEC opr			IX1	6A	ff	6	-	-	-	-	-
DEC opr			IX	7A		5	-	-	-	-	-

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

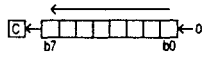
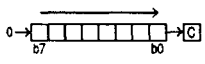
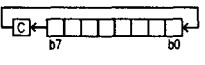

Source Form(s)	Operation	Description	Addressing Mode for Operand	Machine Coding (hexadecimal)		Cycles	Condition Code				
				Opcode	Operand		H	I	N	Z	C
EOR opr	Exclusive OR accumulator with memory	$A \leftarrow (A) \oplus (M)$	IMM DIR EXT IX2 IX1 IX	A8 B8 C8 D8 E8 F8	ii dd hh ll ee ff ff	2 3 4 5 4 3	-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

Table 12-12. 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	-	-	-	-	-
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	From Stack				
							↕	↕	↕	↕	↕
RTS	Return from subroutine	$SP \leftarrow (SP) + 1$ ; pull (PCH) $SP \leftarrow (SP) + 1$ ; pull (PCL)	INH	81		6	-	-	-	-	-
SBC opr	Subtract memory and carry bit from accumulator	$A \leftarrow (A) - (M) - C$	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 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 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 from accumulator	$A \leftarrow (A) - (M)$	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 to index register	$X \leftarrow (A)$	INH	97		2	-	-	-	-	-
TST opr TSTA TSTX TST opr TST opr	Test memory or register 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 to accumulator	$A \leftarrow (X)$	INH	9F		2	-	-	-	-	-
WAIT	Enable interrupts; halt CPU		INH	8F		2	-	0	-	-	-

# 12.4 Opcode Map

Table 12-13 is an opcode map of the M68HC05 instruction set.

Table 12-13. Opcode Map

Bit Manipulation		Branch		Read-Modify-Write			Control			Register/Memory					
DIR	DIR	REL	DIR	INH	INH	IX1	IX	INH	INH	IMM	DIR	EXT	IX2	IX1	IX
HI	0000														
LO	0001	2	0010	4	0100	5	0101	6	0110	7	0111	8	1000	9	1001
0	BRSET0	BRA	NEG	NEGA		NEG	NEG	RTI		SUB	SUB	SUB	SUB	SUB	SUB
1	BRCLR0	BRN	DIR			IX1	IX	INH		IMM	DIR	EXT	IX2	IX1	IX
0001	BRCLR0	BRN	REL					RTS		CMP	CMP	CMP	CMP	CMP	CMP
2	BRSET1	BHI						1		2	3	4	5	6	7
0010	BRSET1	BHI	REL	11						SBC	SBC	SBC	SBC	SBC	SBC
3	BRCLR1	BLS	COM	COMA		COM	COM	SWI		CPX	CPX	CPX	CPX	CPX	CPX
0011	BRCLR1	BLS	DIR	INH		IX1	IX	INH		IMM	DIR	EXT	IX2	IX1	IX
4	BRSET2	BCC	LSR	LSRA		LSR	LSR			AND	AND	AND	AND	AND	AND
0100	BRSET2	BCC	REL	INH		IX1	IX			IMM	DIR	EXT	IX2	IX1	IX
5	BRCLR2	BCS								BIT	BIT	BIT	BIT	BIT	BIT
0101	BRCLR2	BCS	REL							IMM	DIR	EXT	IX2	IX1	IX
6	BRSET3	BNE	ROR	RORA		ROR	ROR			LDA	LDA	LDA	LDA	LDA	LDA
0110	BRSET3	BNE	DIR	INH		IX1	IX			IMM	DIR	EXT	IX2	IX1	IX
7	BRCLR3	BEO	ASR	ASRA		ASR	ASR			STA	STA	STA	STA	STA	STA
0111	BRCLR3	BEO	REL	INH		IX1	IX	TAX		IMM	DIR	EXT	IX2	IX1	IX
8	BRSET4	BHCC	LSL/ASL	LSLA		LSL/ASL	LSL/ASL			EOR	EOR	EOR	EOR	EOR	EOR
1000	BRSET4	BHCC	DIR	INH		IX1	IX			IMM	DIR	EXT	IX2	IX1	IX
9	BRCLR4	BHCS	ROL	ROLA		ROL	ROL			ADC	ADC	ADC	ADC	ADC	ADC
1001	BRCLR4	BHCS	REL	INH		IX1	IX	SEC		IMM	DIR	EXT	IX2	IX1	IX
A	BRSET5	BPL	DEC	DECA		DEC	DEC			ORA	ORA	ORA	ORA	ORA	ORA
1010	BRSET5	BPL	DIR	INH		IX1	IX	CLI		IMM	DIR	EXT	IX2	IX1	IX
B	BRCLR5	BMI								ADD	ADD	ADD	ADD	ADD	ADD
1011	BRCLR5	BMI	REL					SEI		IMM	DIR	EXT	IX2	IX1	IX
C	BRSET6	BMC	INC	INCA		INC	INC			JMP	JMP	JMP	JMP	JMP	JMP
1100	BRSET6	BMC	DIR	INH		IX1	IX	RSP		IMM	DIR	EXT	IX2	IX1	IX
D	BRCLR6	BMS	TST	TSTA		TST	TST			BSR	BSR	BSR	BSR	BSR	BSR
1101	BRCLR6	BMS	REL	INH		IX1	IX	NOP		REL	DIR	EXT	IX2	IX1	IX
E	BRSET7	BIL								LDX	LDX	LDX	LDX	LDX	LDX
1110	BRSET7	BIL	REL					STOP		IMM	DIR	EXT	IX2	IX1	IX
F	BRCLR7	BIH	CLR	CLRA		CLR	CLR			STX	STX	STX	STX	STX	STX
1111	BRCLR7	BIH	DIR	INH		IX1	IX	TXA		IMM	DIR	EXT	IX2	IX1	IX
			REL	INH		IX1	IX	WAIT		IMM	DIR	EXT	IX2	IX1	IX

LEGEND

F	High Byte of Opcode in Hexadecimal
1111	High Byte of Opcode in Binary
0	Low Byte of Opcode in Hexadecimal
SUB	Low Byte of Opcode in Binary

ABBREVIATIONS FOR ADDRESSING MODES

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

	Number of Bytes/Addressing Mode	Number of Cycles	Opcode Mnemonic
1	1	3	SUB
2	2	3	IX
3	3	3	IX
4	4	3	IX
5	5	3	IX
6	6	3	IX
7	7	3	IX
8	8	3	IX
9	9	3	IX
A	A	3	IX
B	B	3	IX
C	C	3	IX
D	D	3	IX
E	E	3	IX
F	F	3	IX



## SECTION 13 ELECTRICAL SPECIFICATIONS

This section contains MCU electrical specifications and timing information.

### 13.1 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 13-1. 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 13-1. 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 MC68HC05P6P, DW, S (Standard) MC68HC05P6CP, CDW, CS (Extended) MC68HC05P6VP, VDW, VS (Automotive) MC68HC05P6MP, MDW, MS (Automotive)	$T_A$	0 to +70 -40 to +85 -40 to +105 -40 to +125	°C
Storage Temperature Range	$T_{STG}$	-65 to +150	°C

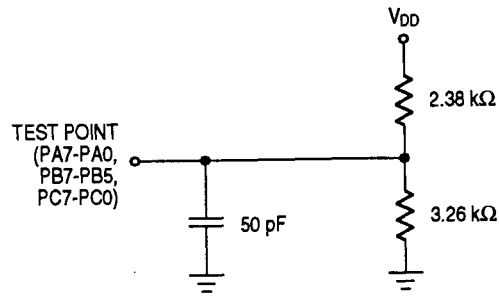
NOTES:

- |   |   |
|---|---|
| <ul style="list-style-type: none"> <li>1. P = Plastic dual in-line package (PDIP)</li> <li>2. DW = Small outline integrated circuit (SOIC)</li> <li>3. S = Ceramic dual in-line package (Cerdip)</li> </ul> | <ul style="list-style-type: none"> <li>4. C = Extended temperature range (-40 to +85 °C)</li> <li>5. V = Automotive temperature range (-40 to +105 °C)</li> <li>6. M = Automotive temperature range (-40 to +125 °C)</li> </ul> |
|---|---|

### 13.2 Thermal Characteristics

**Table 13-2. Thermal Resistance**

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic SOIC	$R_{\theta JA}$	60 60	°C/W



**Figure 13-1. Test Load**

### 13.3 Power Considerations

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

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

where:

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

$\theta_{JA}$  = Package thermal resistance, junction to ambient in  $^{\circ}\text{C}/\text{W}$

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

$P_{INT} = I_{CC} \times V_{CC}$ , watts — chip internal power

$P_{I/O}$  = Power dissipation on input and output pins — user-determined

For most applications  $P_{I/O} \ll P_{INT}$  and can be neglected.

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

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

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

$$K = P_A \times (T_A + 273 \text{ }^{\circ}\text{C}) + R_{\theta JA} \times P_D \quad (3)$$

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

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

**Table 13-3. 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 V
Output High Voltage ( $I_{LOAD} = -0.8 \text{ mA}$ ) PA7-PA0, PB7/SCK-PB5/SDO, PC7/ $V_{RH}$ -PC3/AN3, PC2-PC0, PD5, TCMF	$V_{OH}$	$V_{DD} - 0.8$	—	—	V
Output Low Voltage ( $I_{LOAD} = 1.6 \text{ mA}$ ) PA7-PA0, PB7/SCK-PB5/SDO, PC7/ $V_{RH}$ -PC3/AN3, PC2-PC0, PD5, TCMF	$V_{OL}$	—	—	0.4	V
Input High Voltage PA7-PA0, PB7/SCK-PB5/SDO, PC7/ $V_{RH}$ -PC3/AN3, PC2-PC0, PD5, PD7/TCAP, IRQ, RESET, OSC1	$V_{IH}$	$0.7 \times V_{DD}$	—	$V_{DD}$	V
Input Low Voltage PA7-PA0, PB7/SCK-PB5/SDO, PC7/ $V_{RH}$ -PC3/AN3, PC2-PC0, PD5, PD7/TCAP, IRQ, RESET, OSC1	$V_{IL}$	$V_{SS}$	—	$0.2 \times V_{DD}$	V
Supply Current (NOTES 3-6) RUN WAIT (A/D Converter On) WAIT (A/D Converter Off) STOP 25 °C 0 to 70 °C (Standard) -40 to 125 °C	$I_{DD}$	— — — — — — —	3.6 1.8 1.3 2 — — —	7.0 4.0 2.0 30 50 100	mA mA mA $\mu\text{A}$ $\mu\text{A}$ $\mu\text{A}$
I/O Ports Hi-Z Leakage Current PA7-PA0, PB7-PB5, PC7-PC0, PD5	$I_{IL}$	—	—	$\pm 10$	$\mu\text{A}$
A/D Ports Hi-Z Leakage Current	$I_{OZ}$	—	—	$\pm 1$	$\mu\text{A}$
Input Current RESET, IRQ, OSC1, PD7/TCAP	$I_{IN}$	—	—	$\pm 1$	$\mu\text{A}$
Capacitance Ports (As Input or Output) RESET, IRQ	$C_{OUT}$ $C_{IN}$	— —	— —	12 8	pF pF

**NOTES:**

- $V_{DD} = 5.0 \text{ Vdc} \pm 10\%$ ,  $V_{SS} = 0 \text{ Vdc}$ ,  $T_A = T_L$  to  $T_H$ , unless otherwise noted.
- Typical values at midpoint of voltage range, 25°C only.
- RUN (operating)  $I_{DD}$  and WAIT  $I_{DD}$  measured using external square wave clock source ( $f_{osc} = 4.2 \text{ MHz}$ ); all inputs 0.2 V from rail; no dc loads; less than 50 pF on all outputs;  $C_L = 20 \text{ pF}$  on OSC2.
- WAIT  $I_{DD}$  and STOP  $I_{DD}$ : all ports configured as inputs;  $V_{IL} = 0.2 \text{ V}$ ;  $V_{IH} = V_{DD} - 0.2 \text{ V}$ .
- STOP  $I_{DD}$  measured with  $OSC1 = V_{SS}$ .
- WAIT  $I_{DD}$  is affected linearly by the OSC2 capacitance.

## 13.5 DC Electrical Characteristics ( $V_{DD} = 3.3 \text{ Vdc}$ )

**Table 13-4. DC Electrical Characteristics ( $V_{DD} = 3.3 \text{ Vdc}$ )**

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage ( $I_{LOAD} \leq 10.0 \mu\text{A}$ )	$V_{OL}$ $V_{OH}$	— $V_{DD} - 0.1$	— —	0.1 —	V
Output High Voltage ( $I_{LOAD} = -0.2 \text{ mA}$ ) PA7-PA0, PB7/SCK-PB5/SDO, PC7/ $V_{RH}$ -PC3/AN3, PC2-PC0, PD5, TCMP	$V_{OH}$	$V_{DD} - 0.3$	—	—	V
Output Low Voltage ( $I_{LOAD} = 0.4 \text{ mA}$ ) PA7-PA0, PB7/SCK-PB5/SDO, PC7/ $V_{RH}$ -PC3/AN3, PC2-PC0, PD5, TCMP	$V_{OL}$	—	—	0.3	V
Input High Voltage PA7-PA0, PB7/SCK-PB5/SDO, PC7/ $V_{RH}$ -PC3/AN3, PC2-PC0, PD5, PD7/TCAP, IRQ, RESET, OSC1	$V_{IH}$	$0.7 \times V_{DD}$	—	$V_{DD}$	V
Input Low Voltage PA7-PA0, PB7/SCK-PB5/SDO, PC7/ $V_{RH}$ -PC3/AN3, PC2-PC0, PD5, PD7/TCAP, IRQ, RESET, OSC1	$V_{IL}$	$V_{SS}$	—	$0.2 \times V_{DD}$	V
Data-Retention Mode Supply Voltage	$V_{RM}$	2.0	—	—	V
Supply Current (NOTES 3-6)	$I_{DD}$				
RUN		—	1.6	2.5	mA
WAIT (A/D Converter On)		—	0.9	1.4	mA
WAIT (A/D Converter Off)		—	0.4	1.0	mA
STOP		—	—	—	—
25°C		—	1.0	20	$\mu\text{A}$
0 to 70°C (Standard)		—	—	40	$\mu\text{A}$
-40 to +125°C		—	—	50	$\mu\text{A}$
I/O Ports Hi-Z Leakage Current PA7-PA0, PB7/SCK-PB5/SDO, PC7/ $V_{RH}$ -PC3/AN3, PC2-PC0, PD5	$I_{IL}$	—	—	$\pm 10$	$\mu\text{A}$
Input Current RESET, IRQ, OSC1, PD5, PD7/TCAP	$I_{IN}$	—	—	$\pm 1$	$\mu\text{A}$
Capacitance					
Ports (As Input or Output)	$C_{OUT}$	—	—	12	pF
RESET, IRQ, PD5, PD7/TCAP	$C_{IN}$	—	—	8	pF

**NOTES:**

- $V_{DD} = 3.3 \text{ Vdc} \pm 10\%$ ,  $V_{SS} = 0 \text{ Vdc}$ ,  $T_A = T_L$  to  $T_H$  unless otherwise noted.
- Typical values at midpoint of voltage range, 25°C only.
- RUN (operating)  $I_{DD}$  and WAIT  $I_{DD}$  measured using external square wave clock source ( $f_{OSC} = 2.1 \text{ MHz}$ ); all inputs 0.2 V from rail; no dc loads; less than 50 pF on all outputs;  $C_L = 20 \text{ pF}$  on OSC2.
- WAIT  $I_{DD}$  and STOP  $I_{DD}$ : all ports configured as inputs;  $V_{IL} = 0.2 \text{ V}$ ,  $V_{IH} = V_{DD} - 0.2 \text{ V}$ .
- STOP  $I_{DD}$  measured with  $OSC1 = V_{SS}$ .
- WAIT  $I_{DD}$  is affected linearly by the OSC2 capacitance.

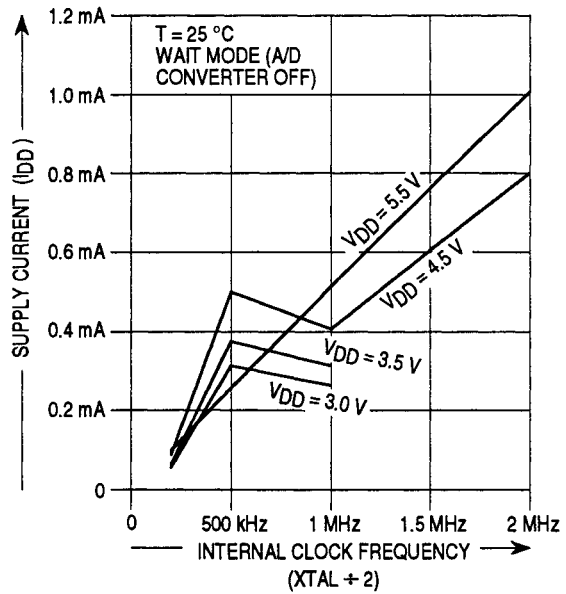
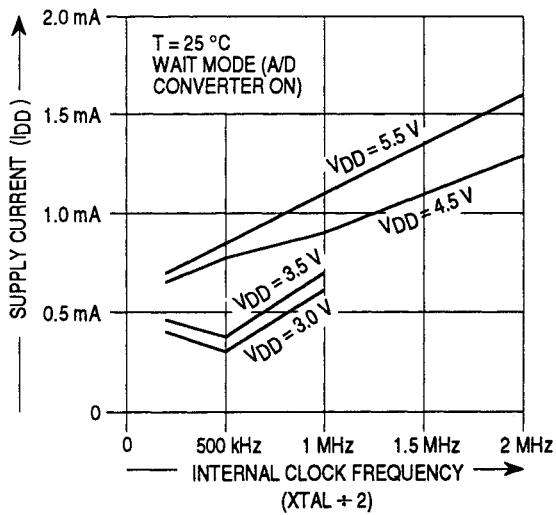
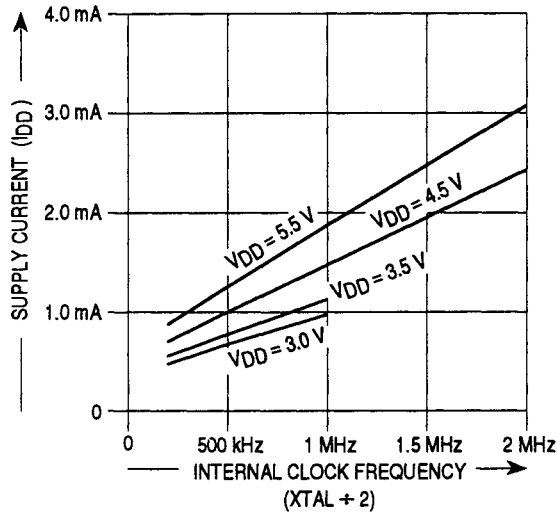


Figure 13-2.  $I_{DD}$  vs Internal Clock Frequency ( $T = 25\text{ }^{\circ}\text{C}$ )

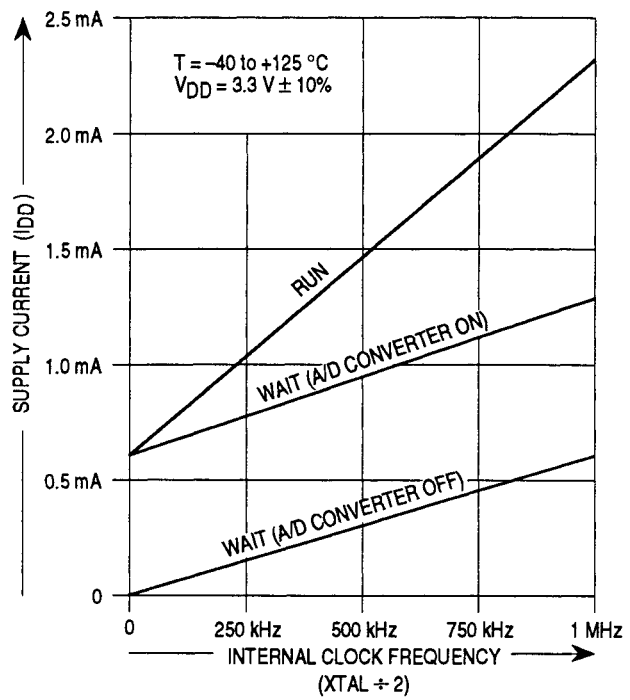
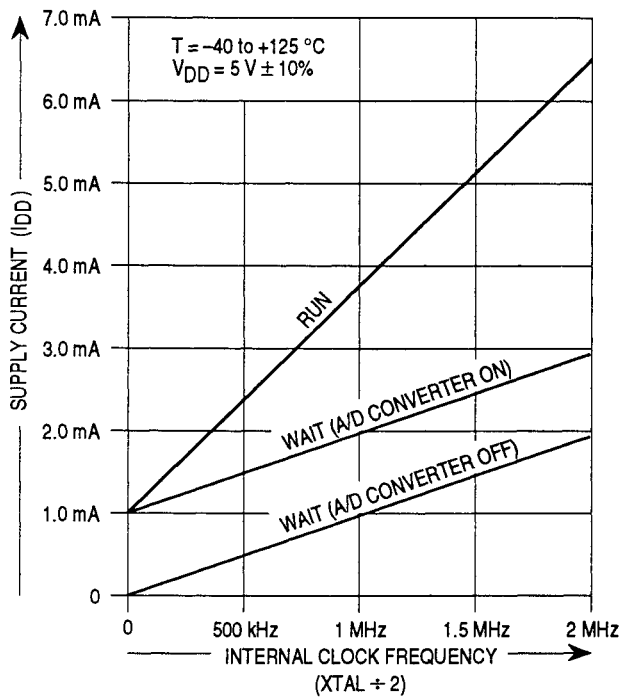


Figure 13-3.  $I_{DD}$  vs Internal Clock Frequency ( $T = -40 \text{ }^\circ\text{C}$  to  $+125 \text{ }^\circ\text{C}$ )

## 13.6 A/D Converter Characteristics

**Table 13-5. A/D Converter Characteristics**

Characteristic	Min	Max	Unit
Resolution	8	8	Bit
Absolute Accuracy ( $4.0 > V_{RH} > V_{DD}$ ) (NOTE 2)	—	$\pm 1-1/2$	LSB
Conversion Range ( $V_{RH}$ Pin)	$V_{SS}$	$V_{DD}$	V
Conversion Time (Includes Sampling Time)			
External Clock (XTAL)	32	32	$t_{AD}$
Internal RC Oscillator (ADRC = 1)	32	32	$\mu s$
Monotonicity	Inherent (Within Total Error)		
Zero Input Reading ( $V_{IN} = 0$ V)	00	01	Hex
Full-Scale Reading ( $V_{IN} = V_{RH}$ )	FF	FF	Hex
Sample Acquisition Time (NOTE 3)			
External Clock (XTAL)	12	12	$t_{AD}$
Internal RC Oscillator (ADRC) = 1	—	12	$\mu s$
Input Capacitance PC6/AN0, PC5/AN1, PC4/AN2, PC3/AN3	—	12	pF
Analog Input Voltage	$V_{SS}$	$V_{RH}$	V
Input Leakage (NOTE 5)			
PC6/AN0, PC5/AN1, PC4/AN2, PC3/AN3	—	$\pm 1$	$\mu A$
$V_{RH}$	—	$\pm 1$	$\mu A$

**NOTES:**

1.  $V_{DD} = 5.0$  Vdc  $\pm 10\%$ ,  $V_{SS} = 0$  Vdc,  $T_A = T_L$  to  $T_H$ , unless otherwise noted.
2. A/D accuracy may decrease proportionately as  $V_{RH}$  is reduced below 4.0 V.
3. Source impedances greater than 10 kohm adversely affect internal RC charging time during input sampling.
4.  $t_{AD} = t_{CYC}$  if clock source is MCU.
5. The external system error caused by input leakage current is approximately equal to the product of R source and input current.

## 13.7 Control Timing ( $V_{DD} = 5.0 \text{ Vdc}$ )

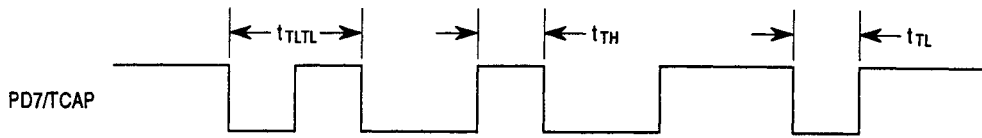
**Table 13-6. Control Timing ( $V_{DD} = 5.0 \text{ Vdc}$ )**

Characteristic	Symbol	Min	Max	Unit
Oscillator Frequency Crystal Option	fosc	—	4.2	MHz
External Clock Option		dc	4.2	MHz
Internal Operating Frequency Crystal (fosc + 2) External Clock (fosc + 2)	fOP	—	2.1	MHz
		dc	2.1	MHz
Cycle Time	tCYC	480	—	ns
Crystal Oscillator Startup Time	tOXOV	—	100	ms
STOP Recovery Startup Time (Crystal Oscillator)	tILCH	—	100	ms
RESET Pulse Width	tRL	1.5	—	tCYC
Timer Resolution (NOTE 2)	tRESL	4.0	—	tCYC
Interrupt Pulse Width Low (Edge-Triggered)	tILIH	125	—	ns
Interrupt Pulse Period	tILIL	(NOTE 3)	—	tCYC
OSC1 Pulse Width	tOH, tOL	90	—	ns
RC Oscillator Stabilization Time	tRCON	—	5	$\mu\text{s}$
A/D On Current Stabilization Time	tADON	—	100	$\mu\text{s}$

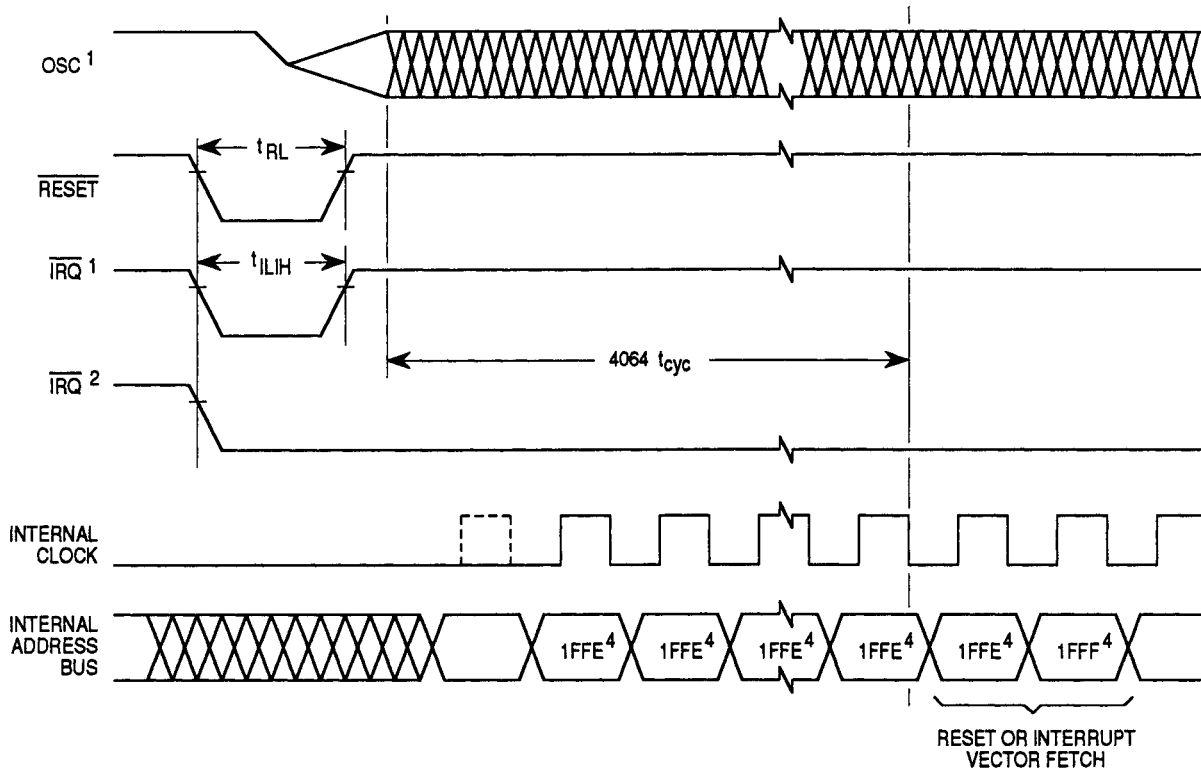
**NOTES:**

1.  $V_{DD} = 5.0 \text{ Vdc} \pm 10\%$ ,  $V_{SS} = 0 \text{ Vdc}$ ,  $T_A = T_L$  to  $T_H$ , unless otherwise noted.
2. Since a 2-bit prescaler in the timer must count four internal cycles (tCYC), this is the limiting minimum factor in determining the timer resolution.
3. The minimum period tILIL should not be less than the number of cycle times it takes to execute the interrupt service routine plus 19 tCYC.





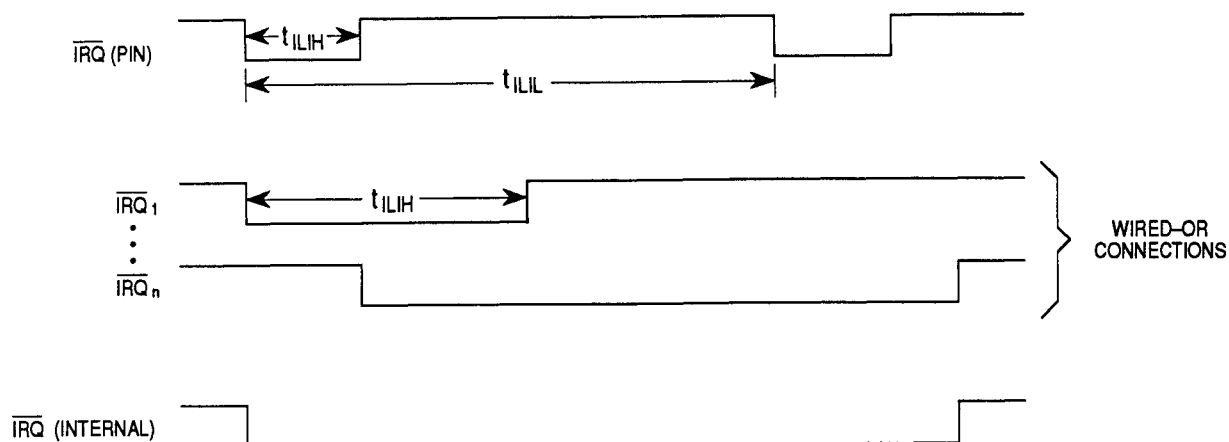
**Figure 13-4. TCAP Timing**



**NOTES:**

1. Represents the internal clocking of OSC1 pin.
2. External interrupt edge-triggered mask option.
3. External interrupt edge- and level-triggered mask option.
4. Reset vector shown for timing example.

**Figure 13-5. STOP Recovery Timing**



**Figure 13-6. External Interrupt Timing**

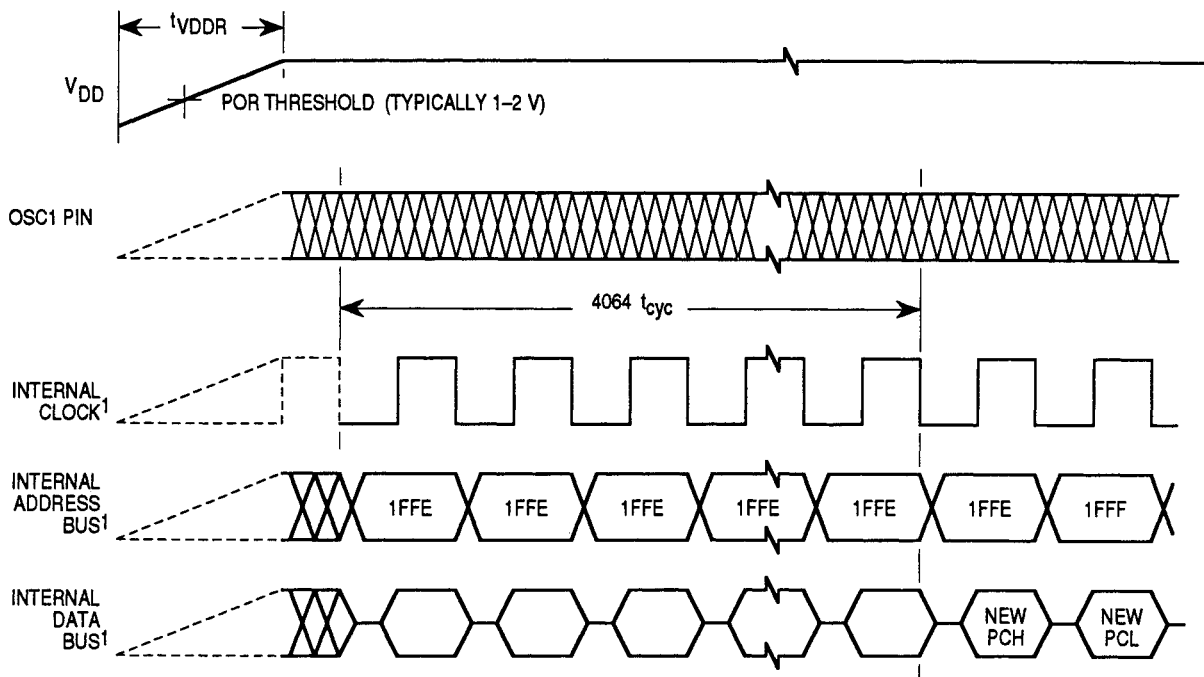
**13.8 Control Timing ( $V_{DD} = 3.3 \text{ Vdc}$ )**

**Table 13-7. Control Timing ( $V_{DD} = 3.3 \text{ Vdc}$ )**

Characteristic	Symbol	Min	Max	Unit
Oscillator Frequency Crystal Option External Clock Option	$f_{osc}$	– dc	2.0 2.0	MHz MHz
Internal Operating Frequency Crystal ( $f_{osc} + 2$ ) External Clock ( $f_{osc} + 2$ )	$f_{op}$	– dc	1.0 1.0	MHz MHz
Cycle Time	$t_{CYC}$	1	–	ms
Crystal Oscillator Startup Time	$t_{OXOV}$	–	100	ms
STOP Recovery Startup Time (Crystal Oscillator)	$t_{ILCH}$	–	100	ms
$\overline{\text{RESET}}$ Pulse Width	$t_{RL}$	1.5	–	$t_{CYC}$
Timer Resolution (NOTE 2)	$t_{RESL}$	4.0	–	$t_{CYC}$
Interrupt Pulse Width Low (Edge-Triggered)	$t_{ILIH}$	250	–	ns
Interrupt Pulse Period	$t_{ILIL}$	(NOTE 3)	–	$t_{CYC}$
OSC1 Pulse Width	$t_{OH}, t_{OL}$	200	–	ns

NOTES:

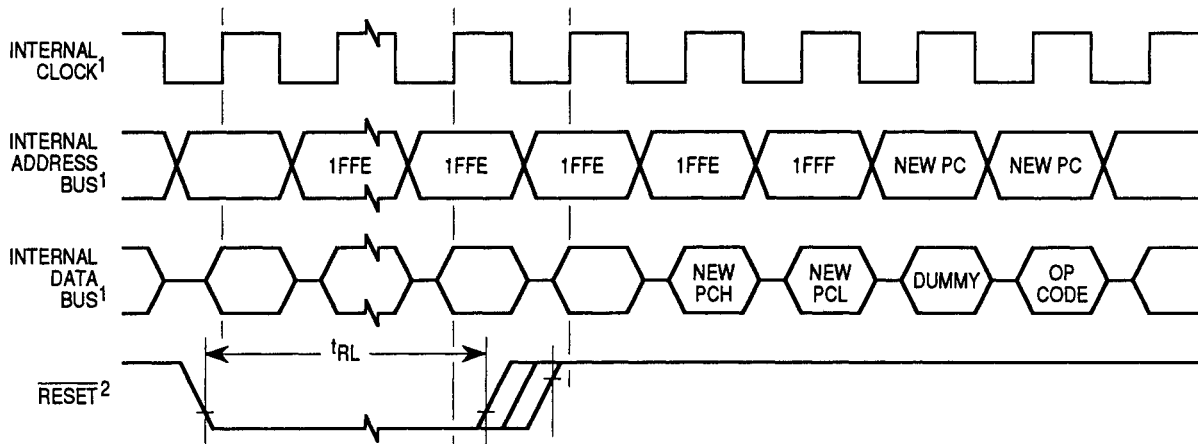
- $V_{DD} = 3.3 \text{ Vdc} \pm 10\%$ ,  $V_{SS} = 0 \text{ Vdc}$ ,  $T_A = T_L$  to  $T_H$ , unless otherwise noted.
- Because a 2-bit prescaler in the timer must count four internal cycles ( $t_{CYC}$ ), this is the limiting minimum factor in determining the timer resolution.
- The minimum period  $t_{ILIL}$  should not be less than the number of cycle times it takes to execute the interrupt service routine plus 19  $t_{CYC}$ .



**NOTES:**

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

**Figure 13-7. 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  $\overline{RESET}$  initiates reset sequence.

**Figure 13-8. External Reset Timing**

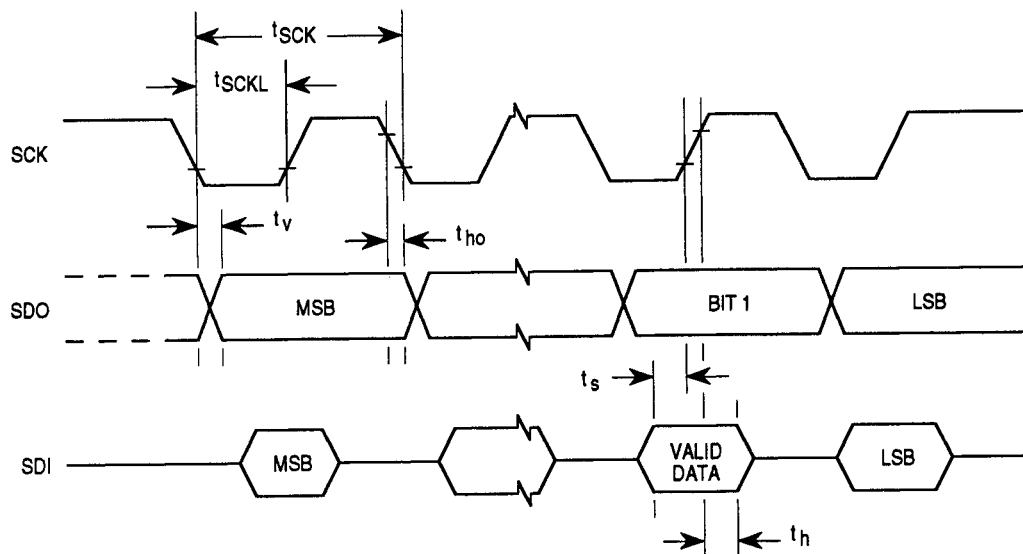
## 13.9 SIOP Timing ( $V_{DD} = 5.0 \text{ Vdc}$ )

**Table 13-8. SIOP Timing ( $V_{DD} = 5.0 \text{ Vdc}$ )**

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation Master Slave	$f_{SIOP(M)}$ $f_{SIOP(S)}$	0.25 dc	0.25 525	$f_{OP}$ kHz
Cycle time Master Slave	$t_{SCK(M)}$ $t_{SCK(S)}$	4.0 —	4.0 1920	$t_{CYC}$ ns
Clock (SCK) Low Time ( $f_{OP} = 2.1 \text{ MHz}$ )	$t_{SCKL}$	932	—	ns
SDO Data Valid Time	$t_V$	—	200	ns
SDO Hold Time	$t_{HO}$	0	—	ns
SDI Setup Time	$t_S$	100	—	ns
SDI Hold Time	$t_H$	100	—	ns

**NOTES:**

- $V_{DD} = 5.0 \text{ Vdc} \pm 10\%$ ,  $V_{SS} = 0 \text{ Vdc}$ ,  $T_A = T_L$  to  $T_H$ , unless otherwise noted.
- $f_{OP} = f_{OSC} \div 2 = 2.1 \text{ MHz}$  maximum;  $t_{CYC} = 1 \div f_{OP}$ .
- In master mode, SCK is generated by dividing the internal clock ( $f_{OP}$ ) by 4.



**NOTES:**

- This diagram applies to both the master and slave modes of the SIOP.
- Bit order is shown for MSB first option.

**Figure 13-9. SIOP Timing**

## 13.10 SIOP Timing ( $V_{DD} = 3.3$ Vdc)

**Table 13-9. SIOP Timing ( $V_{DD} = 3.3$  Vdc)**

Characteristic	Symbol	Min	Max	Unit
Frequency of operation Master Slave	$f_{SIOP(M)}$ $f_{SIOP(S)}$	0.25 dc	0.25 250	$f_{OP}$ kHz
Cycle time Master Slave	$t_{SCK(M)}$ $t_{SCK(S)}$	4.0 –	4.0 4000	$t_{CYC}$ ns
Clock (SCK) Low Time ( $f_{OP} = 1.0$ MHz)	$t_{SCKL}$	1980	–	ns
SDO Data Valid Time	$t_V$	–	400	ns
SDO Hold Time	$t_{HO}$	0	–	ns
SDI Setup Time	$t_S$	200	–	ns
SDI Hold Time	$t_H$	200	–	ns

**NOTES:**

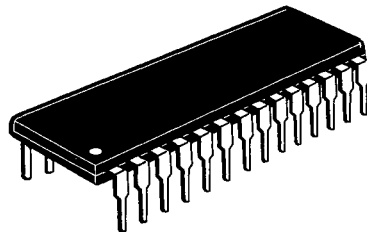
- $V_{DD} = 3.3$  Vdc  $\pm$  10%,  $V_{SS} = 0$  Vdc,  $T_A = T_L$  to  $T_H$ , unless otherwise noted.
- $f_{OP} = 1.0$  MHz maximum.



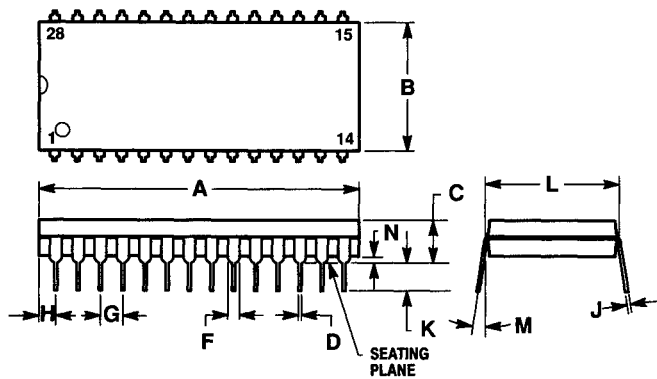
## SECTION 14 MECHANICAL SPECIFICATIONS

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

### 14.1 PDIP



SCALE 1:1



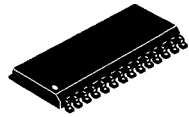
**NOTES:**

1. POSITIONAL TOLERANCE OF LEADS (D), SHALL BE WITHIN 0.25mm (0.010) AT MAXIMUM MATERIAL CONDITION, IN RELATION TO SEATING PLANE AND EACH OTHER.
2. DIMENSION L TO CENTER OF LEADS WHEN FORMED PARALLEL.
3. DIMENSION B DOES NOT INCLUDE MOLD FLASH.
4. 710-01 OBSOLETE, NEW STANDARD 710-02.

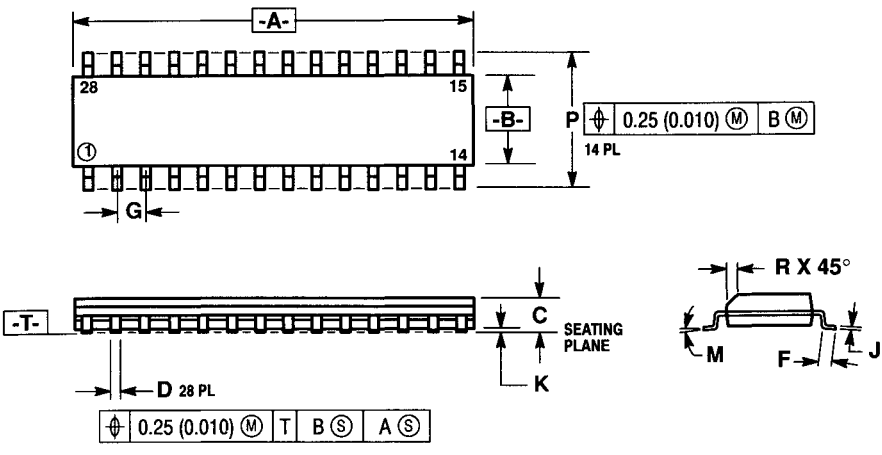
DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	36.45	37.21	1.435	1.465
B	13.72	14.22	0.540	0.560
C	3.94	5.08	0.155	0.200
D	0.36	0.56	0.014	0.022
F	1.02	1.52	0.040	0.060
G	2.54 BSC		0.100 BSC	
H	1.65	2.16	0.065	0.085
J	0.20	0.38	0.008	0.015
K	2.92	3.43	0.115	0.135
L	15.24 BSC		0.600 BSC	
M	0°	15°	0°	15°
N	0.51	1.02	0.020	0.040

**Figure 14-1. Case #710-02**

# 14.2 SOIC



SCALE 1:1



- NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
  2. CONTROLLING DIMENSION: MILLIMETER.
  3. DIMENSION A AND B DO NOT INCLUDE MOLD PROTRUSION.
  4. MAXIMUM MOLD PROTRUSION 0.15 (0.006) PER SIDE.
  5. 751F-01 AND -02 OBSOLETE, NEW STANDARD 751F-03.

DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	17.80	18.05	0.701	0.711
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	0.41	0.90	0.016	0.035
G	1.27 BSC		0.050 BSC	
J	0.229	0.317	0.0090	0.0125
K	0.127	0.292	0.0050	0.0115
M	0°	8°	0°	8°
P	10.05	10.55	0.395	0.415
R	0.25	0.75	0.010	0.029

Figure 14-2. Case #751F-03



## SECTION 15 ORDERING INFORMATION

This section contains instructions for ordering custom-masked ROM MCUs.

### 15.1 MCU Ordering Forms

To initiate an order for a ROM-based MCU, first obtain the current ordering form for the MCU from a Motorola representative. Submit the following items when ordering MCUs:

- A current MCU ordering form that is **completely filled out** (Contact your Motorola sales office for assistance.)
- A signed XC status letter, if applicable (See **15.5 XC Status Letter.**)
- A copy of the customer specification if the customer specification deviates from the Motorola specification for the MCU
- Customer's application program on one of the media listed in **15.2 Application Program Media**

The current MCU ordering form is also available through the Motorola Freeware Bulletin Board Service (BBS). The telephone number is (512) 891-FREE. After making the connection, type bbs in lowercase letters and press the return key to start the BBS software.

## 15.2 Application Program Media

Please deliver the application program to Motorola in one of the following media:

- Macintosh®<sup>1</sup> 3-1/2-inch diskette (double-sided 800K or double-sided high-density 1.4M)
- MS-DOS®<sup>2</sup> or PC-DOS®<sup>3</sup> 3-1/2-inch diskette (double-sided 720K or double-sided high-density 1.44M)
- MS-DOS® or PC-DOS® 5-1/4-inch diskette (double-sided double-density 360K or double-sided high-density 1.2M)
- EPROM(s) 2716, 2732, 2764, 27128, 27256, or 27512 (depending on the size of the memory map of the MCU)

Use positive logic for data and addresses.

### 15.2.1 Diskettes

If submitting the application program on a diskette, clearly label the diskette with the following information:

- Customer name
- Customer part number
- Project or product name
- Filename of object code
- Date
- Name of operating system that formatted diskette
- Formatted capacity of diskette

On diskettes, the application program must be in Motorola's S-record format (S1 and S9 records), a character-based object file format generated by M6805 cross assemblers and linkers.

---

<sup>1</sup> Macintosh is a registered trademark of Apple Computer, Inc.

<sup>2</sup> MS-DOS is a registered trademark of Microsoft, Inc.

<sup>3</sup> PC-DOS is a registered trademark of International Business Machines Corporation.

## NOTE

Begin the application program at the first user ROM location. Program addresses must correspond exactly to the available on-chip user ROM addresses as shown in the memory map. **Write \$00 in all non-user ROM locations or leave all non-user ROM locations blank.** Refer to the current MCU ordering form for additional requirements.

If the memory map has two user ROM areas with the same addresses, then write the two areas in separate files on the diskette. Label the diskette with both filenames.

In addition to the object code, a file containing the source code can be included. Motorola keeps this code confidential and uses it only to expedite ROM pattern generation in case of any difficulty with the object code. Label the diskette with the filename of the source code.

### 15.2.2 EPROMs

If submitting the application program in an EPROM, clearly label the EPROM with the following information:

- Customer name
- Customer part number
- Checksum
- Project or product name
- Date

## NOTE

Begin the application program at the first user ROM location. Program addresses must correspond exactly to the available on-chip user ROM addresses as shown in the memory map. **Write \$00 in all non-user ROM locations.** Refer to the current MCU ordering form for additional requirements.

Submit the application program in one EPROM large enough to contain the entire memory map. If the memory map has two user ROM areas with the same addresses, then write the two areas on separate EPROMs. Label the EPROMs with the addresses they contain.

Pack EPROMs securely in a conductive IC carrier for shipment. Do not use Styrofoam.

### **15.3 ROM Program Verification**

The primary use for the on-chip ROM is to contain the customer's application program. The customer develops and debugs the application program and then submits the MCU order along with his application program.

Motorola inputs the customer's application program code into a computer program that generates a listing verify file. The listing verify file represents the memory map of the MCU. The listing verify file contains the user ROM code and may also contain non-user ROM code, such as self-check code. Motorola sends the customer a computer printout of the listing verify file along with a listing verify form.

To aid the customer in checking the listing verify file, Motorola will program the listing verify file into customer-supplied blank EPROMs or preformatted Macintosh or DOS disks. All original pattern media are filed for contractual purposes and are not returned.

Check the listing verify file thoroughly, then complete and sign the listing verify form and return the listing verify form to Motorola. The signed listing verify form constitutes the contractual agreement for the creation of the custom mask.

## **15.4 ROM Verification Units (RVUs)**

After receiving the signed listing verify form, Motorola manufactures a custom photographic mask. The mask contains the customer's application program and is used to process silicon wafers. The application program cannot be changed after the manufacture of the mask begins. Motorola then produces ten MCUs, called RVUs, and sends the RVUs to the customer. RVUs are usually packaged in unmarked ceramic and tested to 5 Vdc at room temperature. RVUs are not tested to environmental extremes because their sole purpose is to demonstrate that the customer's user ROM pattern was properly implemented. The ten RVUs are free of charge with the minimum order quantity but are not production parts. RVUs are not guaranteed by Motorola Quality Assurance.

## **15.5 XC Status Letter**

The XC status letter is for customer acknowledgement that the MCU is a pilot production device. As a pilot production device, the MCU part number has an XC prefix. When the MCU meets all of Motorola's formal quality and reliability requirements, the XC prefix is replaced with MC. The MCU ordering form indicates whether or not the MCU order requires an XC status letter.



Date:

To:

Subject: XC Status of Device

This letter requests formal authorization from \_\_\_\_\_  
for Motorola Microcontroller Division to ship XC \_\_\_\_\_  
devices as pilot production MCUs.

The XC prefix indicates that the MCU has yet to meet Motorola's formal quality and reliability requirements and is still in the pilot production phase. The pilot production phase lasts approximately six months as the necessary qualification and stress tests are completed to bring the MCU to fully qualified MC status. The manufacture of XC MCUs is similar to the manufacture of standard production MCUs and includes the following:

- Processing per production shop order
- 100% testing per current data sheet
- Standard QA inspection and tests
- Complete traceability
- Preliminary reliability testing

This letter requests that a representative of \_\_\_\_\_  
acknowledge by signing below that he understands the XC pilot production  
status and that he will receive XC \_\_\_\_\_  
pilot production MCUs.

\_\_\_\_\_  
John H. Sayce  
Reliability and Quality Assurance Manager  
Advanced Microcontroller Division/CSIC Microcontroller Division

Customer representative: Please sign below and return this letter to the  
following address:

Motorola CSIC Microcontroller Division  
6501 William Cannon Drive West  
Austin, TX 78735  
Mail Drop OE 39

\_\_\_\_\_  
Customer Representative

\_\_\_\_\_  
Title

\_\_\_\_\_  
Date











**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**



MC68HC05P6/D



hit C

# 68HC05P6 : Microcontroller

The Motorola MC68HC05P6 microcontroller is a low-cost combination of an M68HC05 Family microprocessor with a 4-channel, 8-bit analog-to-digital (A/D) converter, a 16-bit timer with output compare and input capture, a serial communications port (SIOP), and a computer operating properly (COP) watchdog timer. The HC05 CPU core contains 176 bytes of RAM, 4k bytes of user ROM, and 21 input/output (I/O) pins (20 bidirectional, 1 input-only). This device is available in a 28-pin SOIC, or PDIP.

## 68HC05P6 Features

- Low Cost HC05 Core
- 28-Pin SOIC, or PDIP
- 4672 Bytes of User EPROM (Including 48 Bytes of Page Zero ROM and 16 Bytes of User Vectors)
- 176 Bytes of On-Chip RAM
- Self Check ROM
- 4-Channel 8-Bit A/D Converter
- SIOP Serial Communications Port
- 16-Bit Timer with Output Compare and Input Capture
- 20 Bidirectional I/O Lines and 1 Input-Only line
- On-chip oscillator with crystal or ceramic resonator connections or resistor-capacitor connections

[\[top\]](#)


## 68HC05P6 Parametrics

ROM (KBytes)	RAM (Bytes)	Timer	I/O Serial	A/D	Operating Voltage (V)	Bus Frequency (Max) (MHz)	Availability
4.5	176	16-Bit 1 I/C 1 O/C	21 SIOP	4-CH 8-Bit	3.3, 5.0	2.1	Now

[\[top\]](#)

## 68HC05P6 Documentation



Page Contents
<ul style="list-style-type: none"> <li>• <a href="#">Features</a></li> <li>• <a href="#">Parametrics</a></li> <li>• <a href="#">Documentation</a></li> <li>• <a href="#">Development Tools/Boards</a></li> <li>• <a href="#">Design Tools</a></li> <li>• <a href="#">Orderable Parts</a> </li> </ul>
Other Info
<ul style="list-style-type: none"> <li>• <a href="#">FAQs</a></li> <li>• <a href="#">Literature Services</a></li> <li>• <a href="#">Acceleration, Pressure, Alarm IC, and Smoke IC Sensors</a></li> <li>• <a href="#">Automotive</a></li> <li>• <a href="#">Consumer &amp; Industrial</a></li> <li>• <a href="#">Microcontrollers</a></li> <li>• <a href="#">Motor Control</a></li> <li>• <a href="#">3rd Party Design Help</a></li> </ul>

**Application Note**

ID	Name	Format	Size K	Rev #	Date Last Modified	Order Availability
<a href="#">AN-HK-22/D</a>	MC68HC05SR3 and MC68HC705SR3 Design Notes	pdf	3241	0	1/01/1994	-
<a href="#">AN-HK-23/D</a>	MC6805R3 and MC68HC05SR3 Technical Comparison	pdf	544	0	1/01/1994	-
<a href="#">AN1050/D</a>	Designing for Electromagnetic Compatibility (EMC) with HCMOS Microcontrollers	pdf	82	0	1/01/2000	<input type="checkbox"/>
<a href="#">AN1055/D</a>	M6805 16-Bit Support Macros	pdf	1048	0	1/01/1990	<input type="checkbox"/>
<a href="#">AN1067/D</a>	Pulse Generation and Detection with Microcontroller Units	pdf	242	1	5/31/2002	<input type="checkbox"/>
<a href="#">AN1222/D</a>	Arithmetic Waveform Synthesis with the HC05/08 MCUs	pdf	24	0	1/01/1993	<input type="checkbox"/>
<a href="#">AN1222SW</a>	Software Files for AN1222 zipped	zip	20	0	1/01/1995	-
<a href="#">AN1259/D</a>	System Design and Layout Techniques for Noise Reduction in MCU-Based Systems	pdf	78	0	1/01/1995	<input type="checkbox"/>
<a href="#">AN1262/D</a>	Simple Real-Time Kernels for M68HC05 Microcontrollers	pdf	84	0	1/01/1995	<input type="checkbox"/>
<a href="#">AN1262SW</a>	Software files for AN1262	zip	11	0	1/01/1995	-
<a href="#">AN1263/D</a>	Designing for Electromagnetic Compatibility with Single-Chip Microcontrollers	pdf	104	0	1/01/1995	<input type="checkbox"/>
<a href="#">AN1292/D</a>	Adding a Voice User Interface to M68HC05 Applications	pdf	155	0	1/01/1996	<input type="checkbox"/>
<a href="#">AN1292SW</a>	Software files for AN1292 zipped	zip	215	0	1/01/1996	-
<a href="#">AN1667/D</a>	Software SCI Implementation to the MISC Communication Protocol	pdf	112	0	7/10/2002	<input type="checkbox"/>
<a href="#">AN1667SW</a>	Software for AN1667, zip format	zip	93	1.0	7/31/2002	-
<a href="#">AN1688/D</a>	MISC Bus Slave Switch Node	pdf	243	0	7/11/2002	<input type="checkbox"/>
<a href="#">AN1705/D</a>	Noise Reduction Techniques for Microcontroller-Based Systems	pdf	67	0	1/01/1999	<input type="checkbox"/>
<a href="#">AN1723/D</a>	Interfacing MC68HC05 Microcontrollers to the IBM AT Keyboard Interface	pdf	274	0	1/01/1997	<input type="checkbox"/>
<a href="#">AN1733/D</a>	Implementing Caller ID Functionality in MC68HC(7)05 Applications	pdf	312	0	1/01/1998	<input type="checkbox"/>
<a href="#">AN1734/D</a>	Pulse Width Modulation Using the 16-Bit Timer	pdf	102	0	1/01/1998	<input type="checkbox"/>
<a href="#">AN1734SW</a>	Software files for AN1734 zipped	zip	2	0	1/01/1997	-
<a href="#">AN1736/D</a>	Variations in the Motorola MC68HC05Px Family	pdf	203	1	1/01/1998	<input type="checkbox"/>
<a href="#">AN1744/D</a>	Resetting Microcontrollers During Power Transitions	pdf	80	0	1/01/1998	<input type="checkbox"/>
<a href="#">AN1752/D</a>	Data Structures for 8-Bit Microcontrollers	pdf	213	1	5/07/2001	<input type="checkbox"/>
<a href="#">AN1757/D</a>	Add a Unique Silicon Serial Number to the HC05	pdf	105	0	1/01/1998	<input type="checkbox"/>
<a href="#">AN1758/D</a>	Add Addressable Switches to the HC05	pdf	111	0	1/01/1998	<input type="checkbox"/>
<a href="#">AN1771/D</a>	Precision Sine-Wave Tone Synthesis Using 8-Bit MCUs	pdf	250	0	1/01/1998	<input type="checkbox"/>
<a href="#">AN1775/D</a>	Expanding Digital Input with an A/D Converter	pdf	86	1	1/01/1998	<input type="checkbox"/>

<a href="#">AN1818/D</a>	Software SCI Routines with the 16-Bit Timer Module	pdf	84	0	1/01/1999	<input type="checkbox"/>
<a href="#">AN1820/D</a>	Software I2C Communications	pdf	55	0	1/01/1999	<input type="checkbox"/>
<a href="#">AN1820SW</a>	Software files for AN1820 zipped	zip	2	0	1/01/1998	-
<a href="#">AN2103/D</a>	Local Interconnect Network (LIN) Demonstration	pdf	953	0	12/01/2000	<input type="checkbox"/>
<a href="#">AN2159/D</a>	Digital Direct Current Ignition System Using HC08 Microcontrollers	pdf	129	0	11/20/2001	<input type="checkbox"/>
<a href="#">AN2159SW</a>	AN2159SW	zip	182	1	3/08/2002	-
<a href="#">AN2321/D</a>	Designing for Board Level Electromagnetic Compatibility	pdf	1628	0	8/15/2002	<input type="checkbox"/>
<a href="#">AN4006/D</a>	Digital Captive Discharge Ignition System Using HC05/HC08 8-Bit Microcontrollers	pdf	61	0	3/27/2000	<input type="checkbox"/>
<a href="#">AN442/D</a>	Driving LCDs with M6805 Microprocessors	pdf	1134	0	1/01/1991	<input type="checkbox"/>
<a href="#">AN463/D</a>	68HC05K0 Infra-red Remote Control	pdf	111	0	1/01/1992	<input type="checkbox"/>
<a href="#">AN464/D</a>	Software Driver Routines for the Motorola MC68HC05 CAN Module	pdf	2859	0	1/01/1993	<input type="checkbox"/>
<a href="#">AN477/D</a>	Simple A/D for MCUs without Built-In A/D Converters	pdf	224	0	1/01/1993	<input type="checkbox"/>
<a href="#">AN991/D</a>	Using the Serial Peripheral Interface to Communicate Between Multiple Microcomputers	pdf	251	1	1/28/2002	<input type="checkbox"/>
<a href="#">ANE416/D</a>	MC68HC05B4 Radio Synthesizer	pdf	1958	0	1/01/1988	<input type="checkbox"/>

## Brochure

ID	Name	Format	Size K	Rev #	Date Last Modified	Order Availability
<a href="#">8-16BITPAK/D</a>	8-16 Bit Microcontrollers Product Portfolio	html	1	0	10/15/2002	<input type="checkbox"/>
<a href="#">FLYREMBEDFLASH/D</a>	Embedded Flash: Changing the Technology World for the Better	pdf	621	1	4/03/2002	<input type="checkbox"/>

## Data Sheets

ID	Name	Format	Size K	Rev #	Date Last Modified	Order Availability
<a href="#">MC68HC05P6/D</a>	MC68HC05P6 Technical Data	pdf	7255	0	1/01/1999	<input type="checkbox"/>
<a href="#">MC68HC05P6AD/D</a>	Addendum to MC68HC05P6 Technical Data	pdf	35	0	1/01/1999	<input type="checkbox"/>

## Engineering Bulletin

ID	Name	Format	Size K	Rev #	Date Last Modified	Order Availability
<a href="#">EB166/D</a>	System Design Considerations: Converting from the MC68HC805B6 to the MC68HC705B16 Microcontroller	pdf	757	0	1/01/1993	<input type="checkbox"/>

<a href="#">EB180/D</a>	Differences between the MC68HC705B16 and the MC68HC705B16N	pdf	20	0	1/01/1996	<input type="checkbox"/>
<a href="#">EB181/D</a>	Frequently Asked Questions and Answers for the M68HC05 Family MCAN Module	pdf	181	0	1/01/1997	<input type="checkbox"/>
<a href="#">EB349/D</a>	RAM Data Retention Considerations for Motorola Microcontrollers	pdf	45	1	6/22/2000	<input type="checkbox"/>
<a href="#">EB396/D</a>	Use of OSC2/XTAL as a Clock Output on Motorola Microcontrollers	pdf	49	0	6/19/2002	<input type="checkbox"/>
<a href="#">EB413/D</a>	Resetting MCUs	pdf	62	0	1/01/2000	<input type="checkbox"/>
<a href="#">EB421/D</a>	The Motorola MCAN Module	pdf	78	0	2/23/2000	<input type="checkbox"/>

## Reference Manual

ID	Name	Format	Size K	Rev #	Date Last Modified	Order Availability
<a href="#">M68HC05AG/AD</a>	M68HC05 Applications Guide	pdf	3272	4	3/18/2002	-
<a href="#">M68HC05TB/D</a>	HC05 Family - Understanding Small Microcontrollers	pdf	2866	2	1/01/1998	<input type="checkbox"/>
<a href="#">MC68HC05CXRG/D</a>	MC68HC05C4, C8, C9, MC68HC705C8, MC68HC805C4, MC68HCL05C4, C8, MC68HSC05C4, C8 Programming Reference	pdf	3150	1	2/23/2000	-

## Roadmap

ID	Name	Format	Size K	Rev #	Date Last Modified	Order Availability
<a href="#">8BITMCURDMAP</a>	8-Bit MCU Family Roadmap	pdf	35	-	-	-

## Selector Guide

ID	Name	Format	Size K	Rev #	Date Last Modified	Order Availability
<a href="#">SG1006/D</a>	Microcontrollers SPS Sales Guide	pdf	600	0	9/26/2002	<input type="checkbox"/>
<a href="#">SG1011/D</a>	Software and Development Tools Sales Guide	pdf	259	1	9/26/2002	<input type="checkbox"/>
<a href="#">SG2000CR/D</a>	Application Selector Guide Index and Cross-Reference.	pdf	62	0	6/24/2002	<input type="checkbox"/>

[\[top\]](#)

## 68HC05P6 Development Tools/Boards

ID	Name	Vendor ID	Order Availability
<a href="#">KITMMDS05P6A</a>	Modular Development System (MMDS) Kits	MOTOROLA	<input type="checkbox"/>
<a href="#">KITMMEVS05P6A</a>	Modular Evaluation System (MMEVS)	MOTOROLA	<input type="checkbox"/>



[M68ICS05P](#)

M68ICS05P Development Tool Kit

MOTOROLA

[CWHC05](#)

CodeWarrior Development Tools for HC05

METROWERKS

[\[top\]](#)

## Design Tools

### Software

ID	Name	Vendor ID	Format	Size K	Rev #
<a href="#">HC05DRIVEWAYS</a>	Aisys Driveway for 68HC05	AISYS	exe	17205	2.4
<a href="#">MATH16ACOD</a>	General Math routines		asm	4	-
<a href="#">MATHAGBCOD</a>	General Math routines		asm	6	-

### Software Tools/Assemblers

ID	Name	Vendor ID	Format	Size K	Rev #
<a href="#">ASHC5ASM</a>	DOS based freeware assembler	MOTOROLA	arc	55	0

### Software Tools/Emulation

ID	Name	Vendor ID	Format	Size K	Rev #
<a href="#">M68EML05P6A</a>	M68EML05P6A Emulation module configuration/help file for MMDS and MMEVS.	MOTOROLA	zip	12	1

### Software/Application Software/Code Examples

ID	Name	Vendor ID	Format	Size K	Rev #
<a href="#">C8THERMSW</a>	HC05 Software Example: Home Thermostat example using the 705C8 with indoor/outdoor temperature and time of day	MOTOROLA	zip	11	-
<a href="#">FLOAT05COD</a>	Floating Point routines	MOTOROLA	zip	13	-
<a href="#">HC05DELAYSW</a>	HC05 Software Example: Subroutine that delays for a whole number of milliseconds	MOTOROLA	zip	2	-
<a href="#">HC05EXSW</a>	Library containing software examples in assembly for 68HC05	MOTOROLA	zip	45	-
<a href="#">HC05KEYINTSW</a>	HC05 Software Examples: Using keyboard interrupts and decoding a matrix keypad	MOTOROLA	zip	7	-
<a href="#">HC05KEYPADSW</a>	HC05 Software Example: Keypad debounce and decode. When a key is found, it is changed to ASCII and displayed on an LCD	MOTOROLA	zip	2	-

<a href="#">HC05LCDSW</a>	HC05 Software Example: Initializes an LCD and displays ABCDEF...S	MOTOROLA	zip	1	-
<a href="#">HC05SCISW</a>	HC05 Software Example: Serial Communications Interface example	MOTOROLA	zip	1	-
<a href="#">HC05SPISW</a>	HC05 Software Example: Serial Peripheral Interface example	MOTOROLA	zip	1	-
<a href="#">HC05SWITCHSW</a>	HC05 Software Example: Simple program that reads the state of a switch on a general-purpose I/O pin and lights an LED based on the state of the switch	MOTOROLA	zip	1	-
<a href="#">HC05TIMERSW</a>	HC05 Software Example: Using the 68HC05 16-bit Timer	MOTOROLA	zip	1	-
<a href="#">J1ALOOPSW</a>	HC05 Software Example: Paced loop program example using the 68HC705J1A	MOTOROLA	zip	1	-
<a href="#">J1APWMSW</a>	HC05 Software Example: Low frequency PWM example using the 68HC705J1A real-time interrupt and timer overflow interrupt	MOTOROLA	zip	1	-
<a href="#">J1AUARTSW</a>	HC05 Software example: Software UART example that transmits and receives data on the 68HC705J1A	MOTOROLA	zip	2	-
<a href="#">K1THERMSW</a>	HC05 Software Example: Thermometer project using the 68HC705K1	MOTOROLA	zip	5	-
<a href="#">SAMPPROGCOD</a>	Example routines	MOTOROLA	exe	35	-
<a href="#">THERM-CCOD</a>	Thermometer example in C	MOTOROLA	zip	11	-

## Software/Operating Systems

ID	Name	Vendor ID	Format	Size K	Rev #
<a href="#">PE68HC05SIM</a>	Windows upgrades for P&E's simulator software for 68HC05	PEMICRO	html	0	-

[\[top\]](#)

## Orderable Parts Information

PartNumber	Package Info	<a href="#">Life Cycle Description (code)</a>	Remarks	<a href="#">Budgetary</a>	Order Availability
				<a href="#">Price QTY 1000+ (\$US)</a>	
MC68HC05P6P	Plastic Dual In-Line Package (PDIP)	PRODUCT STABLE GROWTH/MATURITY(3)	Please go to motorola.com/mcu. Scroll down to Online Resources and refer to the Customer Interface Tool for more information.	\$2.54	-

MC68HC05P6DW	Small Outline Integrated Circuit (SOIC)	PRODUCT STABLE GROWTH/MATURITY(3)	Please go to <a href="http://motorola.com/mcu">motorola.com/mcu</a> . Scroll down to Online Resources and refer to the Customer Interface Tool for more information.	\$2.54	-
MC68HC05P6FB	10X10 mm Quad Flat Pack (QFP)	PRODUCT STABLE GROWTH/MATURITY(3)	Please go to <a href="http://motorola.com/mcu">motorola.com/mcu</a> . Scroll down to Online Resources and refer to the Customer Interface Tool for more information.	\$2.54	-
MCC68HC05PD6	-	NOT RECOMMENDED(DECLINING)(5)	Please go to <a href="http://motorola.com/mcu">motorola.com/mcu</a> . Scroll down to Online Resources and refer to the Customer Interface Tool for more information.	-	-
MC68HC705P6ECDW	Small Outline (Wide-Body SOIC)	PRODUCT MATURITY/SATURATION(4)	-40 to +85 C	\$1.89	<input type="checkbox"/>
MC68HC705P6ECP	Plastic Dual-in-Line (PDIP)	PRODUCT MATURITY/SATURATION(4)	-40 to +85 C	\$1.89	<input type="checkbox"/>
MC68HC705P6EMDW	Small Outline (Wide-Body SOIC)	PRODUCT MATURITY/SATURATION(4)	-40 to +125 C	\$2.07	<input type="checkbox"/>
MC68HC705P6EMP	Plastic Dual-in-Line (PDIP)	PRODUCT MATURITY/SATURATION(4)	-40 to +125 C	\$2.07	<input type="checkbox"/>
MC68HC705P6EVDW	Small Outline (Wide-Body SOIC)	PRODUCT MATURITY/SATURATION(4)	-40 to +105 C	\$1.98	<input type="checkbox"/>

MC68HC705P6EVP

Plastic  
Dual-in-  
Line  
(PDIP)

PRODUCT  
MATURITY/SATURATION(4)

-40 to +105 C

\$1.98



[\[top\]](#)

[Motorola Home](#) | [Semiconductors](#) | [Login](#) | [Support](#) | [Contact Us](#) | [Site Map](#)  
[Products](#) | [Documentation](#) | [Tools](#) | [Design Resources](#) | [Applications](#)



# Semiconductors

[Motorola](#) > [Semiconductors](#) >

## 68HC05P6 : Microcontroller



The Motorola MC68HC05P6 microcontroller is a low-cost combination of an M68HC05 Family microprocessor with a 4-channel, 8-bit analog-to-digital (A/D) converter, a 16-bit timer with output compare and input capture, a serial communications port (SIOP), and a computer operating properly (COP) watchdog timer. The HC05 CPU core contains 176 bytes of RAM, 4k bytes of user ROM, and 21 input/output (I/O) pins (20 bidirectional, 1 input-only). This device is available in a 28-pin SOIC, or PDIP.







 [Block Diagram](#)

### 68HC05P6 Features






- Low Cost HC05 Core
- 28-Pin SOIC, or PDIP
- 4672 Bytes of User EPROM (Including 48 Bytes of Page Zero ROM and 16 Bytes of User Vectors)
- 176 Bytes of On-Chip RAM
- Self Check ROM
- 4-Channel 8-Bit A/D Converter
- SIOP Serial Communications Port
- 16-Bit Timer with Output Compare and Input Capture
- 20 Bidirectional I/O Lines and 1 Input-Only line
- On-chip oscillator with crystal or ceramic resonator connections or resistor-capacitor connections

 [Return to Top](#)

#### Page Contents:

-  [Features](#)
-  [Documentation](#)
-  [Tools](#)
-  [Orderable Parts](#) 
-  [Related Links](#)

#### Other Info:

-  [FAQs](#)
-  [3rd Party Design Help](#)
-  [3rd Party Tool](#)
-  [Vendors](#)
-  [3rd Party Trainers](#)

#### Rate this Page












Care to Comment?

## 68HC05P6 Documentation

### Documentation

#### Application Note

ID	Name	Vendor ID	Format	Size K	Rev #	Date Last Modified	Order Availability
<a href="#">AN-HK-22</a>	MC68HC05SR3 and MC68HC705SR3 Design Notes	MOTOROLA	pdf	0	0	1/01/1994	
<a href="#">AN-HK-23</a>	MC6805R3 and MC68HC05SR3 Technical Comparison	MOTOROLA	pdf	0	0	1/01/1994	
<a href="#">AN1050_D</a>	Designing for Electromagnetic Compatibility (EMC) with HCMOS Microcontrollers	MOTOROLA	pdf	82	0	1/01/2000	-
<a href="#">AN1055/D</a>	M6805 16-Bit Support Macros	MOTOROLA	pdf	1048	0	1/01/1990	
<a href="#">AN1067/D</a>	Pulse Generation and Detection with Microcontroller Units	MOTOROLA	pdf	242	1	5/31/2002	
<a href="#">AN1222/D</a>	Arithmetic Waveform Synthesis with the HC05/08 MCUs	MOTOROLA	pdf	24	0	1/01/1993	
<a href="#">AN1222SW</a>	Software Files for AN1222 zipped	MOTOROLA	zip	20	0	1/01/1995	-
<a href="#">AN1259/D</a>	System Design and Layout Techniques for Noise Reduction in MCU-Based Systems	MOTOROLA	pdf	78	0	1/01/1995	

<a href="#">AN1262/D</a>	Simple Real-Time Kernels for M68HC05 Microcontrollers	MOTOROLA	pdf	84	0	1/01/1995	<a href="#">ORDER</a> 
<a href="#">AN1262SW</a>	Software files for AN1262	MOTOROLA	zip	11	0	1/01/1995	-
<a href="#">AN1263/D</a>	Designing for Electromagnetic Compatibility with Single-Chip Microcontrollers	MOTOROLA	pdf	104	0	1/01/1995	<a href="#">ORDER</a> 
<a href="#">AN1292/D</a>	Adding a Voice User Interface to M68HC05 Applications	MOTOROLA	pdf	155	0	1/01/1996	<a href="#">ORDER</a> 
<a href="#">AN1292SW</a>	Software files for AN1292 zipped	MOTOROLA	zip	215	0	1/01/1996	-
<a href="#">AN1516/D</a>	Liquid Level Control Using a Motorola Pressure Sensor	MOTOROLA	pdf	77	2	1/24/2003	<a href="#">ORDER</a> 
<a href="#">AN1667/D</a>	Software SCI Implementation to the MISC Communication Protocol	MOTOROLA	pdf	112	0	7/10/2002	<a href="#">ORDER</a> 
<a href="#">AN1667SW</a>	Software for AN1667, zip format	MOTOROLA	zip	93	1.0	7/31/2002	-
<a href="#">AN1688/D</a>	MISC Bus Slave Switch Node	MOTOROLA	pdf	243	0	7/11/2002	<a href="#">ORDER</a> 
<a href="#">AN1705/D</a>	Noise Reduction Techniques for Microcontroller-Based Systems	MOTOROLA	pdf	67	0	1/01/1999	<a href="#">ORDER</a> 
<a href="#">AN1723/D</a>	Interfacing MC68HC05 Microcontrollers to the IBM AT Keyboard Interface	MOTOROLA	pdf	274	0	1/01/1997	<a href="#">ORDER</a> 
<a href="#">AN1733/D</a>	Implementing Caller ID Functionality in MC68HC(7)05 Applications	MOTOROLA	pdf	312	0	1/01/1998	<a href="#">ORDER</a> 
<a href="#">AN1734/D</a>	Pulse Width Modulation Using the 16-Bit Timer	MOTOROLA	pdf	102	0	1/01/1998	<a href="#">ORDER</a> 
<a href="#">AN1734SW</a>	Software files for AN1734 zipped	MOTOROLA	zip	2	0	1/01/1997	-
<a href="#">AN1736/D</a>	Variations in the Motorola MC68HC05Px Family	MOTOROLA	pdf	203	1	1/01/1998	<a href="#">ORDER</a> 
<a href="#">AN1744/D</a>	Resetting Microcontrollers During Power Transitions	MOTOROLA	pdf	80	0	1/01/1998	<a href="#">ORDER</a> 
<a href="#">AN1752/D</a>	Data Structures for 8-Bit Microcontrollers	MOTOROLA	pdf	213	1	5/07/2001	<a href="#">ORDER</a> 
<a href="#">AN1757/D</a>	Add a Unique Silicon Serial Number to the HC05	MOTOROLA	pdf	105	0	1/01/1998	<a href="#">ORDER</a> 
<a href="#">AN1758/D</a>	Add Addressable Switches to the HC05	MOTOROLA	pdf	111	0	1/01/1998	<a href="#">ORDER</a> 
<a href="#">AN1771/D</a>	Precision Sine-Wave Tone Synthesis Using 8-Bit MCUs	MOTOROLA	pdf	250	0	1/01/1998	<a href="#">ORDER</a> 
<a href="#">AN1775/D</a>	Expanding Digital Input with an A/D Converter	MOTOROLA	pdf	86	1	1/01/1998	<a href="#">ORDER</a> 
<a href="#">AN1818/D</a>	Software SCI Routines with the 16-Bit Timer Module	MOTOROLA	pdf	84	0	1/01/1999	<a href="#">ORDER</a> 
<a href="#">AN1820/D</a>	Software I2C Communications	MOTOROLA	pdf	55	0	1/01/1999	<a href="#">ORDER</a> 
<a href="#">AN1820SW</a>	Software files for AN1820 zipped	MOTOROLA	zip	2	0	1/01/1998	-
<a href="#">AN2103/D</a>	Local Interconnect Network (LIN) Demonstration	MOTOROLA	pdf	953	0	12/01/2000	<a href="#">ORDER</a> 
<a href="#">AN2159/D</a>	Digital Direct Current Ignition System Using HC08 Microcontrollers	MOTOROLA	pdf	129	0	11/20/2001	<a href="#">ORDER</a> 
<a href="#">AN2159SW</a>	AN2159SW	MOTOROLA	zip	182	1	3/08/2002	-
<a href="#">AN2321/D</a>	Designing for Board Level Electromagnetic Compatibility	MOTOROLA	pdf	1628	0	8/15/2002	<a href="#">ORDER</a> 
<a href="#">AN4006/D</a>	Digital Captive Discharge Ignition System Using HC05/HC08 8-Bit Microcontrollers	MOTOROLA	pdf	61	0	3/27/2000	<a href="#">ORDER</a> 

<a href="#">AN442/D</a>	Driving LCDs with M6805 Microprocessors	MOTOROLA	pdf	1134	0	1/01/1991	<a href="#">ORDER</a>
<a href="#">AN463/D</a>	68HC05K0 Infra-red Remote Control	MOTOROLA	pdf	111	0	1/01/1992	<a href="#">ORDER</a>
<a href="#">AN464/D</a>	Software Driver Routines for the Motorola MC68HC05 CAN Module	MOTOROLA	pdf	2859	0	1/01/1993	<a href="#">ORDER</a>
<a href="#">AN477/D</a>	Simple A/D for MCUs without Built-In A/D Converters	MOTOROLA	pdf	224	0	1/01/1993	<a href="#">ORDER</a>
<a href="#">AN991/D</a>	Using the Serial Peripheral Interface to Communicate Between Multiple Microcomputers	MOTOROLA	pdf	251	1	1/28/2002	<a href="#">ORDER</a>
<a href="#">ANE416/D</a>	MC68HC05B4 Radio Synthesizer	MOTOROLA	pdf	1958	0	1/01/1988	<a href="#">ORDER</a>

### Brochure

ID	Name	Vendor ID	Format	Size K	Rev #	Date Last Modified	Order Availability
<a href="#">BR68HC08FAMAM/D</a>	68HC08 Family: High Performance and Flexibility	MOTOROLA	pdf	57	2	5/21/2003	<a href="#">ORDER</a>
<a href="#">FLYREMBEDFLASH/D</a>	Embedded Flash: Changing the Technology World for the Better	MOTOROLA	pdf	68	2	5/21/2003	<a href="#">ORDER</a>

### Data Sheets

ID	Name	Vendor ID	Format	Size K	Rev #	Date Last Modified	Order Availability
<a href="#">MC68HC05P6/D</a>	MC68HC05P6 Technical Data	MOTOROLA	pdf	7255	0	1/01/1999	<a href="#">ORDER</a>
<a href="#">MC68HC05P6AD/D</a>	Addendum to MC68HC05P6 Technical Data	MOTOROLA	pdf	35	0	1/01/1999	<a href="#">ORDER</a>

### Engineering Bulletin

ID	Name	Vendor ID	Format	Size K	Rev #	Date Last Modified	Order Availability
<a href="#">EB166/D</a>	System Design Considerations: Converting from the MC68HC805B6 to the MC68HC705B16 Microcontroller	MOTOROLA	pdf	757	0	1/01/1993	<a href="#">ORDER</a>
<a href="#">EB180/D</a>	Differences between the MC68HC705B16 and the MC68HC705B16N	MOTOROLA	pdf	20	0	1/01/1996	<a href="#">ORDER</a>
<a href="#">EB181/D</a>	Frequently Asked Questions and Answers for the M68HC05 Family MCAN Module	MOTOROLA	pdf	181	0	1/01/1997	<a href="#">ORDER</a>
<a href="#">EB349/D</a>	RAM Data Retention Considerations for Motorola Microcontrollers	MOTOROLA	pdf	45	1	6/22/2000	<a href="#">ORDER</a>
<a href="#">EB396/D</a>	Use of OSC2/XTAL as a Clock Output on Motorola Microcontrollers	MOTOROLA	pdf	49	0	6/19/2002	<a href="#">ORDER</a>
<a href="#">EB413/D</a>	Resetting MCUs	MOTOROLA	pdf	62	0	1/01/2000	<a href="#">ORDER</a>
<a href="#">EB421/D</a>	The Motorola MCAN Module	MOTOROLA	pdf	78	0	2/23/2000	<a href="#">ORDER</a>

### Fact Sheets

ID	Name	Vendor ID	Format	Size K	Rev #	Date Last Modified	Order Availability
<a href="#">CWDEVSTUDFACTHC08</a>	Development Studio	MOTOROLA	pdf	48	2	5/13/2002	-

### Product Change Notices

ID	Name	Vendor ID	Format	Size K	Rev #	Date Last Modified	Order Availability
<a href="#">PCN8698</a>	CARBON FIBER ITW QFP TRAY CONVERSION	MOTOROLA	htm	100	0	3/31/2003	-

## Reference Manual

ID	Name	Vendor ID	Format	Size K	Rev #	Date Last Modified	Order Availability
<a href="#">M68HC05TB/D</a>	HC05 Family - Understanding Small Microcontrollers	MOTOROLA	pdf	2866	2	1/01/1998	<a href="#">ORDER</a>
<a href="#">MC68HC05CXRG/D</a>	MC68HC05C4, C8, C9, MC68HC705C8, MC68HC805C4, MC68HCL05C4, C8, MC68HSC05C4, C8 Programming Reference	MOTOROLA	pdf	3150	1	2/23/2000	-

## Roadmap

ID	Name	Vendor ID	Format	Size K	Rev #	Date Last Modified	Order Availability
<a href="#">8BITMCURD</a>	8-Bit MCU Family Roadmap	MOTOROLA	pdf	30	0	9/01/2002	-

## Selector Guide

ID	Name	Vendor ID	Format	Size K	Rev #	Date Last Modified	Order Availability
<a href="#">SG1002</a>	Analog Selector Guide - Quarter 4, 2003	MOTOROLA	pdf	579	0	10/24/2003	<a href="#">ORDER</a>
<a href="#">SG1006</a>	Microcontrollers Selector Guide - Quarter 4, 2003	MOTOROLA	pdf	826	0	10/24/2003	<a href="#">ORDER</a>
<a href="#">SG1010</a>	Sensors Selector Guide - Quarter 4, 2003	MOTOROLA	pdf	219	0	10/24/2003	<a href="#">ORDER</a>
<a href="#">SG1011</a>	Software and Development Tools Selector Guide - Quarter 4, 2003	MOTOROLA	pdf	287	0	10/24/2003	<a href="#">ORDER</a>
<a href="#">SG2000CR</a>	Application Selector Guide Index and Cross-Reference.	MOTOROLA	pdf	95	3	11/11/2003	<a href="#">ORDER</a>
<a href="#">SG2039</a>	Application Selector Guide - Vacuum Cleaners Vacuum Cleaners	MOTOROLA	pdf	0	0	6/17/2003	<a href="#">ORDER</a>

[Return to Top](#)

## 68HC05P6 Tools

### Hardware Tools

#### Adapters

ID	Name	Vendor ID	Format	Size K	Rev #	Order Availability
<a href="#">PA-DSO-2803(6)(Z)-D420-28/2</a>	Prototyping Adapter	<a href="#">LOGSYS</a>	-	-	-	-
<a href="#">PA28SO1-08-3(6)</a>	Programming Adapter	<a href="#">LOGSYS</a>	-	-	-	-

#### Emulators/Probes/Wigglers

ID	Name	Vendor ID	Format	Size K	Rev #	Order Availability
<a href="#">AX-6811</a>	AX-6811	<a href="#">HITEX</a>	-	-	-	-
<a href="#">IC10000</a>	iC1000 PowerEmulator	<a href="#">ISYS</a>	-	-	-	-
<a href="#">IC20000</a>	iC2000 PowerEmulator	<a href="#">ISYS</a>	-	-	-	-
<a href="#">IC40000</a>	iC4000 ActiveEmulator	<a href="#">ISYS</a>	-	-	-	-

#### Evaluation/Development Boards and Systems

ID	Name	Vendor ID	Format	Size K	Rev #	Order Availability
<a href="#">KITMMDS05P6A</a>	Modular Development System (MMDS) Kits	MOTOROLA	-	-	-	<a href="#">BUY</a>
<a href="#">KITMMEVS05P6A</a>	Modular Evaluation System (MMEVS)	MOTOROLA	-	-	-	-
<a href="#">M68ICS05P</a>	M68ICS05P Development Tool Kit	MOTOROLA	-	-	-	<a href="#">BUY</a>



## Programmers

ID	Name	Vendor ID	Format	Size K	Rev #	Order Availability
<a href="#">POWERLAB</a>	Universal Programmer	<a href="#">SYSGEN</a>	-	-	-	-

## Software

### Application Software

#### Code Examples

ID	Name	Vendor ID	Format	Size K	Rev #	Order Availability
<a href="#">C8THERMSW</a>	HC05 Software Example: Home Thermostat example using the 705C8 with indoor/outdoor temperature and time of day	MOTOROLA	zip	11	-	-
<a href="#">FLOAT05COD</a>	Floating Point routines	MOTOROLA	zip	13	-	-
<a href="#">HC05DELAWSW</a>	HC05 Software Example: Subroutine that delays for a whole number of milliseconds	MOTOROLA	zip	2	-	-
<a href="#">HC05EXSW</a>	Library containing software examples in assembly for 68HC05	MOTOROLA	zip	45	-	-
<a href="#">HC05KEYINTSW</a>	HC05 Software Examples: Using keyboard interrupts and decoding a matrix keypad	MOTOROLA	zip	7	-	-
<a href="#">HC05KEYPADSW</a>	HC05 Software Example: Keypad debounce and decode. When a key is found, it is changed to ASCII and displayed on an LCD	MOTOROLA	zip	2	-	-
<a href="#">HC05LCDSW</a>	HC05 Software Example: Initializes an LCD and displays ABCDEF...S	MOTOROLA	zip	1	-	-
<a href="#">HC05SCISW</a>	HC05 Software Example: Serial Communications Interface example	MOTOROLA	zip	1	-	-
<a href="#">HC05SPISW</a>	HC05 Software Example: Serial Peripheral Interface example	MOTOROLA	zip	1	-	-
<a href="#">HC05SWITCHSW</a>	HC05 Software Example: Simple program that reads the state of a switch on a general-purpose I/O pin and lights an LED based on the state of the switch	MOTOROLA	zip	1	-	-
<a href="#">HC05TIMERSW</a>	HC05 Software Example: Using the 68HC05 16-bit Timer	MOTOROLA	zip	1	-	-
<a href="#">J1ALOOPSW</a>	HC05 Software Example: Paced loop program example using the 68HC705J1A	MOTOROLA	zip	1	-	-
<a href="#">J1APWMSW</a>	HC05 Software Example: Low frequency PWM example using the 68HC705J1A real-time interrupt and timer overflow interrupt	MOTOROLA	zip	1	-	-
<a href="#">J1AUARTSW</a>	HC05 Software example: Software UART example that transmits and receives data on the 68HC705J1A	MOTOROLA	zip	2	-	-
<a href="#">K1THERMSW</a>	HC05 Software Example: Thermometer project using the 68HC705K1	MOTOROLA	zip	5	-	-
<a href="#">MATH16ACOD</a>	General Math routines	MOTOROLA	asm	4	-	-
<a href="#">MATHAGBCOD</a>	General Math routines	MOTOROLA	asm	6	-	-
<a href="#">SAMPPROGCOD</a>	Example routines	MOTOROLA	exe	35	-	-
<a href="#">THERM-CCOD</a>	Thermometer example in C	MOTOROLA	zip	11	-	-

**Software Tools**  
**Assemblers**

ID	Name	Vendor ID	Format	Size K	Rev #	Order Availability
<a href="#">ASHC5ASM</a>	DOS based freeware assembler	MOTOROLA	arc	55	0	-
<a href="#">AX6805</a>	AX6805 relocatable/absolute macro assembler for HC05	<a href="#">COSMIC</a>	-	-	-	-

**Compilers**

ID	Name	Vendor ID	Format	Size K	Rev #	Order Availability
<a href="#">CWHC05</a>	CodeWarrior Development Tools for HC05	<a href="#">METROWERKS</a>	-	-	-	<a href="#">BUY</a>
<a href="#">CX6805</a>	CX6805 C Cross Compiler for HC05	<a href="#">COSMIC</a>	-	-	-	-

**Debuggers**

ID	Name	Vendor ID	Format	Size K	Rev #	Order Availability
<a href="#">CWHC05</a>	CodeWarrior Development Tools for HC05	<a href="#">METROWERKS</a>	-	-	-	<a href="#">BUY</a>
<a href="#">ZAP 6805 MMDS</a>	ZAP 6805 MMDS Debugger	<a href="#">COSMIC</a>	-	-	-	-
<a href="#">ZAP 6805 SIM</a>	ZAP 6805 Simulator Debugger	<a href="#">COSMIC</a>	-	-	-	-
<a href="#">AX-6811</a>	AX-6811	<a href="#">HITEX</a>	-	-	-	-

**Emulation**

ID	Name	Vendor ID	Format	Size K	Rev #	Order Availability
<a href="#">M68EML05P6A</a>	M68EML05P6A Emulation module configuration/help file for MMDS and MMEVS.	MOTOROLA	zip	12	1	-

**IDE (Integrated Development Environment)**

ID	Name	Vendor ID	Format	Size K	Rev #	Order Availability
<a href="#">CWHC05</a>	CodeWarrior Development Tools for HC05	<a href="#">METROWERKS</a>	-	-	-	<a href="#">BUY</a>
<a href="#">IDEA05</a>	IDEA05 integrated development environment for HC05	<a href="#">COSMIC</a>	-	-	-	-
<a href="#">IC-SW-OPR</a>	winIDEA	<a href="#">ISYS</a>	-	-	-	-

**Models**

**Instruction Set Simulator**

ID	Name	Vendor ID	Format	Size K	Rev #	Order Availability
<a href="#">PE68HC05SIM</a>	Windows upgrades for P&E's simulator software for 68HC05	<a href="#">PEMICRO</a>	html	0	-	-

**Performance and Testing**

ID	Name	Vendor ID	Format	Size K	Rev #	Order Availability
<a href="#">AX-6811</a>	AX-6811	<a href="#">HITEX</a>	-	-	-	-

[Return to Top](#)

**Orderable Parts Information**

PartNumber	Package Info	Tape and Reel	Life Cycle Description (code)	Budgetary Price QTY 1000+ (\$US)	Additional Info	Order Availability

MC68HC05P6DW	<a href="#">SOIC 28W</a>	No	PRODUCT STABLE GROWTH/MATURITY(3)	\$2.54	<a href="#">more</a>	-
MC68HC05P6FB	<a href="#">LQFP 32 7*7*1.4P0.8</a>	No	PRODUCT STABLE GROWTH/MATURITY(3)	\$2.54	<a href="#">more</a>	-
MC68HC05P6P	<a href="#">PDIP 28</a>	No	PRODUCT STABLE GROWTH/MATURITY(3)	\$2.54	<a href="#">more</a>	-
MC68HC705P6ECDW	<a href="#">SOIC 28W</a>	No	PRODUCT MATURITY/SATURATION(4)	\$1.89	<a href="#">more</a>	<a href="#">BUY</a> 
MC68HC705P6ECP	<a href="#">PDIP 28</a>	No	PRODUCT MATURITY/SATURATION(4)	\$1.89	<a href="#">more</a>	<a href="#">BUY</a> 
MC68HC705P6EMDW	<a href="#">SOIC 28W</a>	No	PRODUCT MATURITY/SATURATION(4)	\$2.07	<a href="#">more</a>	<a href="#">BUY</a> 
MC68HC705P6EMP	<a href="#">PDIP 28</a>	No	PRODUCT MATURITY/SATURATION(4)	\$2.07	<a href="#">more</a>	<a href="#">BUY</a> 
MC68HC705P6EVDW	<a href="#">SOIC 28W</a>	No	PRODUCT MATURITY/SATURATION(4)	\$1.98	<a href="#">more</a>	<a href="#">BUY</a> 
MC68HC705P6EVP	<a href="#">PDIP 28</a>	No	PRODUCT MATURITY/SATURATION(4)	\$1.98	<a href="#">more</a>	<a href="#">BUY</a> 
MCC68HC05PD6	CHIPS SM <50000 SQ MILS	No	NOT RECOMMENDED(DECLINING)(5)	-	<a href="#">more</a>	-

**NOTE:** Are you looking for an obsolete orderable part? Click [HERE](#) to check our distributors' inventory.

[Return to Top](#)

#### Related Links

- [Automotive](#)
- [Microcontrollers](#)
- [Motor Control](#)

[Return to Top](#)