

# MC68HC05RC8 MC68HC05RC16

## General Release Specification

October 24, 1996

CSIC MCU Design Center  
Austin, Texas



*Motorola reserves the right to make changes without further notice to any products herein to improve reliability, function or design. Motorola does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part.*

## List of Sections

Section 1. General Description . . . . .	15
Section 2. Memory . . . . .	27
Section 3. Central Processor Unit . . . . .	33
Section 4. Interrupts . . . . .	37
Section 5. Resets . . . . .	45
Section 6. Low-Power Modes . . . . .	53
Section 7. Parallel Input/Output (I/O) . . . . .	57
Section 8. Core Timer . . . . .	61
Section 9. Carrier Modulator Transmitter (CMT) . . . . .	67
Section 10. Instruction Set . . . . .	85
Section 11. Electrical Specifications . . . . .	103
Section 12. Mechanical Specifications . . . . .	111
Section 13. Ordering Information . . . . .	115
Appendix A. MC68HC05RC8 . . . . .	119

# List of Sections

## Table of Contents

### Section 1. General Description

1.1	Contents .....	15
1.2	Introduction .....	16
1.3	Features .....	16
1.4	Mask Options .....	19
1.5	Signal Description .....	21
1.5.1	$V_{DD}$ and $V_{SS}$ .....	23
1.5.2	$\overline{IRQ}$ (Maskable Interrupt Request) .....	23
1.5.3	$OSC1$ and $OSC2$ .....	24
1.5.4	$\overline{RESET}$ .....	25
1.5.5	$\overline{LPRST}$ .....	25
1.5.6	$IRO$ .....	25
1.5.7	$PA0$ – $PA7$ .....	25
1.5.8	$PB0$ – $PB7$ .....	25
1.5.9	$PC0$ – $PC3$ ( $PC4$ – $PC7$ ) .....	26

### Section 2. Memory

2.1	Contents .....	27
2.2	Introduction .....	27
2.3	Memory Map .....	27
2.3.1	ROM .....	30
2.3.2	ROM Security .....	30
2.3.3	RAM .....	31
2.4	Input/Output Programming .....	31

## Section 3. Central Processor Unit

3.1	Contents . . . . .	33
3.2	Introduction . . . . .	33
3.3	Accumulator . . . . .	34
3.4	Index Register . . . . .	34
3.5	Condition Code Register . . . . .	35
3.6	Stack Pointer . . . . .	36
3.7	Program Counter . . . . .	36

## Section 4. Interrupts

4.1	Contents . . . . .	37
4.2	Introduction . . . . .	37
4.3	CPU Interrupt Processing . . . . .	38
4.4	Reset Interrupt Sequence . . . . .	39
4.5	Software Interrupt (SWI) . . . . .	39
4.6	Hardware Interrupts . . . . .	41
4.7	External Interrupt ( $\overline{\text{IRQ}}$ /Port B Keyscan) . . . . .	41
4.8	External Interrupt Timing . . . . .	42
4.9	Carrier Modulator Transmitter Interrupt (CMT) . . . . .	42
4.10	Core Timer Interrupt . . . . .	43

## Section 5. Resets

5.1	Contents . . . . .	45
5.2	Introduction . . . . .	45
5.3	External Reset ( $\overline{\text{RESET}}$ ) . . . . .	46
5.4	Low-Power External Reset (LPRST) . . . . .	48

5.5	Internal Resets . . . . .	48
5.5.1	Power-On Reset (POR). . . . .	48
5.5.2	Computer Operating Properly Reset (COPR) . . . . .	49
5.5.2.1	Resetting the COP . . . . .	49
5.5.2.2	COP During Wait Mode . . . . .	49
5.5.2.3	COP During Stop Mode . . . . .	49
5.5.2.4	COP Watchdog Timer Considerations . . . . .	50
5.5.2.5	COP Register . . . . .	51
5.5.3	Illegal Address . . . . .	51

## Section 6. Low-Power Modes

6.1	Contents . . . . .	53
6.2	Introduction . . . . .	53
6.3	Stop Mode . . . . .	53
6.4	Stop Recovery . . . . .	54
6.5	Wait Mode . . . . .	54
6.6	Low-Power Reset . . . . .	55

## Section 7. Parallel Input/Output (I/O)

7.1	Contents . . . . .	57
7.2	Introduction . . . . .	57
7.3	Port A . . . . .	57
7.4	Port B . . . . .	58
7.5	Port C . . . . .	58
7.6	Input/Output Programming . . . . .	59

## Section 8. Core Timer

8.1	Contents . . . . .	61
8.2	Introduction . . . . .	61
8.3	Core Timer Control and Status Register . . . . .	63
8.4	Core Timer Counter Register . . . . .	65
8.5	Computer Operating Properly (COP) Reset . . . . .	66
8.6	Timer During Wait Mode . . . . .	66

## Section 9. Carrier Modulator Transmitter (CMT)

9.1	Contents . . . . .	67
9.2	Introduction . . . . .	67
9.3	Overview . . . . .	68
9.4	Carrier Generator . . . . .	70
9.4.1	Time Counter . . . . .	71
9.4.2	Carrier Generator Data Registers (CHR1, CLR1, CHR2, and CLR2) . . . . .	72
9.5	Modulator . . . . .	74
9.5.1	Time Mode . . . . .	76
9.5.2	FSK Mode . . . . .	77
9.5.3	Extended Space Operation . . . . .	78
9.5.3.1	End Of Cycle (EOC) Interrupt . . . . .	79
9.5.3.2	Modulator Control and Status Register . . . . .	80
9.5.4	Modulator Period Data Registers (MDR1, MDR2, and MDR3) . . . . .	83



## Section 10. Instruction Set

10.1	Contents . . . . .	85
10.2	Introduction . . . . .	86
10.3	Addressing Modes . . . . .	86
10.3.1	Inherent . . . . .	87
10.3.2	Immediate . . . . .	87
10.3.3	Direct . . . . .	87
10.3.4	Extended . . . . .	87
10.3.5	Indexed, No Offset . . . . .	88
10.3.6	Indexed, 8-Bit Offset . . . . .	88
10.3.7	Indexed, 16-Bit Offset . . . . .	88
10.3.8	Relative . . . . .	89
10.4	Instruction Types . . . . .	89
10.4.1	Register/Memory Instructions . . . . .	90
10.4.2	Read-Modify-Write Instructions . . . . .	91
10.4.3	Jump/Branch Instructions . . . . .	92
10.4.4	Bit Manipulation Instructions . . . . .	94
10.4.5	Control Instructions . . . . .	95
10.5	Instruction Set Summary . . . . .	96

## Section 11. Electrical Specifications

11.1	Contents . . . . .	103
11.2	Introduction . . . . .	103
11.3	Maximum Ratings . . . . .	104
11.4	Operating Range . . . . .	105
11.5	Thermal Characteristics . . . . .	105
11.6	DC Electrical Characteristics (5.0 Vdc) . . . . .	106
11.7	DC Electrical Characteristics (2.2 Vdc) . . . . .	107
11.8	Control Timing (5.0 Vdc and 2.2 V <sub>dc</sub> ) . . . . .	109

## Section 12. Mechanical Specifications

12.1	Contents . . . . .	111
12.2	Introduction . . . . .	111
12.3	28-Pin Plastic Dual In-Line Package (Case 710-02) . . . . .	112
12.4	28-Pin Small Outline Integrated Circuit Package (Case 751F-04) . . . . .	112
12.5	44-Pin Plastic Leaded Chip Carrier Package (Case 777-02) . . . . .	113

## Section 13. Ordering Information

13.1	Contents . . . . .	115
13.2	Introduction . . . . .	115
13.3	MCU Ordering Forms . . . . .	115
13.4	Application Program Media . . . . .	116
13.5	ROM Program Verification . . . . .	117
13.6	ROM Verification Units (RVUs) . . . . .	118
13.7	MC Order Numbers . . . . .	118

## Appendix A. MC68HC05RC8

A.1	Contents . . . . .	119
A.2	Introduction . . . . .	119
A.3	Memory Map . . . . .	119

## List of Figures

Figure	Title	Page
1-1	MC68HC05RC16 Block Diagram . . . . .	18
1-2	28-Pin DIP Pinout . . . . .	21
1-3	28-Pin SOIC Pinout . . . . .	22
1-4	44-Pin PLCC Pinout . . . . .	22
1-5	Oscillator Connections . . . . .	24
2-1	MC68HC05RC16 Memory Map . . . . .	28
2-2	I/O Registers . . . . .	29
3-1	Programming Model . . . . .	33
3-2	Stacking Order . . . . .	34
4-1	Interrupt Processing Flowchart . . . . .	40
4-2	IRQ Function Block Diagram . . . . .	41
5-1	Reset Block Diagram . . . . .	46
5-2	Reset and POR Timing Diagram . . . . .	47
5-3	COP Watchdog Timer Location . . . . .	51
6-1	Stop Recovery Timing Diagram . . . . .	54
6-2	Stop/Wait Flowchart . . . . .	56
7-1	Port B Pullup Option . . . . .	58
7-2	I/O Circuitry . . . . .	60
8-1	Core Timer Block Diagram . . . . .	62
8-2	Core Timer Control and Status Register (CTCSR) . . . . .	63
8-3	Core Timer Counter Register (CTCR) . . . . .	65

## List of Figures

Figure	Title	Page
9-1	Carrier Modulator Transmitter Module Block Diagram . . . . .	69
9-2	Carrier Generator Block Diagram . . . . .	70
9-3	Carrier Generator Data Register CHR1 . . . . .	72
9-4	Carrier Generator Data Register CLR1 . . . . .	72
9-5	Carrier Generator Data Register CHR2 . . . . .	72
9-6	Carrier Generator Data Register CLR2 . . . . .	73
9-7	Modulator Block Diagram . . . . .	75
9-8	CMT Operation in Time Mode . . . . .	77
9-9	Extended Space Operation . . . . .	79
9-10	Modulator Control and Status Register (MCSR) . . . . .	80
9-11	Modulator Period Data Register MDR1 . . . . .	83
9-12	Modulator Period Data Register MDR2 . . . . .	83
9-13	Modulator Period Data Register MDR3 . . . . .	83
11-1	Maximum Supply Current versus Internal Clock Frequency . . . . .	108
A-1	MC68HC05RC8 Memory Map . . . . .	120

## List of Tables

Table	Title	Page
4-1	Vector Address for Interrupts and Reset .....	38
5-1	COP Watchdog Timer Recommendations .....	50
7-1	I/O Pin Functions .....	59
8-1	RTI and COP Rates at 4.096 MHz Oscillator .....	64
10-1	Register/Memory Instructions .....	90
10-2	Read-Modify-Write Instructions .....	91
10-3	Jump and Branch Instructions .....	93
10-4	Bit Manipulation Instructions .....	94
10-5	Control Instructions .....	95
10-6	Instruction Set Summary .....	96
10-7	Opcode Map .....	102
13-1	MC Order Numbers .....	118



## Section 1. General Description

### 1.1 Contents

1.2	Introduction .....	16
1.3	Features .....	16
1.4	Mask Options .....	19
1.5	Signal Description .....	21
1.5.1	$V_{DD}$ and $V_{SS}$ .....	23
1.5.2	$\overline{IRQ}$ (Maskable Interrupt Request) .....	23
1.5.3	$\overline{OSC1}$ and $\overline{OSC2}$ .....	24
1.5.4	$\overline{RESET}$ .....	25
1.5.5	$\overline{LPRST}$ .....	25
1.5.6	IRO .....	25
1.5.7	PA0–PA7 .....	25
1.5.8	PB0–PB7 .....	25
1.5.9	PC0–PC3 (PC4–PC7) .....	26

## 1.2 Introduction

The MC68HC05RC16 is a low-cost addition to the M68HC05 Family of microcontrollers (MCUs) and is suitable for remote control applications. This device contains the HC05 central processing unit (CPU) core, including the 14-stage core timer with real-time interrupt (RTI) and computer operating properly (COP) watchdog systems. On-chip peripherals include a carrier modulator transmitter. The 16-kbyte memory map has 15,936 bytes of user ROM and 352 bytes of RAM. There are 20 input/output (I/O) lines (eight having keyscan pullups/interrupts) and a low-power reset pin. This device is available in 28-pin small outline integrated circuit (SOIC), 28-pin dual in-line (DIP), and 44-pin plastic leaded chip carrier (PLCC) packages. Four additional I/O lines are available for bond out on the higher pin count package.

## 1.3 Features

Features for the MC68HC05RC16 include:

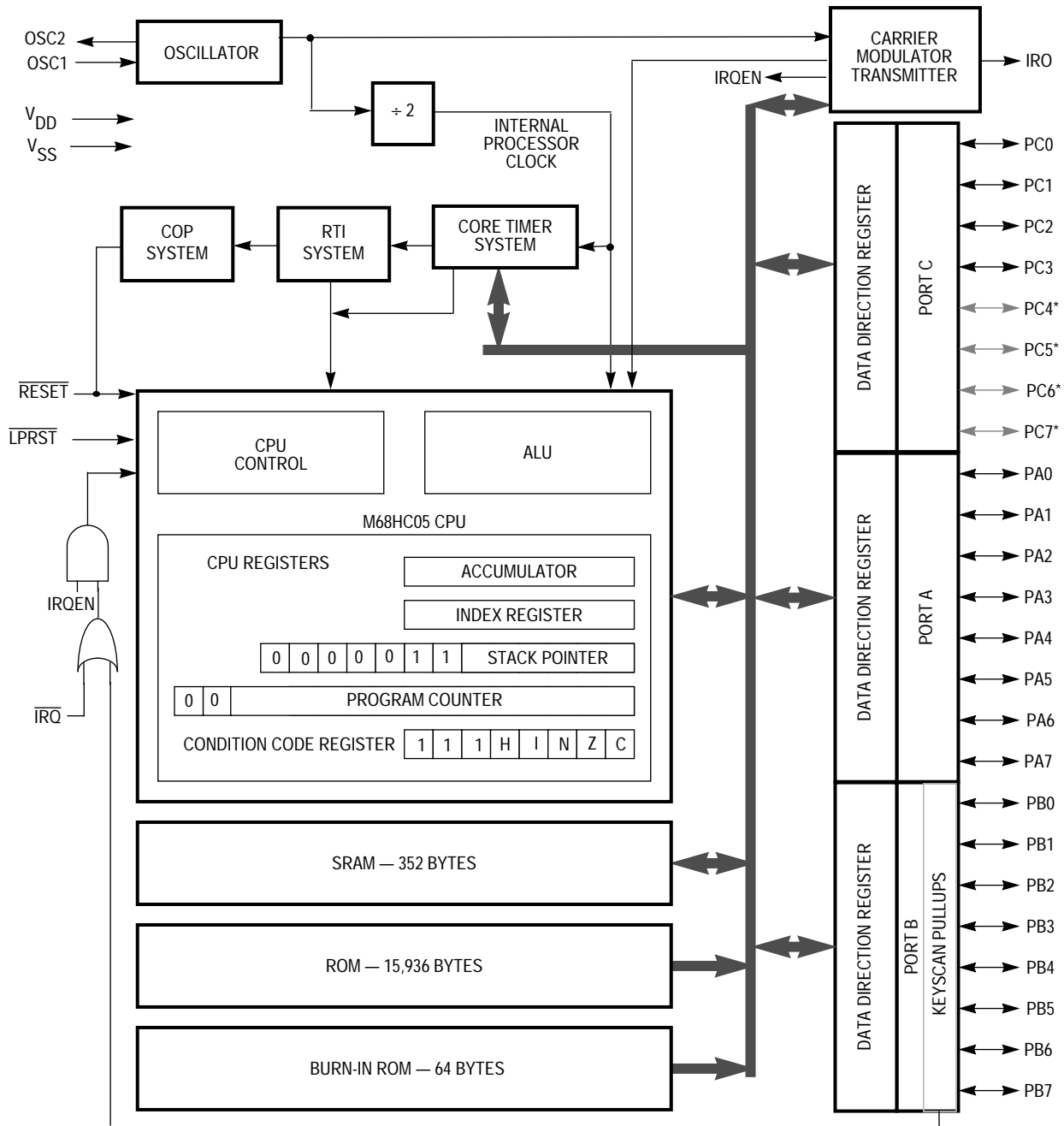
- Low Cost
- HC05 Core
- 28-Pin Plastic Dual In-Line (PDIP), Small Outline Integrated Circuit (SOIC), or Plastic Leaded Chip Carrier (PLCC) Packages
- On-Chip Oscillator with Crystal/Ceramic Resonator
- 4-MHz Maximum Oscillator Frequency at 5 V and 2.2 V Supply
- Fully Static Operation
- 15,936 Bytes of User ROM
- 64 Bytes of Burn-In ROM
- 352 Bytes of On-Chip RAM
- 14-Stage Core Timer with Real-Time Interrupt (RTI) and Computer Operating Properly (COP) Watchdog Circuits
- Carrier Modulator Transmitter Supporting Baseband, Pulse Length Modulator (PLM), and Frequency Shift Keying (FSK) Protocols



- Low-Power Reset Pin
- 20 Bidirectional I/O Lines (Four Additional I/O Lines Available for Bond Out in 44-Lead PLCC Package)
- Mask Programmable Pullups and Interrupts on Eight Port Pins (PB0–PB7)
- High-Current Infrared (IR) Drive Pin
- High-Current Port Pin (PC0)
- Power-Saving Stop and Wait Modes
- Mask Selectable Options:
  - COP Watchdog Timer
  - STOP Instruction Disable
  - Edge-Sensitive or Edge- and Level-Sensitive Interrupt Trigger
  - Port B Pullups for Keyscan
- Illegal Address Reset
- ROM Security Feature

**NOTE:** *A line over a signal name indicates an active low signal. For example, RESET is active low.*

# General Description



\* Marked pins are available only 44-lead PLCC package.

**Figure 1-1. MC68HC05RC16 Block Diagram**

## 1.4 Mask Options

There are 11 total mask options on the MC68HC05RC16 including:

- Eight port B pullups
- IRQ sensitivity
- COP enable/disable
- STOP enable/disable

These are nonprogrammable options in that they are selected at the time of code submission (when masks are made). These options are as follows:

### PB7PU — Port B7 Pullup/Interrupt

- This bit enables or disables the pullup/interrupt on port B, bit 7.
- 1 = Enables the pullup/interrupt
- 0 = Disables the pullup/interrupt

### PB6PU — Port B6 Pullup/Interrupt

- This option enables or disables the pullup/interrupt on port B, bit 6.
- 1 = Enables pullup/interrupt
- 0 = Disables pullup/interrupt

### PB5PU — Port B5 Pullup/Interrupt

- This option enables or disables the pullup/interrupt on port B, bit 5.
- 1 = Enables pullup/interrupt
- 0 = Disables pullup/interrupt

### PB4PU — Port B4 Pullup/Interrupt

- This option enables or disables the pullup/interrupt on port B, bit 4.
- 1 = Enables pullup/interrupt
- 0 = Disables pullup/interrupt

### PB3PU — Port B3 Pullup/Interrupt

- This option enables or disables the pullup/interrupt on port B, bit 3.
- 1 = Enables pullup/interrupt
- 0 = Disables pullup/interrupt

### PB2PU — Port B2 Pullup/Interrupt

This option enables or disables the pullup/interrupt on port B, bit 2.

1 = Enables pullup/interrupt

0 = Disables pullup/interrupt

### PB1PU — Port B1 Pullup/Interrupt

This option enables or disables the pullup/interrupt on port B, bit 1.

1 = Enables pullup/interrupt

0 = Disables pullup/interrupt

### PB0PU — Port B0 Pullup/Interrupt

This option enables or disables the pullup/interrupt on port B, bit 0.

1 = Enables pullup/interrupt

0 = Disables pullup/interrupt

### COPEN — COP Enable

When the COP option is selected (COPEN = 1), the COP watchdog timer is enabled.

When the COP option is deselected (COPEN = 0), the COP watchdog timer is disabled.

### STOPEN — STOP Instruction Enable

When the STOP option is selected (STOPEN = 1), the STOP instruction is enabled.

When the STOP option is deselected (STOPEN = 0), the STOP instruction is equivalent to a WAIT instruction.

### IRQ — IRQ sensitivity

When the IRQ option is selected (IRQ = 1), edge- and level-sensitive IRQ is enabled.

When the IRQ option is deselected (IRQ = 0), edge-only sensitive IRQ is enabled.

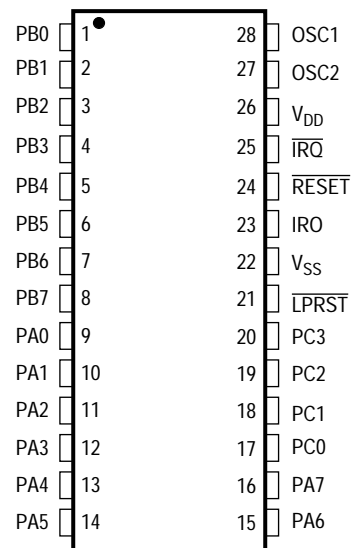
**NOTE:** *The port B keyscan interrupt sensitivity will match that of the IRQ sensitivity. (See [4.7 External Interrupt \(IRQ/Port B Keyscan\)](#) for more information.)*

## 1.5 Signal Description

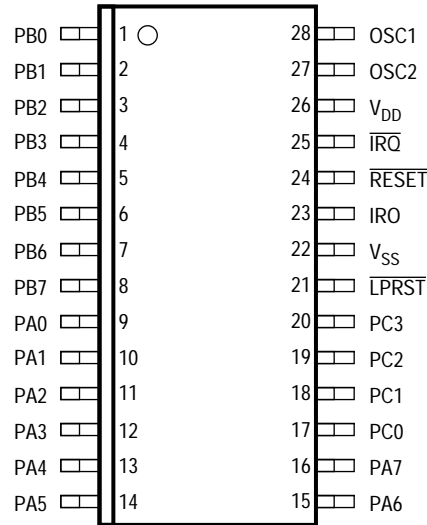
The MC68HC05RC16 is available in

1. 28-pin dual-in-line package (DIP) see [Figure 1-2](#)
2. 28-pin small outline integrated circuit (SOIC) package see [Figure 1-3](#)
3. 44-pin plastic leaded chip carrier (PLCC) package see [Figure 1-4](#)

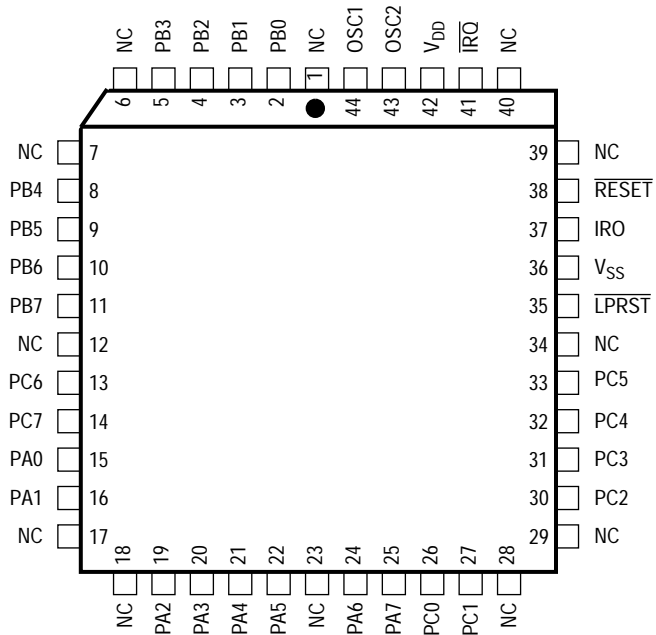
The signals are described in the following subsections.



**Figure 1-2. 28-Pin DIP Pinout**



**Figure 1-3. 28-Pin SOIC Pinout**



NOTE: NC = No Connect  
 All no connects should be tied to an appropriate logic level (either V<sub>DD</sub> or V<sub>SS</sub>).

**Figure 1-4. 44-Pin PLCC Pinout**

### 1.5.1 $V_{DD}$ and $V_{SS}$

Power is supplied to the microcontroller's digital circuits using these two pins.  $V_{DD}$  is the positive supply and  $V_{SS}$  is ground.

### 1.5.2 $\overline{IRQ}$ (Maskable Interrupt Request)

In addition to supplying the EPROM with the required programming voltage, this pin has a mask option as specified by the user that provides one of two different choices of interrupt triggering sensitivity. The options are:

1. Negative edge-sensitive triggering only
2. Both negative edge-sensitive and level-sensitive triggering.

The MCU completes the current instruction before it responds to the interrupt request. When  $\overline{IRQ}$  goes low for at least one  $t_{ILIH}$  (see [11.8 Control Timing \(5.0 Vdc and 2.2 Vdc\)](#)), a logic 1 is latched internally to signify that an interrupt has been requested. When the MCU completes its current instruction, the interrupt latch is tested. If the interrupt latch contains a logic 1 and the interrupt mask bit (I bit) in the condition code register is clear, the MCU then begins the interrupt sequence.

If the option is selected to include level-sensitive triggering, the  $\overline{IRQ}$  input requires an external resistor to  $V_{DD}$  for wired-OR operation.

The  $\overline{IRQ}$  pin contains an internal Schmitt trigger as part of its input to improve noise immunity.

Refer to [Section 4. Interrupts](#) for more detail.

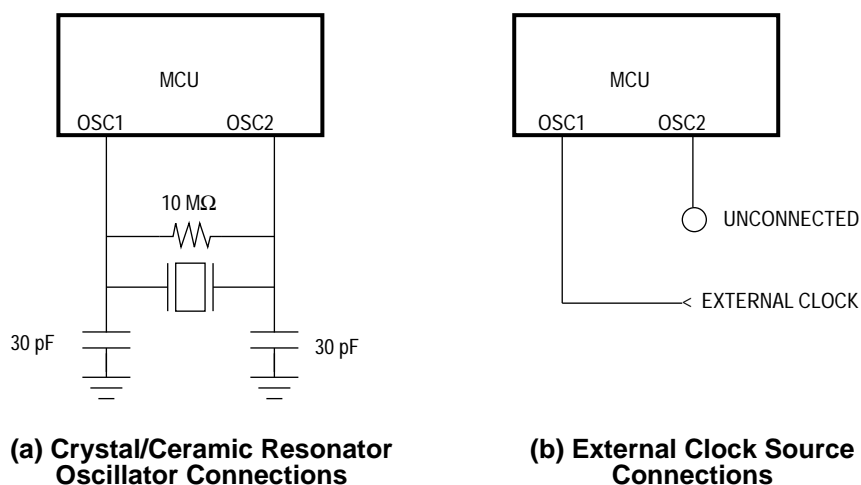
## 1.5.3 OSC1 and OSC2

These pins provide control input for an on-chip clock oscillator circuit. A crystal, a ceramic resonator, or an external signal connects to these pins to provide a system clock. The oscillator frequency is two times the internal bus rate.

**Figure 1-5** shows the recommended circuit when using a crystal. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time.

A ceramic resonator may be used in place of the crystal in cost-sensitive applications. **Figure 1-5** (a) shows the recommended circuit for using a ceramic resonator. The manufacturer of the particular ceramic resonator being considered should be consulted for specific information.

An external clock should be applied to the OSC1 input with the OSC2 pin not connected (see **Figure 1-5** (b)). This setup can be used if the user does not want to run the CPU with a crystal.



**Figure 1-5. Oscillator Connections**



#### 1.5.4 $\overline{\text{RESET}}$

This active-low pin is used to reset the MCU to a known startup state by pulling  $\overline{\text{RESET}}$  low. The  $\overline{\text{RESET}}$  pin contains an internal Schmitt trigger as part of its input to improve noise immunity. See [Section 5. Resets](#).

#### 1.5.5 $\overline{\text{LPRST}}$

The  $\overline{\text{LPRST}}$  pin is an active-low pin and is used to put the MCU into low-power reset mode. In low-power reset mode the MCU is held in reset with all processor clocks halted. See [Section 5. Resets](#).

#### 1.5.6 IRO

The IRO pin is the high-current source and sink output of the carrier modulator transmitter subsystem which is suitable for driving infrared (IR) LED biasing logic. See [Section 9. Carrier Modulator Transmitter \(CMT\)](#).

#### 1.5.7 PA0–PA7

These eight I/O lines comprise port A. The state of any pin is software programmable and all port A lines are configured as inputs during power-on or reset. For detailed information on I/O programming, see [2.4 Input/Output Programming](#).

#### 1.5.8 PB0–PB7

These eight I/O lines comprise port B. The state of any pin is software programmable and all port B lines are configured as inputs during power-on or reset. Each port B I/O line has a mask optionable pullup/interrupt for keyscan. For detailed information on I/O programming, see [2.4 Input/Output Programming](#).

### 1.5.9 PC0–PC3 (PC4–PC7)

These eight I/O lines comprise port C. PC0 is a high-current pin. PC4–PC7 are available only in the 44-lead PLCC package. The state of any pin is software programmable and all port C lines are configured as input during power-on or reset. For detailed information on I/O programming, see [2.4 Input/Output Programming](#).

**NOTE:** *Only four bits of port C are bonded out in 28-pin packages for the MC68HC05RC16, although port C is truly an 8-bit port. Since pins PC4–PC7 are unbonded, software should include the code to set their respective data direction register locations to outputs to avoid floating inputs.*

**NOTE:** *Any unused inputs, I/O ports, and no connects should be tied to an appropriate logic level (either  $V_{DD}$  or  $V_{SS}$ ). Although the I/O ports of the do not require termination, termination is recommended to reduce the possibility of static damage.*

## Section 2. Memory

### 2.1 Contents

2.2	Introduction .....	27
2.3	Memory Map .....	27
2.3.1	ROM .....	30
2.3.2	ROM Security .....	30
2.3.3	RAM .....	31
2.4	Input/Output Programming .....	31

### 2.2 Introduction

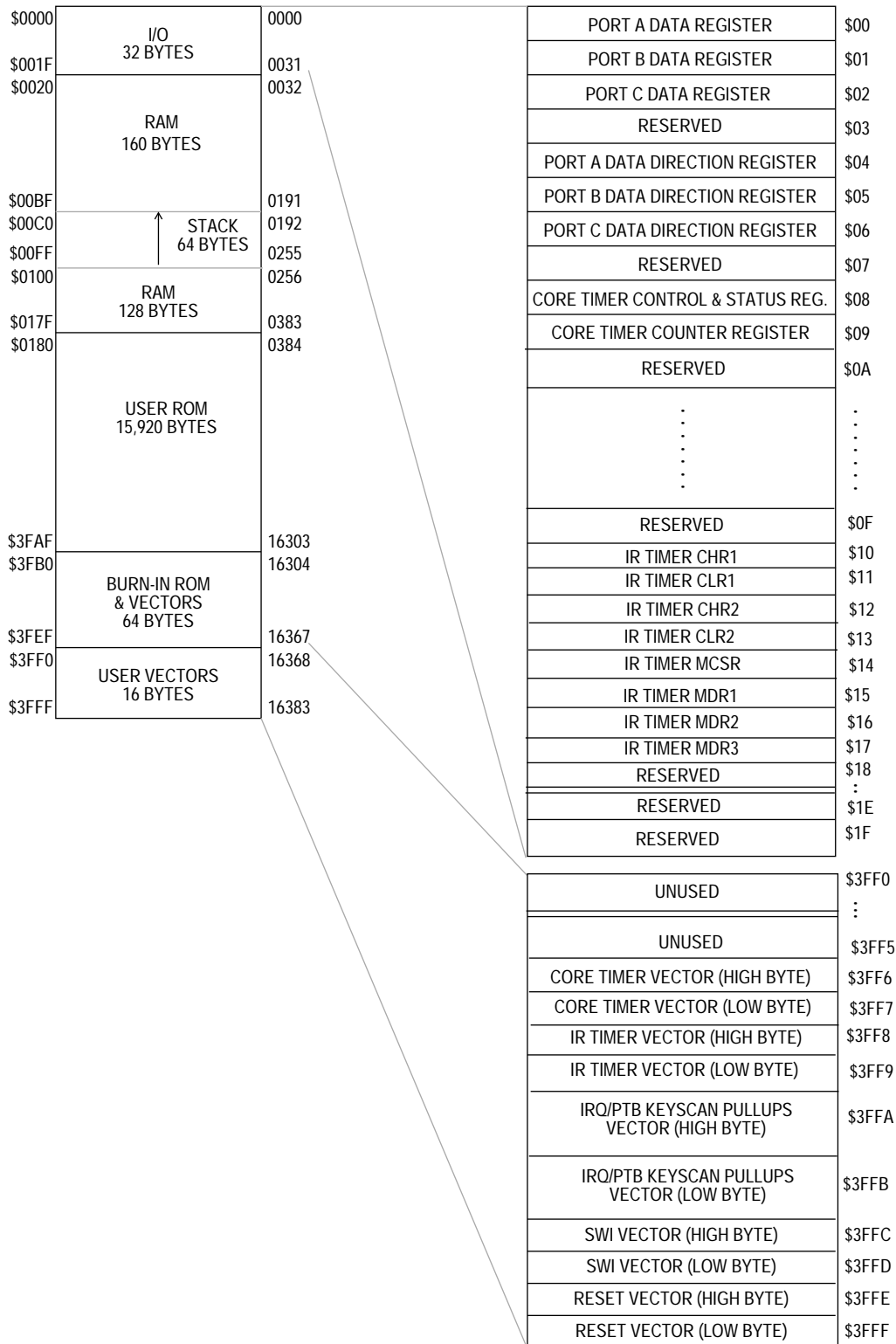
This section describes the organization of the on-chip memory.

### 2.3 Memory Map

The MC68HC05RC16 has a 16-Kbyte memory map consisting of user ROM, RAM, burn-in ROM, and input/output (I/O).

**Figure 2-1** shows the MC68HC05RC16 memory map in user mode.

# Memory



**Figure 2-1. MC68HC05RC16 Memory Map**

Addr.	Register	Bit 7	6	5	4	3	2	1	Bit 0
\$0000	Port A Data Register								
\$0001	Port B Data Register								
\$0002	Port C Data Register								
\$0003	Reserved	R	R	R	R	R	R	R	R
\$0004	Port A Data Direction Register								
\$0005	Port B Data Direction Register								
\$0006	Port C Data Direction Register								
\$0007	Reserved	R	R	R	R	R	R	R	R
\$0008	Timer Control and Status Reg.	CTOF	RTIF	TOFE	RTIE	TOFC	RTFC	RT1	RT0
\$0009	Timer Counter Register								
\$000A	Reserved	R	R	R	R	R	R	R	R
\$000B	Reserved	R	R	R	R	R	R	R	R
\$000C	Reserved	R	R	R	R	R	R	R	R
\$000D	Reserved	R	R	R	R	R	R	R	R
\$000E	Reserved	R	R	R	R	R	R	R	R
\$000F	Reserved	R	R	R	R	R	R	R	R
\$0010	IR Timer CHR1	IROLN	0	PH5	PH4	PH3	PH2	PH1	PH0
\$0011	IR Timer CLR1	IROLP	0	PL5	PL4	PL3	PL2	PL1	PL0
\$0012	IR Timer CHR2	0	0	SH5	SH4	SH3	SH2	SH1	SH0
\$0013	IR Timer CLR2	0	0	SL5	SL4	SL3	SL2	SL1	SL0
\$0014	IR Timer MCSR	EOC	0	EIMSK	EXMRK	BASE	MODE	EOCIE	MCGEN
\$0015	IR Timer MDR1	MB11	MB10	MB9	MB8	SB11	SB10	SB9	SB8
\$0016	IR Timer MDR2	MB7	MB6	MB5	MB4	MB3	MB2	MB1	MB0
\$0017	IR Timer MDR3	SB7	SB6	SB5	SB4	SB3	SB2	SB1	SB0
\$0018	Reserved	R	R	R	R	R	R	R	R
\$0019	Reserved	R	R	R	R	R	R	R	R

R = Reserved

**Figure 2-2. I/O Registers**

Addr.	Register	Bit 7	6	5	4	3	2	1	Bit 0
\$001A	Reserved	R	R	R	R	R	R	R	R
\$001B	Reserved	R	R	R	R	R	R	R	R
\$001C	Reserved	R	R	R	R	R	R	R	R
\$001D	Reserved	R	R	R	R	R	R	R	R
\$001E	Reserved	R	R	R	R	R	R	R	R
\$001F	Reserved	R	R	R	R	R	R	R	R

R

 = Reserved

**Figure 2-2. I/O Registers (Continued)**

### 2.3.1 ROM

The user ROM consists of 15,920 bytes of ROM located from \$0180 to \$3FAF and 16 bytes of user vectors located from \$3FF0 to \$3FFF.

The burn-in ROM is located from \$3FB0 to \$3FEF.

Ten of the user vectors, \$3FF6–\$3FFF, are dedicated to reset and interrupt vectors. The six remaining locations — \$3FF0, \$3FF1, \$3FF2, \$3FF3, \$3FF4, and \$3FF5 — are general-purpose user ROM locations.

### 2.3.2 ROM Security

Security has been incorporated into the MC68HC05RC16 to prevent external viewing of the ROM contents. This feature ensures that customer-developed software remains proprietary.<sup>1</sup>

<sup>1</sup> No security feature is absolutely secure. However, Motorola's strategy is to make reading or copying the ROM difficult for unauthorized users.

### 2.3.3 RAM

The user RAM consists of 352 bytes of a shared stack area. The RAM starts at address \$0020 and ends at address \$017F. The stack begins at address \$00FF. The stack pointer can access 64 bytes of RAM in the range \$00FF to \$00C0.

**NOTE:** *Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.*

## 2.4 Input/Output Programming

In user mode, 20 lines (28-pin PDIP or 28-pin SOIC) or 24 lines (44-lead PLCC) are arranged as three 8-bit I/O ports. These ports are programmable as either inputs or outputs under software control of the data direction registers. For detailed information, refer to [Section 7. Parallel Input/Output \(I/O\)](#).





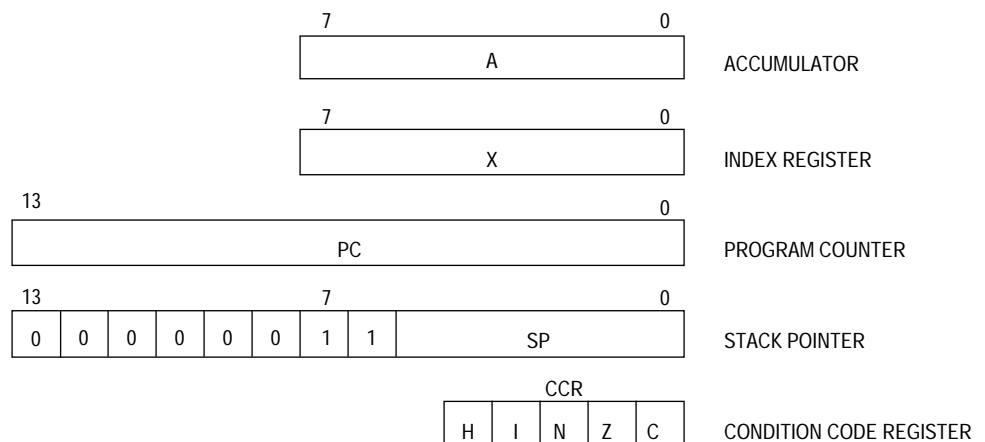
## Section 3. Central Processor Unit

### 3.1 Contents

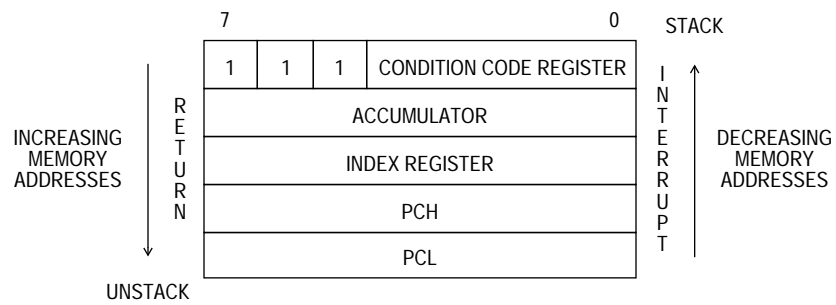
3.2	Introduction . . . . .	33
3.3	Accumulator . . . . .	34
3.4	Index Register. . . . .	34
3.5	Condition Code Register. . . . .	35
3.6	Stack Pointer . . . . .	36
3.7	Program Counter . . . . .	36

### 3.2 Introduction

This section describes the registers of the MC68HC05RC16 central processor unit (CPU). The MCU contains five registers as shown in **Figure 3-1**. The interrupt stacking order is shown in **Figure 3-2**.



**Figure 3-1. Programming Model**

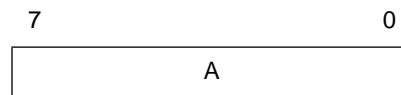


NOTE: Since the stack pointer decrements during pushes, the PCL is stacked first, followed by PCH, etc. Pulling from the stack is in the reverse order.

**Figure 3-2. Stacking Order**

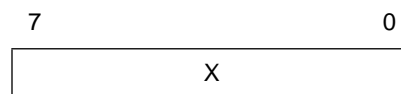
### 3.3 Accumulator

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



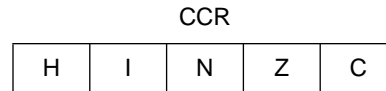
### 3.4 Index Register

The index register (X) is an 8-bit register used for the indexed addressing value to create an effective address. The index register also may be used as a temporary storage area.



### 3.5 Condition Code Register

The condition code register (CCR) is a 5-bit register in which four bits are used to indicate the results of the instruction just executed, and the fifth bit indicates whether interrupts are masked. These bits can be tested individually by a program, and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.



#### H — Half Carry

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

#### I — Interrupt

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

#### N — Negative

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative.

#### Z — Zero

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

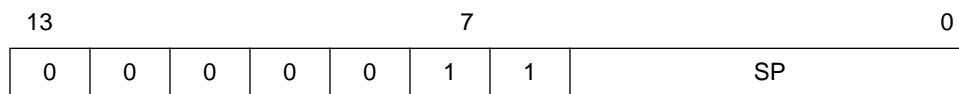
#### C — Carry/Borrow

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions and during shifts and rotates.

## 3.6 Stack Pointer

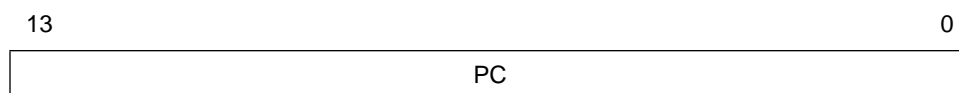
The stack pointer (SP) contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set to location \$00FF. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

When accessing memory, the seven most significant bits are permanently set to 000011. These seven bits are appended to the six least significant register bits to produce an address within the range of \$00FF to \$00C0. Subroutines and interrupts may use up to 64 (decimal) locations. If 64 locations are exceeded, the stack pointer wraps around and loses the previously stored information. A subroutine call occupies two locations on the stack; an interrupt uses five locations.



## 3.7 Program Counter

The program counter (PC) is a 13-bit register that contains the address of the next byte to be fetched.



**NOTE:** *The HC05 CPU core is capable of addressing a 64-Kbyte memory map. For this implementation, however, the addressing registers are limited to an 16-Kbyte memory map.*

## Section 4. Interrupts

### 4.1 Contents

4.2	Introduction . . . . .	37
4.3	CPU Interrupt Processing . . . . .	38
4.4	Reset Interrupt Sequence . . . . .	39
4.5	Software Interrupt (SWI) . . . . .	39
4.6	Hardware Interrupts . . . . .	41
4.7	External Interrupt ( $\overline{\text{IRQ}}$ /Port B Keyscan) . . . . .	41
4.8	External Interrupt Timing . . . . .	42
4.9	Carrier Modulator Transmitter Interrupt (CMT) . . . . .	42
4.10	Core Timer Interrupt . . . . .	43

### 4.2 Introduction

The MCU can be interrupted four different ways:

1. Nonmaskable software interrupt instruction (SWI)
2. External asynchronous interrupt ( $\overline{\text{IRQ}}$ /port B keyscan)
3. Internal carrier modulator transmitter interrupt
4. Internal core timer interrupt

## 4.3 CPU Interrupt Processing

Interrupts cause the processor to save register contents on the stack and to set the interrupt mask (I bit) to prevent additional interrupts. Unlike reset, hardware interrupts do not cause the current instruction execution to be halted, but are considered pending until the current instruction is complete.

If interrupts are not masked (I bit in the CCR is clear) and the corresponding interrupt enable bit is set, the processor will proceed with interrupt processing. Otherwise, the next instruction is fetched and executed. If an interrupt occurs, the processor completes the current instruction, stacks the current CPU register state, sets the I bit to inhibit further interrupts, and finally checks the pending hardware interrupts. If more than one interrupt is pending after the stacking operation, the interrupt with the highest vector location shown in [Table 4-1](#) will be serviced first. The SWI is executed the same as any other instruction, regardless of the I-bit state.

When an interrupt is to be processed, the CPU fetches the address of the appropriate interrupt software service routine from the vector table at locations \$3FF6–\$3FFF as defined in [Table 4-1](#).

**Table 4-1. Vector Address for Interrupts and Reset**

Register	Flag Name	Interrupt	CPU Interrupt	Vector Address
N/A	N/A	Reset	RESET	\$3FFE–\$3FFF
N/A	N/A	Software Interrupt	SWI	\$3FFC–\$3FFD
N/A	N/A	External Interrupts*	IRQ	\$3FFA–\$3FFB
MCSR	EOC	End of Cycle Interrupt	CMT	\$3FF8–\$3FF9
CTCSR	CTOF, RTIF	Real-Time Interrupt Core Timer Overflow	CORE TIMER	\$3FF6–\$3FF7

\*External interrupts include  $\overline{\text{IRQ}}$  and port B keyscan sources.

The M68HC05 CPU does not support interruptible instructions. The maximum latency to the first instruction of the interrupt service routine must include the longest instruction execution time plus stacking overhead.

$$\text{Latency} = (\text{Longest instruction execution time} + 10) \times t_{\text{cyc}} \text{ seconds}$$

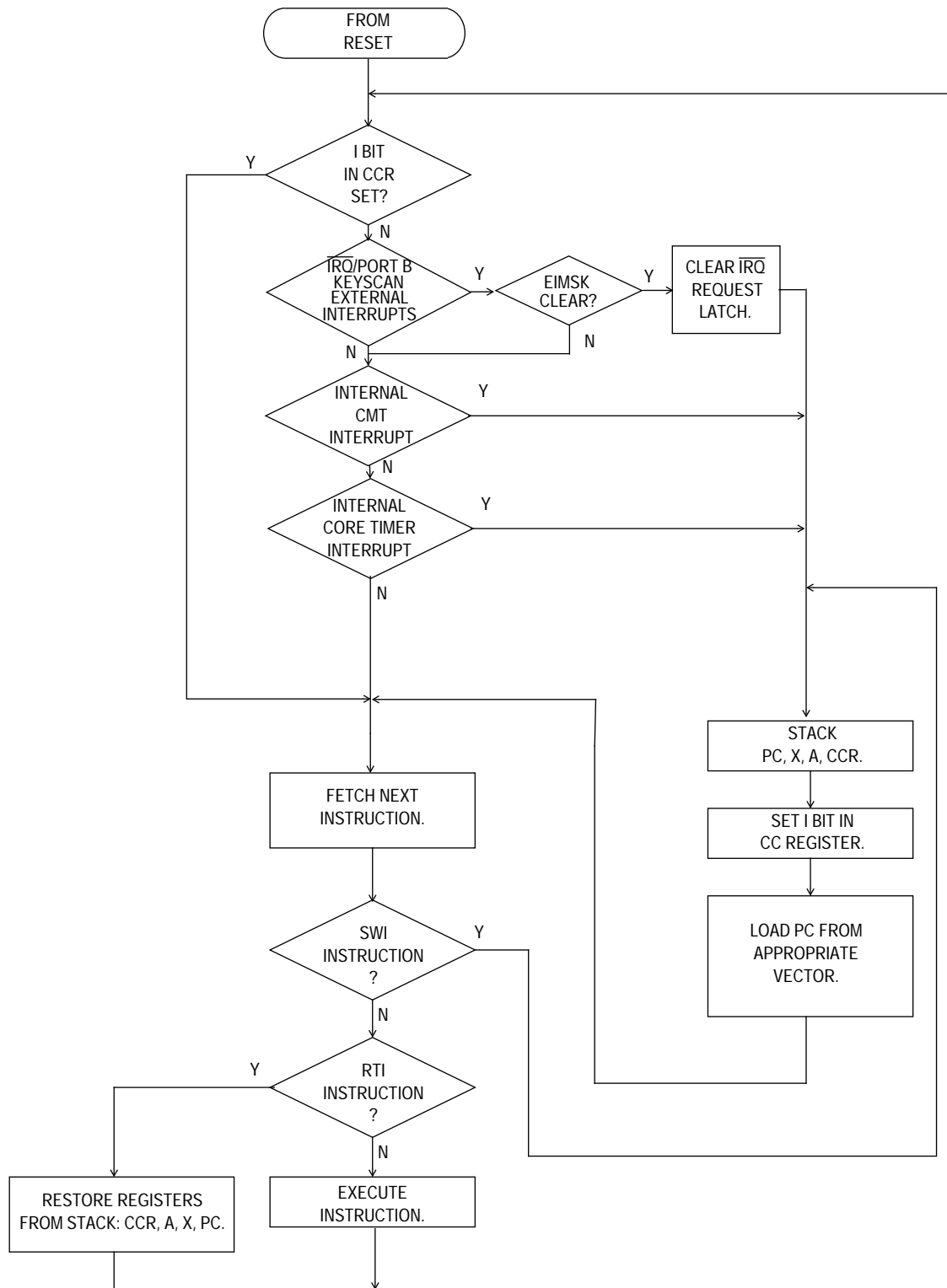
An RTI instruction is used to signify when the interrupt software service routine is completed. The RTI instruction causes the register contents to be recovered from the stack and normal processing to resume at the next instruction that was to be executed when the interrupt took place. **Figure 4-1** shows the sequence of events that occurs during interrupt processing.

## 4.4 Reset Interrupt Sequence

The reset function is not in the strictest sense an interrupt; however, it is acted upon in a similar manner as shown in **Figure 4-1**. A low-level input on the RESET pin or an internally generated RST signal causes the program to vector to its starting address, which is specified by the contents of memory locations \$3FFE and \$3FFF. The I bit in the condition code register is also set. The MCU is configured to a known state during this type of reset.

## 4.5 Software Interrupt (SWI)

The SWI is an executable instruction and a nonmaskable interrupt since it is executed regardless of the state of the I bit in the CCR. If the I bit is zero (interrupts enabled), the SWI instruction executes after interrupts that were pending before the SWI was fetched or before interrupts generated after the SWI was fetched. The interrupt service routine address is specified by the contents of memory locations \$3FFC and \$3FFD.



**Figure 4-1. Interrupt Processing Flowchart**



## 4.6 Hardware Interrupts

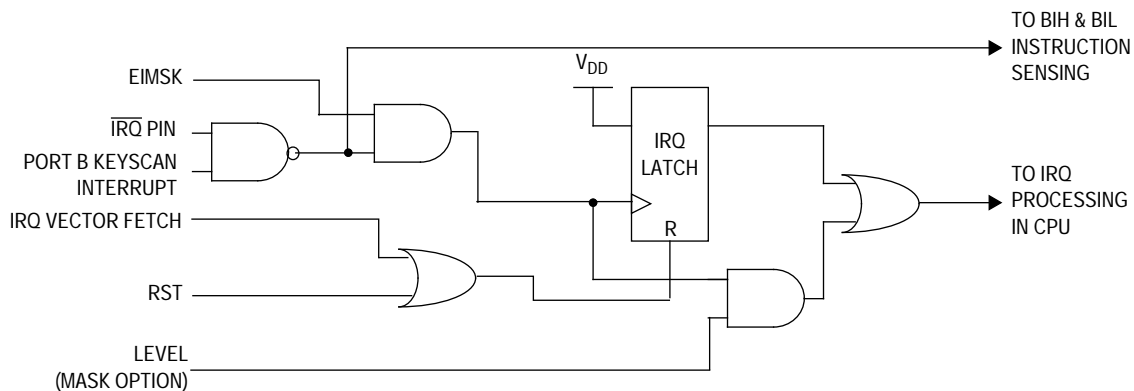
All hardware interrupts except  $\overline{\text{RESET}}$  are maskable by the I bit in the CCR. If the I bit is set, all hardware interrupts (internal and external) are disabled. Clearing the I bit enables the hardware interrupts. The three types of hardware interrupts are explained in the following sections.

### 4.7 External Interrupt ( $\overline{\text{IRQ}}$ /Port B Keyscan)

The  $\overline{\text{IRQ}}$  pin provides an asynchronous interrupt to the CPU. A block diagram of the IRQ function is shown in **Figure 4-2**.

**NOTE:** *The BIH and BIL instructions will apply to the level on the  $\overline{\text{IRQ}}$  pin itself and to the output of the logic OR function with the port B IRQ interrupts. The states of the individual port B pins can be checked by reading the appropriate port B pins as inputs.*

The  $\overline{\text{IRQ}}$  pin is one source of an external interrupt. All port B pins (PB0–PB7) act as other external interrupt sources if the pullup feature is enabled as specified by the user.



**Figure 4-2. IRQ Function Block Diagram**

When edge sensitivity is selected for the IRQ interrupt, it is sensitive to these cases:

1. Falling edge on the  $\overline{\text{IRQ}}$  pin
2. Falling edge on any port B pin with pullup enabled

When edge and level sensitivity is selected for the IRQ interrupt, it is sensitive to these cases:

1. Low level on the  $\overline{\text{IRQ}}$  pin
2. Falling edge on the  $\overline{\text{IRQ}}$  pin
3. Falling edge or low level on any port B pin with pullup enabled

External interrupts also can be masked by setting the EIMSK bit in the MSCR register of the IR remote timer. See [9.5.4 Modulator Period Data Registers \(MDR1, MDR2, and MDR3\)](#) for details.

## 4.8 External Interrupt Timing

If the interrupt mask bit (I bit) of the CCR is set, all maskable interrupts (internal and external) are disabled. Clearing the I bit enables interrupts. The interrupt request is latched immediately following the falling edge of the  $\overline{\text{IRQ}}$  source. It is then synchronized internally and serviced as specified by the contents of \$3FFA and \$3FFB.

Either a level-sensitive and edge-sensitive trigger or an edge-sensitive-only trigger is available via the mask programmable option for the  $\overline{\text{IRQ}}$  pin.

## 4.9 Carrier Modulator Transmitter Interrupt (CMT)

A CMT interrupt occurs when the end of cycle flag (EOC) and the end of cycle interrupt enable (EOCIE) bits are set in the modulator control and status register (MCSR). This interrupt will vector to the interrupt service routine located at the address specified by the contents of memory locations \$3FF8 and \$3FF9.

## 4.10 Core Timer Interrupt

This timer can create two types of interrupts. A timer overflow interrupt occurs whenever the 8-bit timer rolls over from \$FF to \$00 and the enable bit TOFE is set. A real-time interrupt occurs whenever the programmed time elapses and the enable bit RTIE is set. Either of these interrupts vectors to the same interrupt service routine, located at the address specified by the contents of memory locations \$3FF6 and \$3FF7.



## Section 5. Resets

### 5.1 Contents

5.2	Introduction . . . . .	45
5.3	External Reset ( $\overline{\text{RESET}}$ ) . . . . .	46
5.4	Low-Power External Reset ( $\overline{\text{LPRST}}$ ) . . . . .	48
5.5	Internal Resets . . . . .	48
5.5.1	Power-On Reset (POR) . . . . .	48
5.5.2	Computer Operating Properly Reset (COPR) . . . . .	49
5.5.2.1	Resetting the COP . . . . .	49
5.5.2.2	COP During Wait Mode . . . . .	49
5.5.2.3	COP During Stop Mode . . . . .	49
5.5.2.4	COP Watchdog Timer Considerations . . . . .	50
5.5.2.5	COP Register . . . . .	51
5.5.3	Illegal Address . . . . .	51

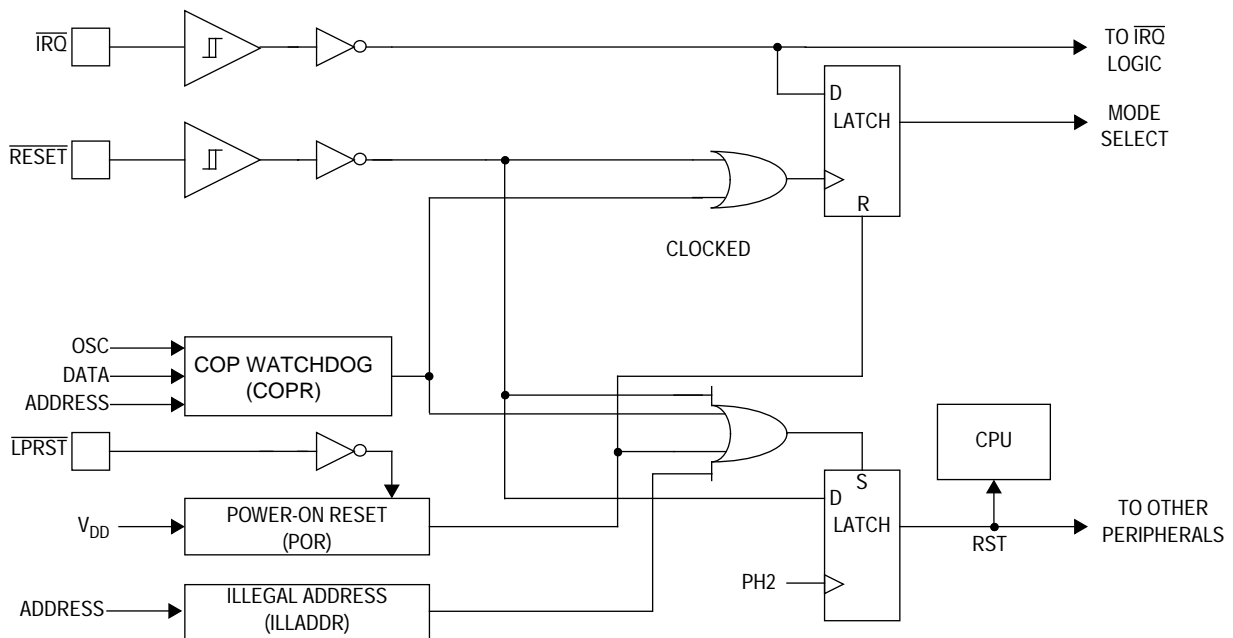
### 5.2 Introduction

The MCU can be reset from five sources: two external inputs and three internal restart conditions. The  $\overline{\text{RESET}}$  and  $\overline{\text{LPRST}}$  pins are inputs as shown in [Figure 5-1](#). All the internal peripheral modules will be reset by the internal reset signal (RST). Refer to [Figure 5-2](#) for reset timing detail.

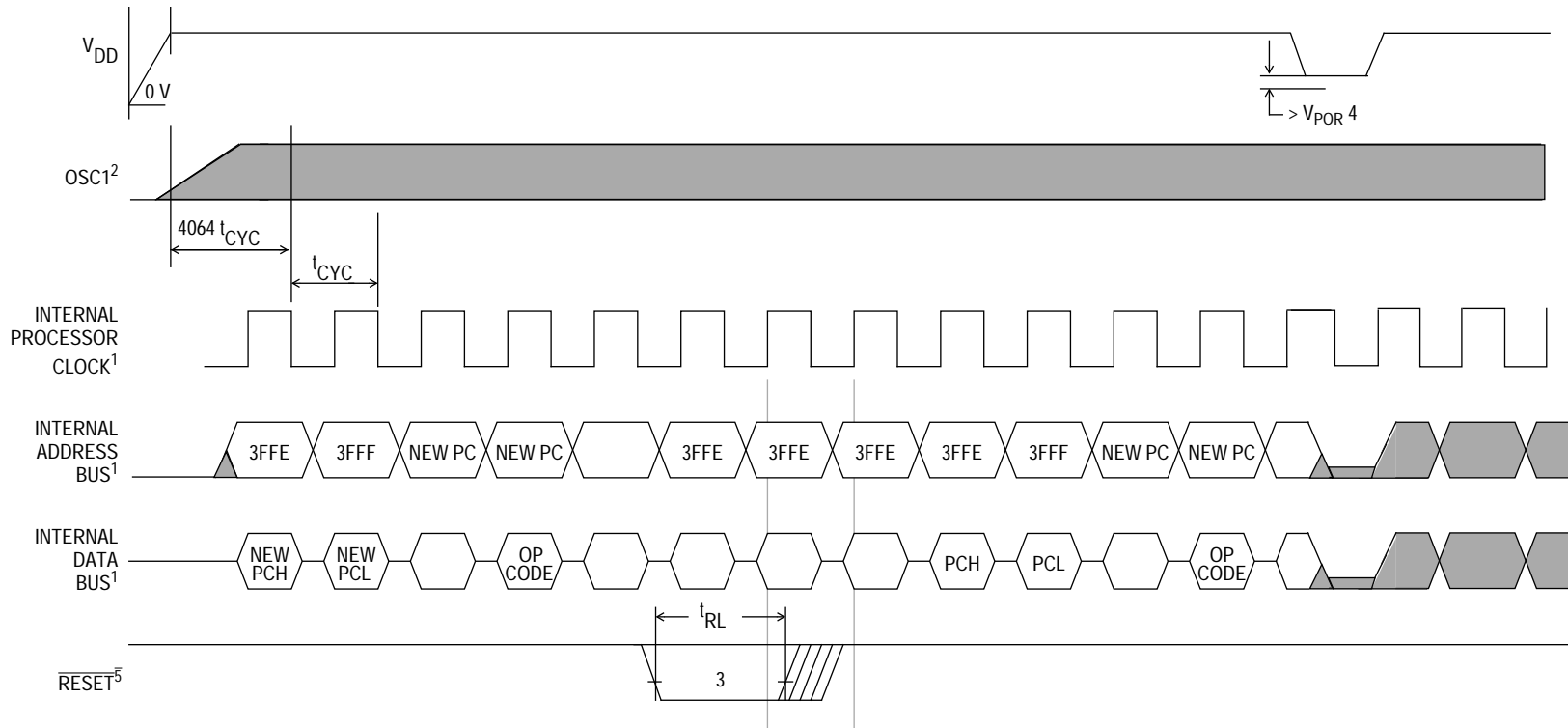
## 5.3 External Reset ( $\overline{\text{RESET}}$ )

The  $\overline{\text{RESET}}$  pin is one of the two external sources of a reset. This pin is connected to a Schmitt trigger input gate to provide an upper and lower threshold voltage separated by a minimum amount of hysteresis. This external reset occurs whenever the  $\overline{\text{RESET}}$  pin is pulled below the lower threshold and remains in reset until the  $\overline{\text{RESET}}$  pin rises above the upper threshold. This active-low input will generate the RST signal and reset the CPU and peripherals. Termination of the external RESET input or the internal COP watchdog reset are the only reset sources that can alter the operating mode of the MCU.

**NOTE:** *Activation of the RST signal is generally referred to as reset of the device, unless otherwise specified.*



**Figure 5-1. Reset Block Diagram**



NOTES:

1. Internal timing signal and bus information are not available externally.
2.  $OSC1$  line is not meant to represent frequency. It is only used to represent time.
3. The next rising edge of the internal processor clock following the rising edge of  $\overline{RESET}$  initiates the reset sequence.
4.  $V_{DD}$  must fall to a level lower than  $V_{POR}$  to be recognized as a power-on reset.
5. The  $\overline{LPRST}$  pin resets the CPU like  $\overline{RESET}$ . However, 4064 POR cycles are executed first, before the reset vector address appears on the internal address bus. (See [5.4 Low-Power External Reset \(LPRST\)](#).)

Figure 5-2. Reset and POR Timing Diagram

## 5.4 Low-Power External Reset ( $\overline{\text{LPRST}}$ )

The  $\overline{\text{LPRST}}$  pin is one of the two external sources of a reset. This external reset occurs whenever the  $\overline{\text{LPRST}}$  pin is pulled below the lower threshold and remains in reset until the  $\overline{\text{LPRST}}$  pin rises. This active low input will, in addition to generating the RST signal and resetting the CPU and peripherals, halt all internal processor clocks. The MCU will remain in this low-power reset condition as long as a logic 0 remains on LPRST. When a logic 1 is applied to LPRST, processor clocks will be re-enabled with the MCU remaining in reset until the 4064 internal processor clock cycle ( $t_{\text{cyc}}$ ) oscillator stabilization delay is completed. If any other reset function is active at the end of this 4064-cycle delay, the RST signal remains in the reset condition until the other reset condition(s) end.

## 5.5 Internal Resets

The three internally generated resets are the initial power-on reset function, the COP watchdog timer reset, and the illegal address detector. Termination of the external reset input, external  $\overline{\text{LPRST}}$  input, or the internal COP watchdog timer are the only reset sources that can alter the operating mode of the MCU. The other internal resets do not have any effect on the mode of operation when their reset state ends.

### 5.5.1 Power-On Reset (POR)

The internal POR is generated on power-up to allow the clock oscillator to stabilize. The POR is strictly for power turn-on conditions and is not able to detect a drop in the power supply voltage (brown-out). There is an oscillator stabilization delay of 4064 internal processor bus clock cycles (PH2) after the oscillator becomes active.

The POR generates the RST signal that resets the CPU. If any other reset function is active at the end of this 4064-cycle delay, the RST signal remains in the reset condition until the other reset condition(s) ends.



## 5.5.2 Computer Operating Properly Reset (COPR)

The MCU contains a watchdog timer that automatically times out if not reset (cleared) within a specific time by a program reset sequence. If the COP watchdog timer is allowed to time out, an internal reset is generated to reset the MCU.

The COP reset function is enabled or disabled by a mask option and is verified during production testing.

### 5.5.2.1 Resetting the COP

Writing a zero to the COPF bit prevents a COP reset. This action resets the counter and begins the time-out period again. The COPF bit is bit 0 of address \$3FF0. A read of address \$3FF0 returns user data programmed at that location.

### 5.5.2.2 COP During Wait Mode

The COP continues to operate normally during wait mode. The software should pull the device out of wait mode periodically and reset the COP by writing to the COPF bit to prevent a COP reset.

### 5.5.2.3 COP During Stop Mode

When the stop enable mask option is selected, stop mode disables the oscillator circuit and thereby turns the clock off for the entire device. When stop is executed, the COP counter will hold its current state. If a reset is used to exit stop mode, the COP counter is reset and held until 4064 POR cycles are completed at this time, counting will begin. If an external IRQ is used to exit stop mode, the COP counter does not wait for the completion of the 4064 POR cycles but does count these cycles. It is, therefore, recommended that the COP is fed before executing the STOP instruction.

#### 5.5.2.4 COP Watchdog Timer Considerations

The COP watchdog timer is active in all modes of operation if enabled by a mask option. If the COP watchdog timer is selected by a mask option, any execution of the STOP instruction (either intentionally or inadvertently due to the CPU being disturbed) causes the oscillator to halt and prevents the COP watchdog timer from timing out. If the COP watchdog timer is selected by a mask option, the COP resets the MCU when it times out. Therefore, it is recommended that the COP watchdog be **disabled** for a system that must have intentional uses of the wait mode for periods longer than the COP time out period.

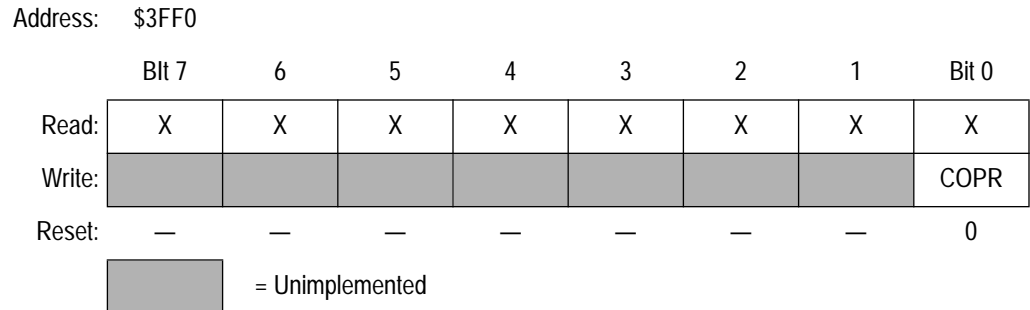
The recommended interactions and considerations for the COP watchdog timer, STOP instruction, and WAIT instruction are summarized in [Table 5-1](#).

**Table 5-1. COP Watchdog Timer Recommendations**

IF the Following Conditions Exist:	THEN the COP Watchdog Timer Should Be as Follows:
Wait Time	
Wait Time <b>Less than</b> COP Time-Out	Enable or Disable COP by Mask Option
Wait Time <b>More than</b> COP Time-Out	Disable COP by Mask Option
Any Length Wait Time	Disable COP by Mask Option

### 5.5.2.5 COP Register

The COP register is shared with the LSB of an unimplemented user interrupt vector as shown in **Figure 5-3**. Reading this location returns whatever user data has been programmed at this location. Writing a zero to the COPR bit in this location clears the COP watchdog timer.



**Figure 5-3. COP Watchdog Timer Location**

### 5.5.3 Illegal Address

An illegal address reset is generated when the CPU attempts to fetch an instruction from I/O address space (\$0000 to \$001F).



## Section 6. Low-Power Modes

### 6.1 Contents

6.2	Introduction . . . . .	53
6.3	Stop Mode . . . . .	53
6.4	Stop Recovery . . . . .	54
6.5	Wait Mode . . . . .	54
6.6	Low-Power Reset . . . . .	55

### 6.2 Introduction

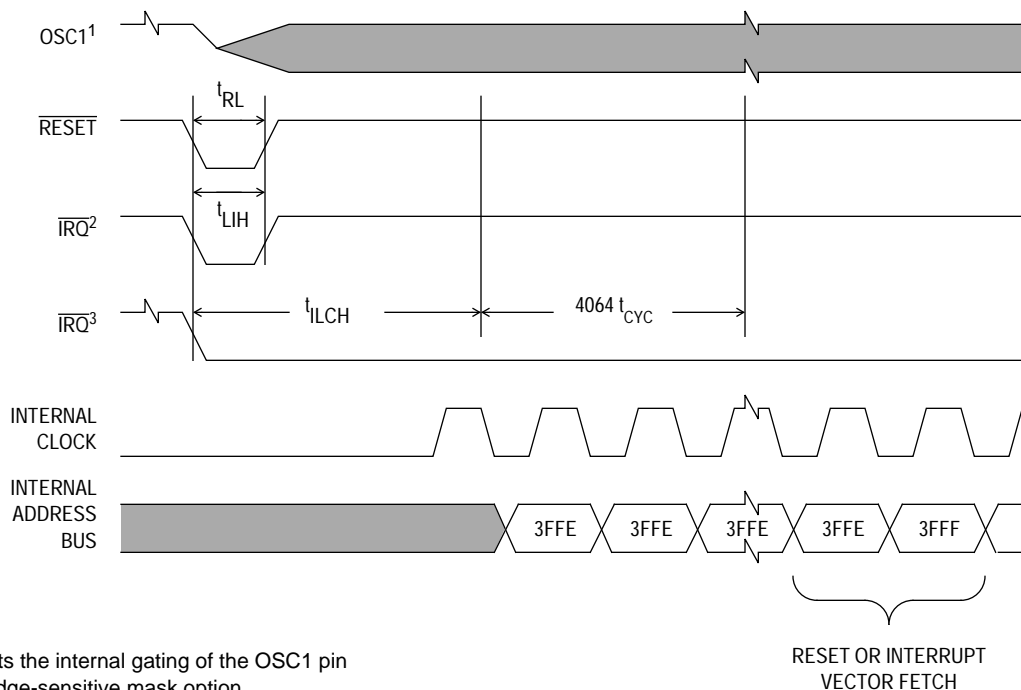
This section describes the low-power modes.

### 6.3 Stop Mode

The STOP instruction places the MCU in its lowest power-consumption mode. In stop mode, the internal oscillator is turned off, halting all internal processing, including timer operation.

During stop mode, the CTCSR (\$08) bits are altered to remove any pending timer interrupt request and to disable any further timer interrupts. The timer prescaler is cleared. The I bit in the CCR is cleared to enable external interrupts. All other registers and memory remain unaltered. All input/output lines remain unchanged.

**NOTE:** *The EIMSK bit is not cleared automatically by the execution of a STOP instruction. Care should be taken to clear this bit before entering stop mode.*



**NOTES:**

1. Represents the internal gating of the OSC1 pin
2.  $\overline{IRQ}$  pin edge-sensitive mask option
3.  $\overline{IRQ}$  pin level and edge-sensitive mask option

RESET OR INTERRUPT VECTOR FETCH

**Figure 6-1. Stop Recovery Timing Diagram**

## 6.4 Stop Recovery

The processor can be brought out of stop mode only by an external interrupt,  $\overline{LPRST}$ , or  $\overline{RESET}$ . Refer to [Figure 6-1](#).

**NOTE:** *If an external interrupt is pending when stop mode is entered, then stop mode will be exited immediately.*

## 6.5 Wait Mode

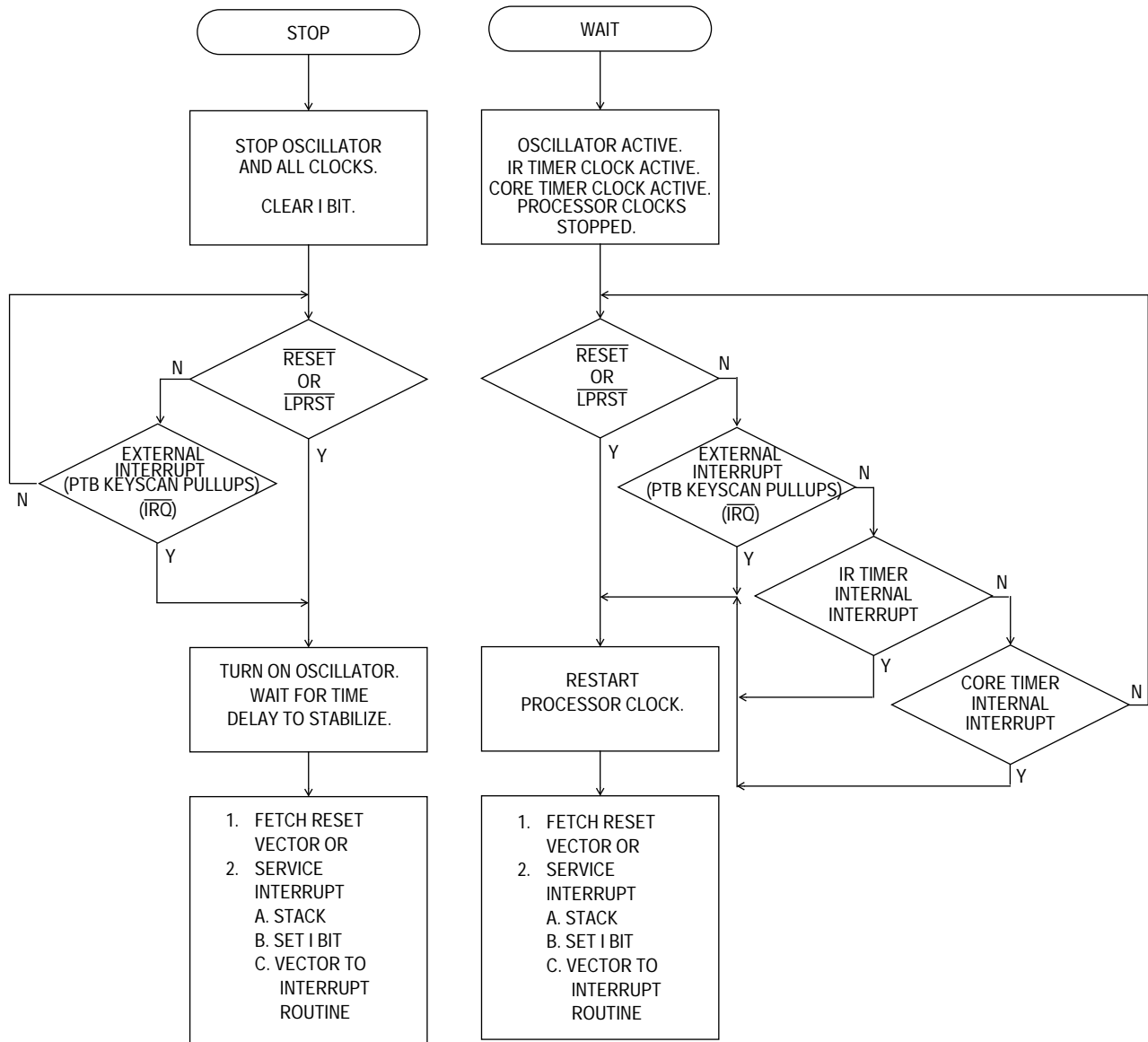
The WAIT instruction places the MCU in a low power-consumption mode, but wait mode consumes more power than stop mode. All CPU action is suspended, but the core timer, the oscillator, and any enabled module remain active. Any interrupt or reset will cause the MCU to exit wait mode. The user must shut off subsystems to reduce power

consumption. Wait current specifications assume CPU operation only and do not include current consumption by any other subsystems.

During wait mode, the I bit in the CCR is cleared to enable interrupts. All other registers, memory, and input/output lines remain in their previous states. The timer may be enabled to allow a periodic exit from wait mode.

## 6.6 Low-Power Reset

Low-power reset mode is entered when a logic 0 is detected on the  $\overline{\text{LPRST}}$  pin. When in this mode (as long as  $\overline{\text{LPRST}}$  is held low), the MCU is held in reset and all internal clocks are halted. Applying a logic 1 to  $\overline{\text{LPRST}}$  will cause the part to exit low-power reset mode and begin counting out the 4064-cycle oscillator stabilization period. Once this time has elapsed, the MCU will begin operation from the reset vectors (\$3FFE–\$3FFF).



**Figure 6-2. Stop/Wait Flowchart**



## Section 7. Parallel Input/Output (I/O)

### 7.1 Contents

7.2	Introduction . . . . .	57
7.3	Port A . . . . .	57
7.4	Port B . . . . .	58
7.5	Port C . . . . .	58
7.6	Input/Output Programming . . . . .	59

### 7.2 Introduction

In user mode, 20 lines (in 28-pin PDIP or SOIC) or 24 lines (in 44-lead PLCC) are arranged as three 8-bit I/O ports. These ports are programmable as either inputs or outputs under software control of the data direction registers.

**NOTE:** *To avoid a glitch on the output pins, write data to the I/O port data register before writing a one to the corresponding data direction register.*

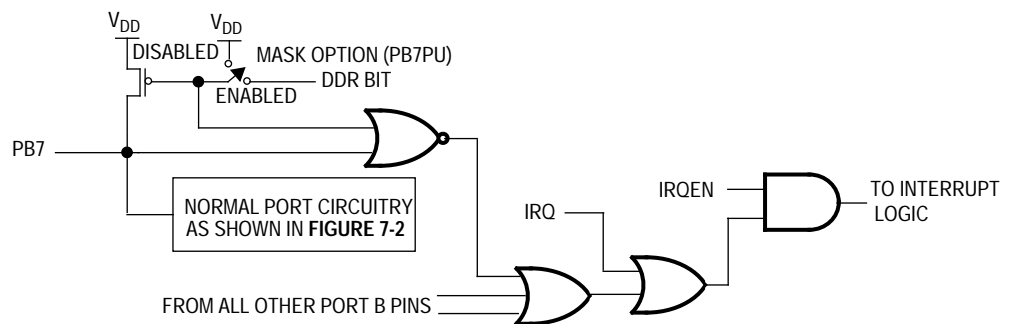
### 7.3 Port A

Port A is an 8-bit bidirectional port which does not share any of its pins with other subsystems. The port A data register is at \$0000 and the data direction register (DDR) is at \$0004. Reset does not affect the data register, but clears the data direction register, thereby returning the ports to inputs. Writing a one to a DDR bit sets the corresponding port bit to output mode.

## 7.4 Port B

Port B is an 8-bit bidirectional port which does not share any of its pins with other subsystems. The address of the port B data register is \$0001 and the data direction register (DDR) is at address \$0005. Reset does not affect the data register, but clears the data direction register, thereby returning the ports to inputs. Writing a one to a DDR bit sets the corresponding port bit to output mode. Each of the port B pins has a mask programmable pullup device that can be enabled. When the pullup device is enabled, this pin will become an interrupt pin also. The edge or edge and level sensitivity of the  $\overline{IRQ}$  pin also will pertain to the enabled port B pins. Care needs to be taken when using port B pins that have the pullup enabled. Before switching from an output to an input, the data should be preconditioned to a logic one or the I bit should be set in the condition code register to prevent an interrupt from occurring. The EIMSK bit in the CMT MCSR register can be used to mask port B keyscan and external interrupts (IRQ).

**NOTE:** *When a port B pin is configured as an output, it's corresponding keyscan interrupt is disabled, regardless of it's mask option.*



**Figure 7-1. Port B Pullup Option**

## 7.5 Port C

Port C is an 8-bit bidirectional port (PC0–PC7) which does not share any of its pins with other subsystems. The port C data register is at \$0003 and the data direction register (DDR) is at \$0006. Reset does not affect the data register, but clears the data direction register, thereby returning

the ports to inputs. Writing a one to a DDR bit sets the corresponding port bit to output mode. Port C pins PC4–PC7 are available only with the 44-lead PLCC package.

**NOTE:** *Only four bits of port C are bonded out in 28-pin packages for the MC68HC05RC16, although port C is truly an 8-bit port. Since pins PC4–PC7 are unbonded, software should include the code to set their respective data direction register locations to outputs to avoid floating inputs.*

## 7.6 Input/Output Programming

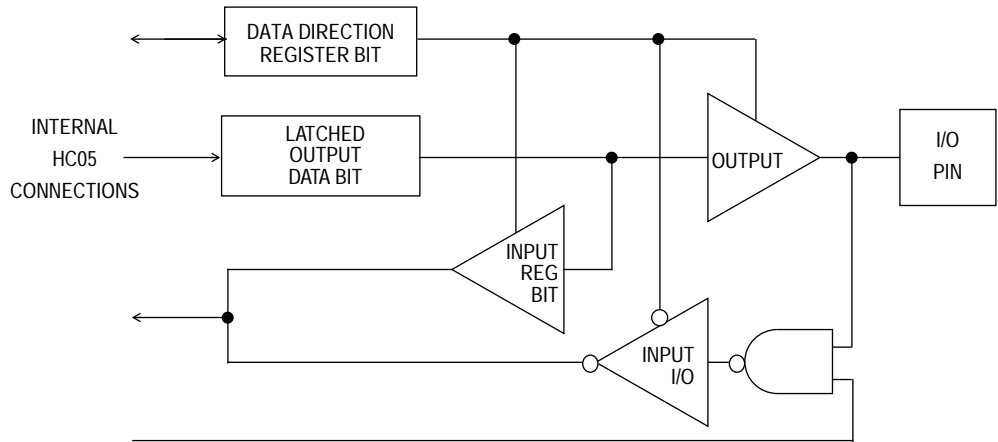
Port pins may be programmed as inputs or outputs under software control. The direction of the pins is determined by the state of the corresponding bit in the port data direction register (DDR). Each I/O port has an associated DDR. Any I/O port pin is configured as an output if its corresponding DDR bit is set to a logic 1. A pin is configured as an input if its corresponding DDR bit is cleared to a logic 0.

At power-on or reset, all DDRs are cleared, which configures all pins as inputs. The data direction registers are capable of being written to or read by the processor. During the programmed output state, a read of the data register actually reads the value of the output data latch and not the I/O pin.

**Table 7-1. I/O Pin Functions**

Access	DDR	I/O Pin Functions
Write	0	The I/O pin is in input mode. Data is written into the output data latch.
Write	1	Data is written into the output data latch and output to the I/O pin.
Read	0	The state of the I/O pin is read.
Read	1	The I/O pin is in an output mode. The output data latch is read.

# Parallel Input/Output (I/O)



**Figure 7-2. I/O Circuitry**

## Section 8. Core Timer

### 8.1 Contents

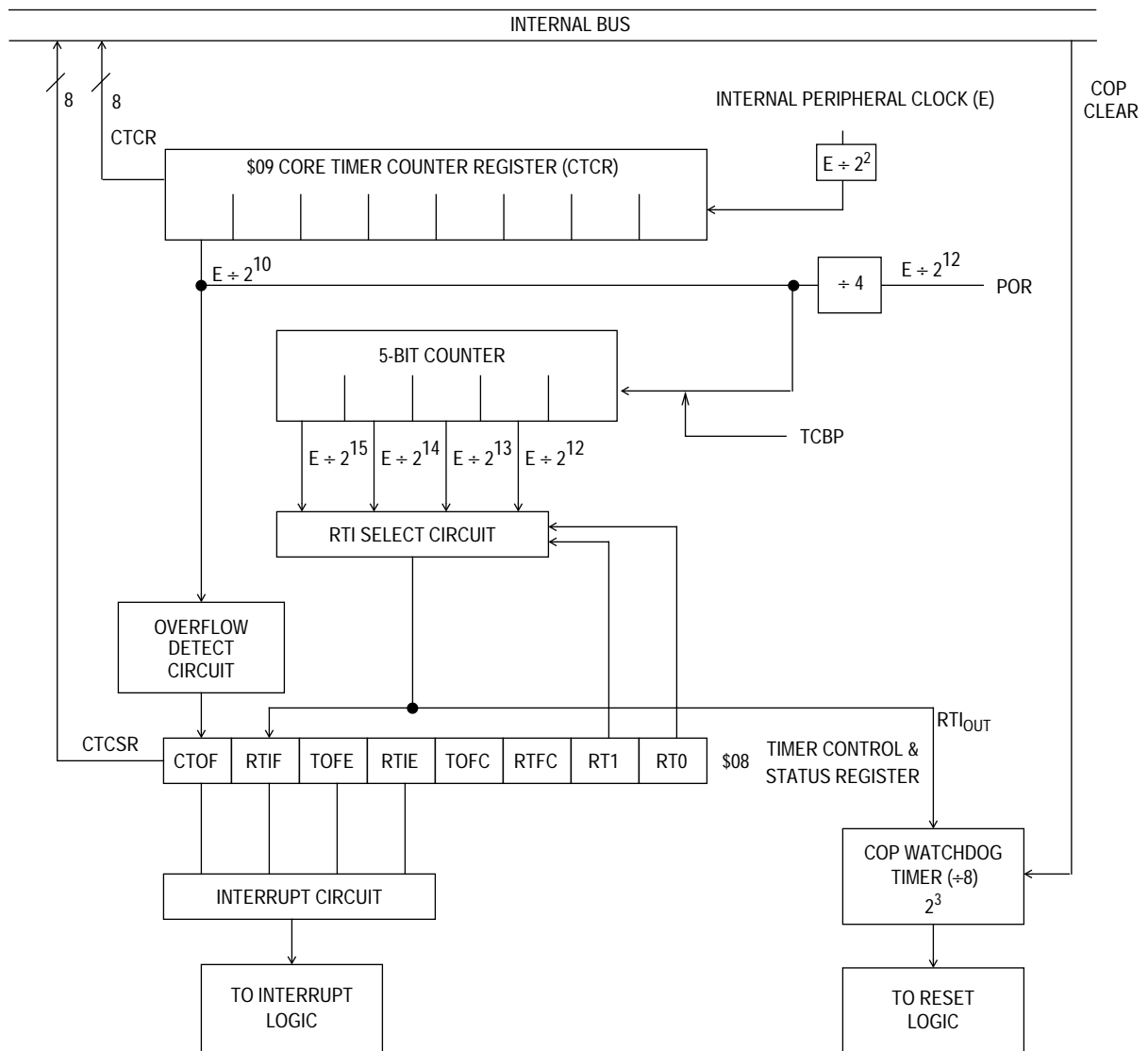
8.2	Introduction . . . . .	61
8.3	Core Timer Control and Status Register . . . . .	63
8.4	Core Timer Counter Register . . . . .	65
8.5	Computer Operating Properly (COP) Reset . . . . .	66
8.6	Timer During Wait Mode . . . . .	66

### 8.2 Introduction

The core timer for this device is a 14-stage multifunctional ripple counter. Features include timer overflow, power-on reset (POR), real-time interrupt (RTI), and COP watchdog timer.

As seen in **Figure 8-1**, the internal peripheral clock is divided by four, and then drives an 8-bit ripple counter. The value of this 8-bit ripple counter can be read by the CPU at any time by accessing the core timer counter register (CTCR) at address \$09. A timer overflow function is implemented on the last stage of this counter, giving a possible interrupt rate of the internal peripheral clock (E)/1024. This point is then followed by three more stages, with the resulting clock (E/4096) driving the real-time interrupt circuit (RTI). The RTI circuit consists of three divider stages with a one-of-four selector. The output of the RTI circuit is further divided by eight to drive the mask optional COP watchdog timer circuit. The RTI rate selector bits and the RTI and CTOF enable bits and flags are located in the timer control and status register at location \$08.

# Core Timer




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

### 8.3 Core Timer Control and Status Register

The CTCSR contains the timer interrupt flag, the timer interrupt enable bits, and the real-time interrupt rate select bits. **Figure 8-2** shows the value of each bit in the CTCSR when coming out of reset.

Address: \$08

Read:	CTOF	RTIF	TOFE	RTIE	0	0	RT1	RT0
Write:					TOFC	RTFC		
Reset:	0	0	0	0	0	0	1	1

 = Unimplemented

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

#### CTOF — Core Timer Overflow

CTOF is a read-only status bit set when the 8-bit ripple counter rolls over from \$FF to \$00. Clearing the CTOF is done by writing a one to TOFC. Writing to this bit has no effect. Reset clears CTOF.

#### RTIF — Real-Time Interrupt Flag

The real-time interrupt circuit consists of a 3-stage divider and a one-of-four selector. The clock frequency that drives the RTI circuit is  $E/2^{12}$  (or  $E \div 4096$  with three additional divider stages giving a maximum interrupt period of 16 milliseconds at a bus rate of 2.024 MHz. RTIF is a clearable, read-only status bit and is set when the output of the chosen (one-of-four selection) stage goes active. Clearing the RTIF is done by writing a one to RTFC. Writing has no effect on this bit. Reset clears RTIF.

#### TOFE — Timer Overflow Enable

When this bit is set, a CPU interrupt request is generated when the CTOF bit is set. Reset clears this bit.

#### RTIE — Real-Time Interrupt Enable

When this bit is set, a CPU interrupt request is generated when the RTIF bit is set. Reset clears this bit.

## TOFC — Timer Overflow Flag Clear

When a one is written to this bit, CTOF is cleared. Writing a zero has no effect on the CTOF bit. This bit always reads as zero.

## RTFC — Real-Time Interrupt Flag Clear

When a one is written to this bit, RTIF is cleared. Writing a zero has no effect on the RTIF bit. This bit always reads as zero.

## RT1–RT0 — Real-Time Interrupt Rate Select

These two bits select one of four taps from the real-time interrupt circuit. Refer to [Table 8-1](#). Reset sets these two bits which selects the lowest periodic rate and gives the maximum time in which to alter these bits if necessary. Care should be taken when altering RT0 and RT1 if the timeout period is imminent or uncertain. If the selected tap is modified during a cycle in which the counter is switching, an RTIF could be missed or an additional one could be generated. To avoid problems, the COP should be cleared before changing RTI taps.

**Table 8-1. RTI and COP Rates at 4.096 MHz Oscillator**

RTI RATE 2.048-MHz Bus		RT1:RT0	MINIMUM COP RATES 2.048-MHz Bus	
2 ms	$2^{12} \div E$	00	$(2^{15}-2^{12})/E$	14 ms
4 ms	$2^{13} \div E$	01	$(2^{16}-2^{13})/E$	28 ms
8 ms	$2^{14} \div E$	10	$(2^{17}-2^{14})/E$	56 ms
16 ms	$2^{15} \div E$	11	$(2^{18}-2^{15})/E$	112 ms




## 8.4 Core Timer Counter Register

The timer counter register is a read-only register that contains the current value of the 8-bit ripple counter at the beginning of the timer chain. This counter is clocked by the CPU clock (E/4) and can be used for various functions, including a software input capture. Extended time periods can be attained using the TOF function to increment a temporary RAM storage location, thereby simulating a 16-bit (or more) counter.

Address: \$09

Read:	D7	D6	D5	D4	D3	D2	D1	D0
Write:								
Reset:	0	0	0	0	0	0	1	1

 = Unimplemented

**Figure 8-3. Core Timer Counter Register (CTCR)**

The power-on cycle clears the entire counter chain and begins clocking the counter. After 4064 cycles, the power-on reset circuit is released, which again clears the counter chain and allows the device to come out of reset. At this point, if  $\overline{\text{RESET}}$  is not asserted, the timer starts counting up from zero and normal device operation begins. When  $\overline{\text{RESET}}$  is asserted any time during operation (other than POR and low-power reset), the counter chain is cleared.

## 8.5 Computer Operating Properly (COP) Reset

The COP watchdog timer function is implemented on this device by using the output of the RTI circuit and further dividing it by eight. The minimum COP reset rates are listed in [Table 8-1](#). If the COP circuit times out, an internal reset is generated and the normal reset vector is fetched. Preventing a COP timeout, or clearing the COP is accomplished by writing a zero to bit 0 of address \$3FF0. When the COP is cleared, only the final divide-by-eight stage (output of the RTI) is cleared.

If the COP watchdog timer is allowed to time out, an internal reset is generated to reset the MCU.

The COP remains enabled after execution of the WAIT instruction and all associated operations apply. If the STOP instruction is disabled, execution of STOP instruction causes the CPU to execute a WAIT instruction. In addition, the COP is prohibited from being held in reset. This prevents a device lock-up condition.

This COP's objective is to make it impossible for this device to become stuck or locked-up and to be sure the COP is able to rescue the part from any situation where it might entrap itself in abnormal or unintended behavior. This function is a mask option.

## 8.6 Timer During Wait Mode

The CPU clock halts during wait mode, but the timer remains active. If interrupts are enabled, a timer interrupt will cause the processor to exit wait mode. The COP is always enabled while in user mode.

## Section 9. Carrier Modulator Transmitter (CMT)

### 9.1 Contents

9.2	Introduction . . . . .	67
9.3	Overview . . . . .	68
9.4	Carrier Generator . . . . .	70
9.4.1	Time Counter . . . . .	71
9.4.2	Carrier Generator Data Registers (CHR1, CLR1, CHR2, and CLR2) . . . . .	72
9.5	Modulator . . . . .	74
9.5.1	Time Mode . . . . .	76
9.5.2	FSK Mode . . . . .	77
9.5.3	Extended Space Operation . . . . .	78
9.5.3.1	End Of Cycle (EOC) Interrupt . . . . .	79
9.5.3.2	Modulator Control and Status Register (MCSR) . . . . .	80
9.5.4	Modulator Period Data Registers (MDR1, MDR2, and MDR3) . . . . .	83

### 9.2 Introduction

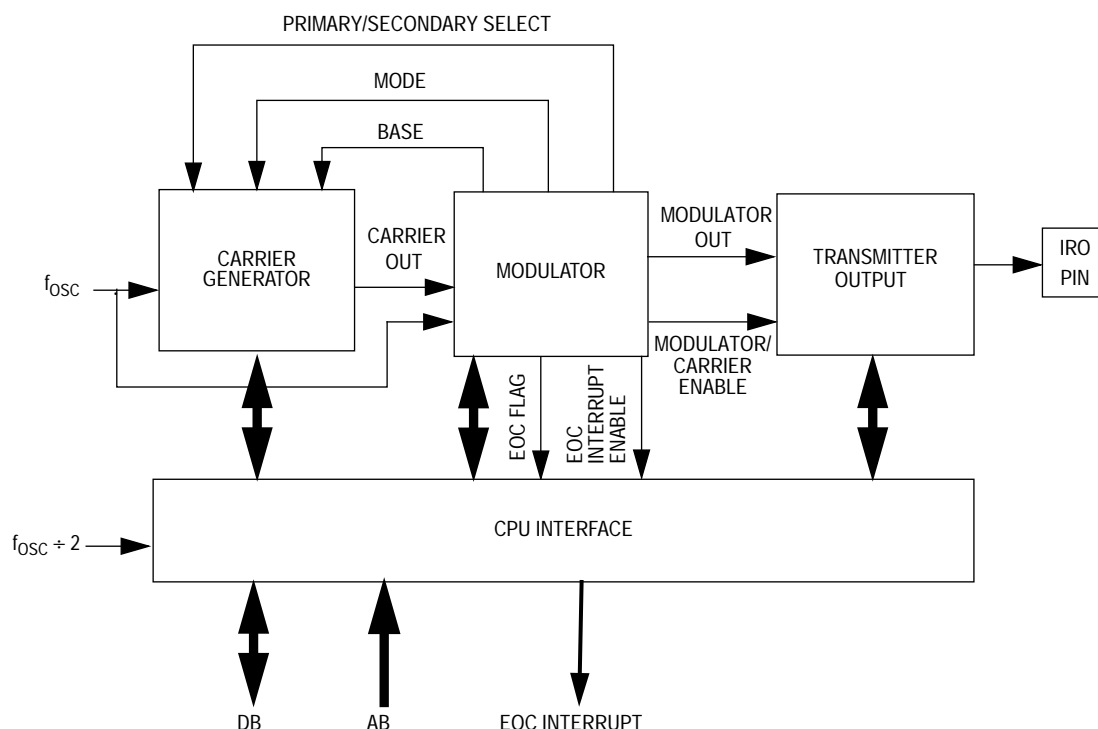
The carrier modulator transmitter (CMT) module provides a means to generate the protocol timing and carrier signals for a wide variety of encoding schemes. It incorporates hardware to off-load the critical and/or lengthy timing requirements associated with code generation from the CPU, releasing much of its bandwidth to handle other tasks such as code data generation, data decompression, or keyboard scanning. The CMT does not include dedicated hardware configurations for specific protocols, but is intended to be sufficiently programmable in its function to handle the timing requirements of most protocols with minimal CPU intervention. When disabled, certain CMT registers can be

used to change the state of the infrared out pin (IRO) directly. This feature allows for the generation of future protocols not readily producible by the current architecture.

### 9.3 Overview

The module consists of carrier generator, modulator, and transmitter output blocks. The block diagram is shown in [Figure 9-1](#).

The carrier generator has a resolution of 500 ns with a 2-MHz oscillator. The user may independently define the high and low times of the carrier signal to determine both period and duty cycle. The carrier generator can generate signals with periods between 1  $\mu$ s (1 MHz) and 64  $\mu$ s (15.6 kHz) in steps of 500 ns. The possible duty cycle options will depend upon the number of counts required to complete the carrier period. For example, a 400-kHz signal has a period of 2.5  $\mu$ s and will therefore require 5 x 500 ns counts to generate. These counts may be split between high and low times so the duty cycles available will be 20% (one high, four low), 40% (two high, three low), 60% (three high, two low) and 80% (four high, one low). For lower frequency signals with larger periods, higher resolution (as a percentage of the total period) duty cycles are possible. The carrier generator may select between two sets of high and low times. When operating in normal mode (subsequently referred to as time mode), just one set will be used. When operating in FSK (frequency shift key) mode, the generator will toggle between the two sets when instructed to do so by the modulator, allowing the user to dynamically switch between two carrier frequencies without CPU intervention. When the BASE bit in the modulator control and status register (MCSR) is set, the carrier output to the modulator is held high continuously to allow for the generation of baseband protocols. See [9.4 Carrier Generator](#).



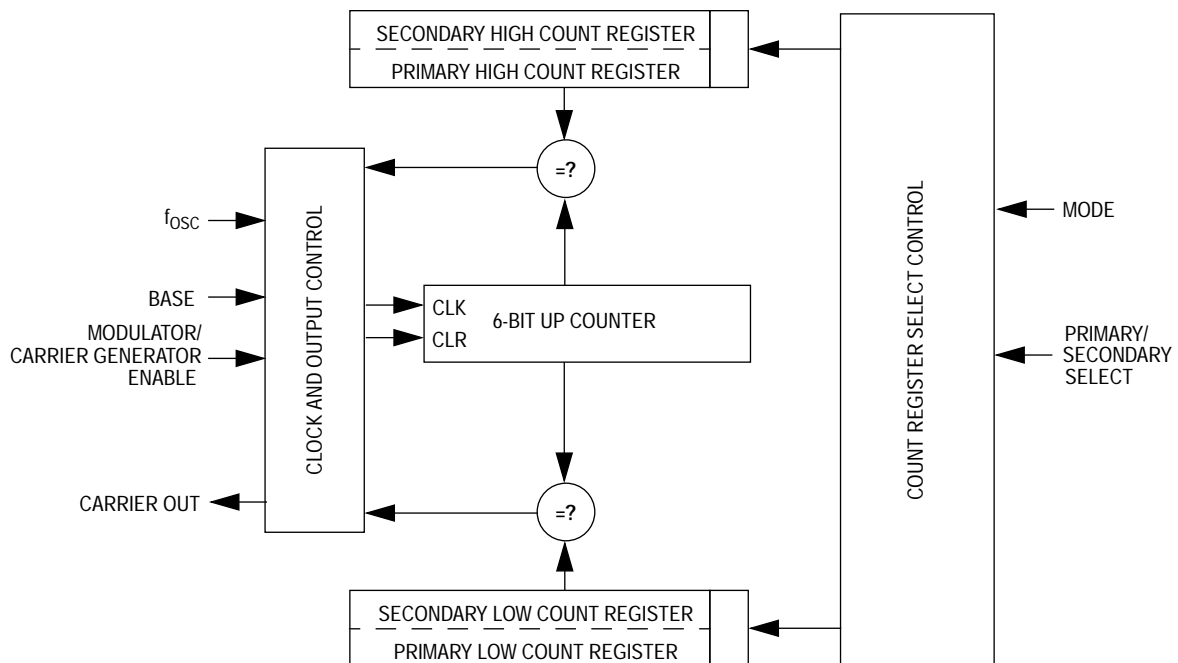
**Figure 9-1. Carrier Modulator Transmitter Module Block Diagram**

The modulator provides a simple method to control protocol timing. The modulator has a resolution of 4  $\mu$ s with a 2-MHz oscillator. It can count system clocks to provide real-time control or it can count carrier clocks for self-clocked protocols. It can either gate the carrier onto the modulator output (TIME), control the logic level of the modulator output (baseband) or directly route the carrier to the modulator output while providing a signal to switch the carrier generator between high/low time register buffers (FSK). See [9.5 Modulator](#).

The transmitter output block controls the state of the infrared out pin (IRO). The modulator output is gated on to the IRO pin when the modulator/carrier generator is enabled. Otherwise, the IRO pin is controlled by the state of the IRO latch, which is directly accessible to the CPU by means of bit 7 of the carrier generator data registers CHR1 and CLR1. The IRO latch can be written to on either edge of the internal bus clock ( $f_{osc}/2$ ), allowing for IR waveforms which have a resolution of twice the bus clock frequency ( $f_{osc}$ ). See [9.4.2 Carrier Generator Data Registers \(CHR1, CLR1, CHR2, and CLR2\)](#).

## 9.4 Carrier Generator

The carrier signal is generated by counting a predetermined number of input clocks (500 ns for a 2-MHz oscillator) for both the carrier high time and the carrier low time. The period is determined by the total number of clocks counted. The duty cycle is determined by the ratio of high time clocks to total clocks counted. The high and low time values are user programmable and are held in two registers. An alternate set of high/low count values is held in another set of registers to allow the generation of dual frequency FSK (frequency shift keying) protocols without CPU intervention. The MCGEN bit in the MCSR must be set and the BASE bit in the MCSR must be cleared to enable carrier generator clocks. The block diagram is shown in [Figure 9-2](#).



**Figure 9-2. Carrier Generator Block Diagram**

### 9.4.1 Time Counter

The high/low time counter is a 6-bit up counter. After each increment, the contents of the counter are compared with the appropriate high or low count value register. When this value is reached, the counter is reset and the compare is redirected to the other count value register. Assuming that the high time count compare register is currently active, a valid compare will cause the carrier output to be driven low. The counter will continue to increment and when reaching the value stored in the selected low count value register, it will be cleared and will cause the carrier output to be driven high. The cycle repeats, automatically generating a periodic signal which is directed to the modulator. The lowest frequency (maximum period) and highest frequency (minimum period) which can be generated are defined below.

$$f_{\min} = f_{\text{osc}} \div (2 \times (2^6 - 1)) \text{ Hz}$$

$$f_{\max} = f_{\text{osc}} \div (2 \times 1) \text{ Hz}$$

In the general case, the carrier generator output frequency is:

$$f_{\text{out}} = f_{\text{osc}} \div (\text{Highcount} + \text{Lowcount}) \text{ Hz}$$

Where:

$$0 < \text{Highcount} < 64 \text{ and}$$

$$0 < \text{Lowcount} < 64$$

**NOTE:** *These equations assume the DIV2 bit (bit 6) of the MCSR is clear. When the DIV2 bit is set, the carrier generator frequency will be half of what is shown in these equations.*

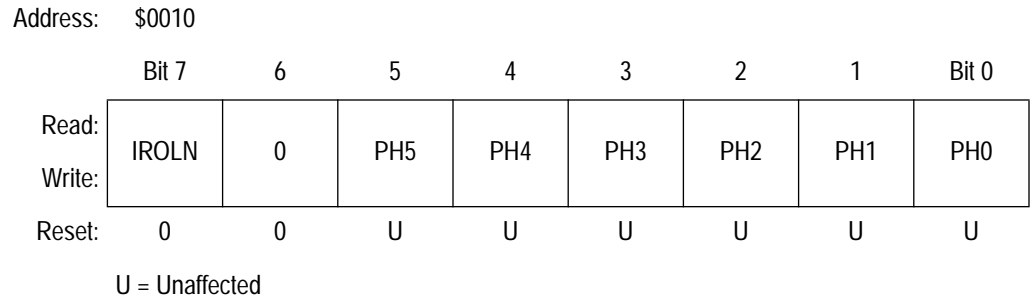
The duty cycle of the carrier signal is controlled by varying the ratio of high time to low + high time. As the input clock period is fixed, the duty cycle resolution will be proportional to the number of counts required to generate the desired carrier period.

$$\text{Duty Cycle} = \frac{\text{Highcount}}{\text{Highcount} + \text{Lowcount}}$$

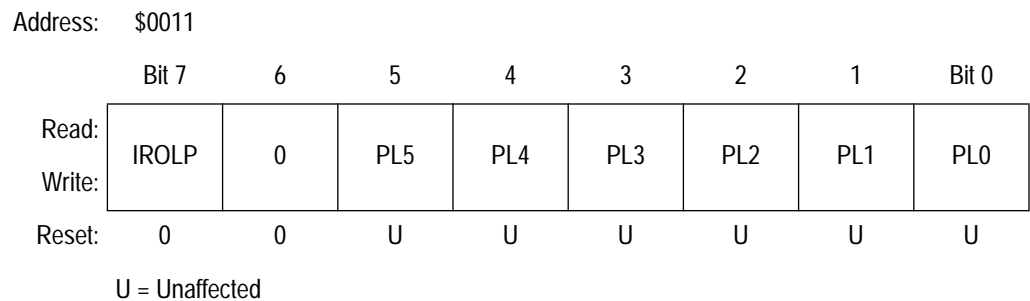
# Carrier Modulator Transmitter (CMT)

## 9.4.2 Carrier Generator Data Registers (CHR1, CLR1, CHR2, and CLR2)

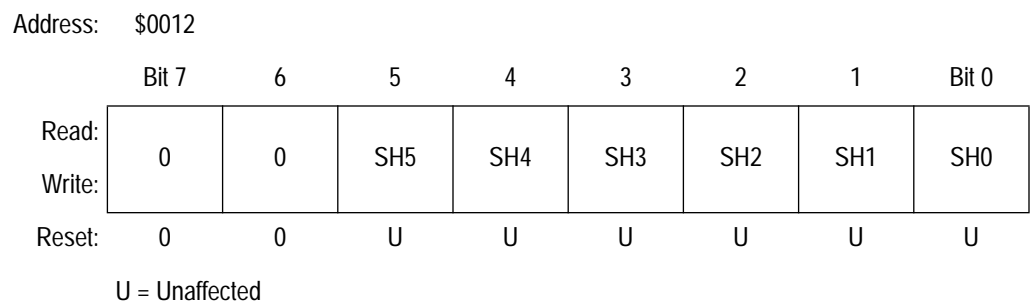
The carrier generator contains two, 7-bit data registers: primary high time (CHR1), primary low time (CLR1); and two, 6-bit data registers: secondary high time (CHR2) and secondary low time (CLR2). Bit 7 of CHR1 and CHR2 is used to read and write the IRO latch.



**Figure 9-3. Carrier Generator Data Register CHR1**



**Figure 9-4. Carrier Generator Data Register CLR1**



**Figure 9-5. Carrier Generator Data Register CHR2**



Address: \$0013

	Bit 7	6	5	4	3	2	1	Bit 0
Read:			SL5	SL4	SL3	SL2	SL1	SL0
Write:	0	0						
Reset:	0	0	U	U	U	U	U	U

U = Unaffected

**Figure 9-6. Carrier Generator Data Register CLR2**

**PH0–PH5 and PL0–PL5 — Primary Carrier High and Low Time Data Values**

When selected, these bits contain the number of input clocks required to generate the carrier high and low time periods. When operating in time mode (see [9.5.1 Time Mode](#)), this register pair is always selected. When operating in FSK mode (see [9.5.2 FSK Mode](#)), this register pair and the secondary register pair are alternately selected under control of the modulator. The primary carrier high and low time values are undefined out of reset. These bits must be written to nonzero values before the carrier generator is enabled to avoid spurious results.

**NOTE:** *Writing to CHR1 to update PH0–PH5 or to CLR1 to update PL0–PL5 will also update the IRO latch. When MCGEN (bit 0 in the MCSR) is clear, the IRO latch value appears on the IRO output pin. Care should be taken that bit 7 of the data to be written to CHR1 or CHL1 should contain the desired state of the IRO latch.*

**SH0–SH5 and SL0–SL5 — Secondary Carrier High and Low Time Data Values**

When selected, these bits contain the number of input clocks required to generate the carrier high and low time periods. When operating in time mode (see [9.5.1 Time Mode](#)), this register pair is never selected. When operating in FSK mode (see [9.5.2 FSK Mode](#)), this register pair and the secondary register pair are alternately selected under control

of the modulator. The secondary carrier high and low time values are undefined out of reset. These bits must be written to nonzero values before the carrier generator is enabled when operating in FSK mode.

### IROLN and IROLP — IRO Latch Control

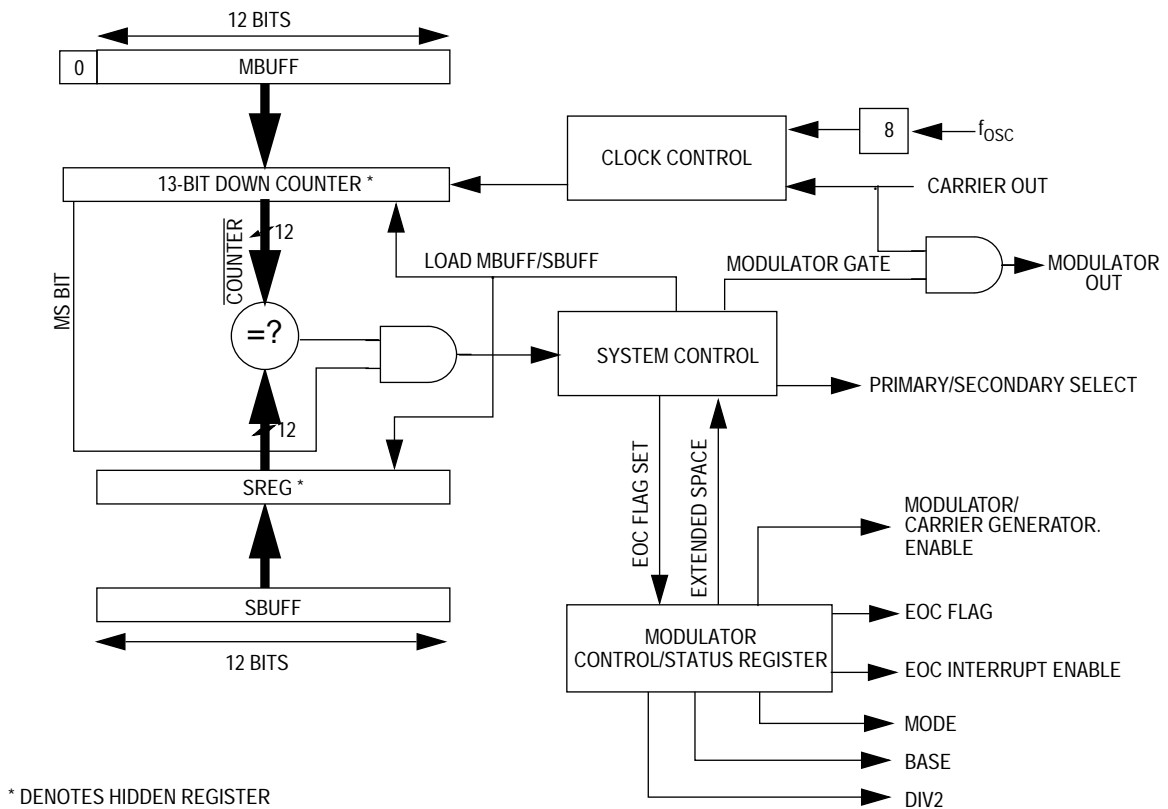
Reading IROLN or IROLP reads the state of the IRO latch. Writing IROLN updates the IRO latch with the data being written on the negative edge of the internal processor clock ( $f_{osc}/2$ ). Writing IROLP updates the IRO latch on the positive edge of the internal processor clock; for example, one  $f_{osc}$  period later. The IRO latch is clear out of reset.

**NOTE:** *Writing to CHR1 to update IROLN or to CLR1 to update IROLP will also update the primary carrier high and low data values. Care should be taken that bits 5–0 of the data to be written to CHR1 or CHL1 should contain the desired values for the primary carrier high or low data.*

## 9.5 Modulator

The modulator consists of a 12-bit down counter with underflow detection which is loaded from the modulation mark period from the mark buffer register, MBUFF. When this counter underflows, the modulator gate is closed and a 12-bit comparator is enabled which continually compares the logical complement of the contents of the (still) decrementing counter with the contents of the modulation space period register, SREG. When a match is obtained, the modulator control gate is opened again. Should  $SREG = 0$ , the match will be immediate and no space period will be generated (for instance, for FSK protocols which require successive bursts of different frequencies). When the match occurs, the counter is reloaded with the contents of MBUFF, SREG is reloaded with the contents of its buffer, SBUFF, and the cycle repeats. The MCGEN bit in the MCSR must be set to enable the modulator timer. The 12-bit MBUFF and SBUFF registers are accessed through three 8-bit modulator period registers, MDR1, MDR2, and MDR3.

The modulator can operate in two modes, time or FSK. In time mode the modulator counts clocks derived from the system oscillator and modulates a single-carrier frequency or no carrier (baseband). In FSK mode, the modulator counts carrier periods and instructs the carrier generator to alternate between two carrier frequencies whenever a modulation period (mark + space counts) expires.



**Figure 9-7. Modulator Block Diagram**

### 9.5.1 Time Mode

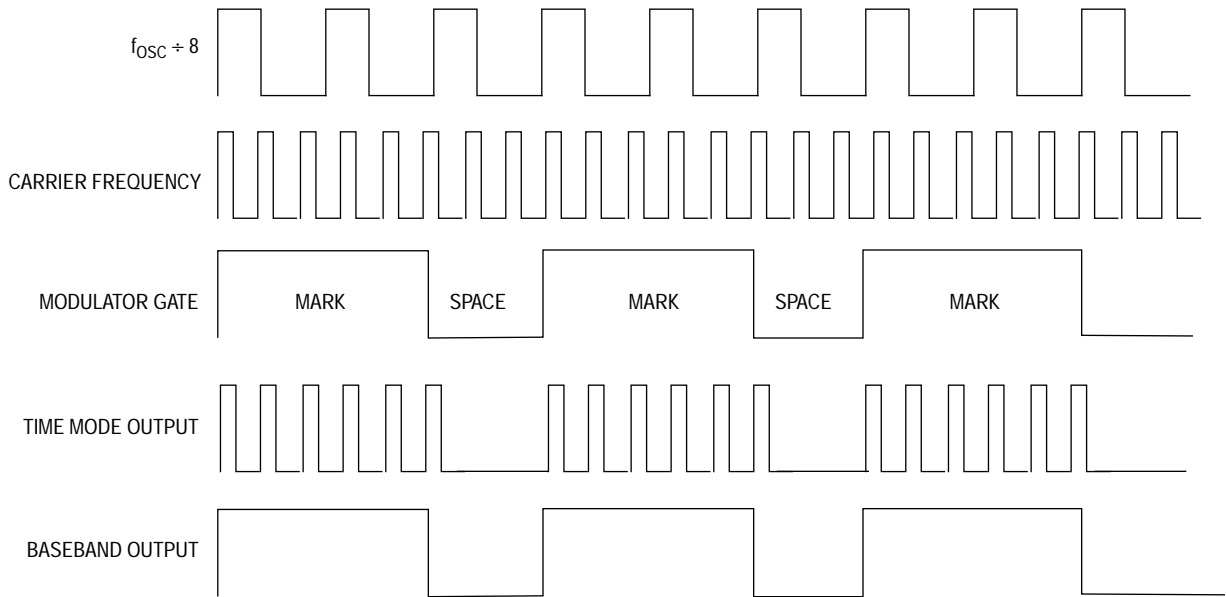
When the modulator operates in time mode, the modulation mark and space periods consist of zero or an integer number of  $f_{osc} \div 8$  clocks (= 250 kHz @ 2 MHz osc). This provides a modulator resolution of 4  $\mu$ s and a maximum mark and space periods of about 16 ms (each). However, to prevent carrier glitches which could affect carrier spectral purity, the modulator control gate and carrier clock are synchronized. The carrier signal is activated when the modulator gate opens. The modulator gate can only close when the carrier signal is low (the output logic level during space periods is low). If the carrier generator is in baseband mode (BASE bit in MCSR is high), the modulator output will be at a logic one for the duration of the mark period and at a logic zero for the duration of a space period. See [Figure 9-8](#).

The mark and space time equations are:

$$t_{\text{mark}} = \frac{(\text{MBUFF} + 1) \times 8}{f_{\text{osc}}} \text{secs}$$

$$t_{\text{space}} = \frac{\text{SBUFF} \times 8}{f_{\text{osc}}} \text{secs}$$

Setting the DIV2 bit in the MCSR will double mark and space times.



**Figure 9-8. CMT Operation in Time Mode**

### 9.5.2 FSK Mode

When the modulator operates in FSK mode, the modulation mark and space periods consist of an integer number of carrier clocks (space period can be zero). When the mark period expires, the space period is transparently started (as in time mode); however, in FSK mode the carrier switches between data registers in preparation for the next mark period. The carrier generator toggles between primary and secondary data register values whenever the modulator mark period expires. The space period provides an interpulse gap (no carrier), but if  $SBUFF = 0$ , then the modulator and carrier generator will switch between carrier frequencies without a gap or any carrier glitches (zero space).

Using timing data for carrier burst and interpulse gap length calculated by the CPU, FSK mode can automatically generate a phase-coherent, dual-frequency FSK signal with programmable burst and interburst gaps.

The mark and space time equations for FSK mode are:

$$t_{\text{mark}} = \frac{\text{MBUFF} + 1}{f_{\text{cg}}} \text{secs}$$

$$t_{\text{space}} = \frac{\text{SBUFF}}{f_{\text{cg}}} \text{secs}$$

Where  $f_{\text{cg}}$  is the frequency output from the carrier generator, setting the DIV2 bit in the MCSR will double mark and space times.

### 9.5.3 Extended Space Operation

In either time or FSK mode, the space period can be made longer than the maximum possible value of SBUFF. Setting the EXSPC bit in the MCSR will force the modulator to treat the next modulation period (beginning with the next load of MBUFF/SBUFF) as a space period equal in length to the mark and space counts combined. Subsequent modulation periods will consist entirely of these extended space periods with no mark periods. Clearing EXSPC will return the modulator to standard operation at the beginning of the next modulation period. To calculate the length of an extended space in time mode, use the equation:

$$t_{\text{exspace}} = \frac{((\text{SBUFF}_1) + (\text{MBUFF}_2 + 1 + \text{SBUFF}_2) + \dots + (\text{MBUFF}_n + 1 + \text{SBUFF}_n)) \times 8}{f_{\text{osc}}} \text{secs}$$

Where:

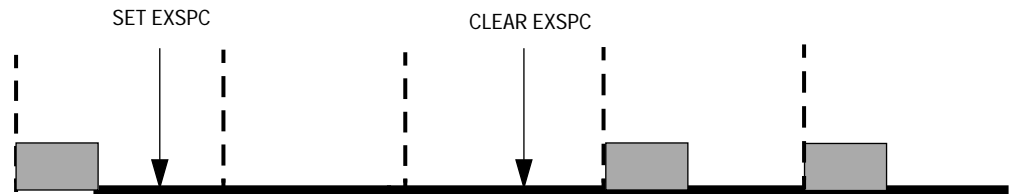
the subscripts 1, 2, ... n refer to the modulation periods that elapsed while the EXSPC bit was set.

Similarly, to calculate the length of an extended space in FSK mode, use the equation:

$$t_{\text{exspace}} = \frac{((\text{SBUFF}_1) + (\text{MBUFF}_2 + 1 + \text{SBUFF}_2) + \dots + (\text{MBUFF}_n + 1 + \text{SBUFF}_n))}{f_{\text{cg}}} \text{secs}$$

Where  $f_{cg}$  is the frequency output from the carrier generator. For an example of extended space operation, see [Figure 9-9](#).

**NOTE:** The EXSPC feature can be used to emulate a zero mark event.



**Figure 9-9. Extended Space Operation**

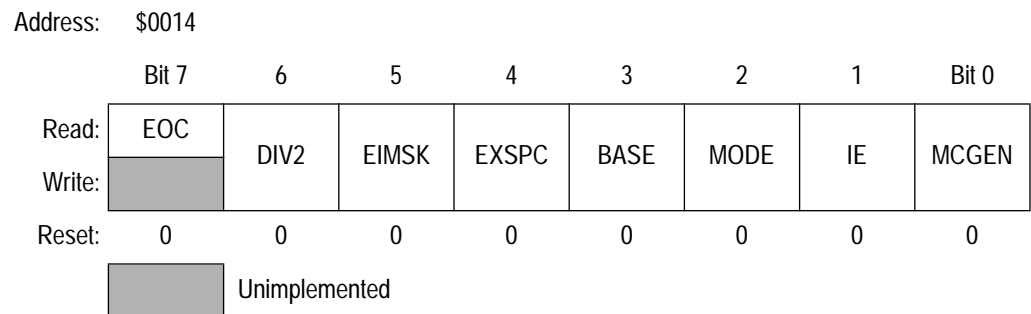
### 9.5.3.1 End Of Cycle (EOC) Interrupt

At the end of each cycle (when the counter is reloaded from MBUFF), the end of cycle (EOC) flag is set. If the interrupt enable bit was previously set, an interrupt also will be issued to the CPU. The EOC interrupt provides a means for the user to reload new mark/space values into the MBUFF and SBUFF registers. As the EOC interrupt is coincident with reloading the counter, MBUFF does not require additional buffering and may be updated with a new value for the next period from within the EOC interrupt service routine (ISR). To allow both mark and space period values to be updated from within the same ISR, SREG is buffered by SBUFF. The contents written to SBUFF are transferred to the active register SREG at the end of every cycle regardless of the state of the EOC flag. The EOC flag is cleared by a read of the modulator control and status register (MCSR) followed by an access of MDR2 or MDR3. The EOC flag must be cleared within the ISR to prevent another interrupt being generated after exiting the ISR. If the EOC interrupt is not being used ( $IE = 0$ ), the EOC flag need not be cleared.

# Carrier Modulator Transmitter (CMT)

## 9.5.3.2 Modulator Control and Status Register

The modulator control and status register (MCSR) contains the modulator and carrier generator enable (MCGEN), interrupt enable (IE), mode select (MODE), baseband enable (BASE), extended space (EXSPC), and external interrupt mask (EIMSK) control bits, divide-by-two prescaler (DIV2) bit, and the end of cycle (EOC) status bit.



**Figure 9-10. Modulator Control and Status Register (MCSR)**

### EOC — End Of Cycle Status Flag

EOC is set when a match occurs between the contents of the space period register, SREG, and the down counter. This is recognized as the end of the modulation cycle. At this time, the counter is initialized with the (possibly new) contents of the mark period buffer, MBUFF, and the space period register, SREG, is loaded with the (possibly new) contents of the space period buffer, SBUFF. This flag is cleared by a read of the MCSR followed by an access of MDR2 or MDR3. The EOC flag is cleared by reset.

1 = End of modulator cycle (counter = SBUFF) has occurred

0 = Current modulation cycle in progress

### DIV2 — Divide-by-two prescaler

The divide-by-two prescaler causes the CMT to be clocked at the bus rate when enabled; 2 x the bus rate when disabled ( $f_{osc}$ ). This bit is not double buffered and so should not be set during a transmission.

1 = Divide-by-two prescaler enabled

0 = Divide-by-two prescaler disabled



#### EIMSK — External Interrupt Mask

The external interrupt mask bit is used to mask IRQ and keyscan interrupts. This bit is cleared by reset.

- 1 = IRQ and keyscan interrupts masked
- 0 = IRQ and keyscan interrupts enabled

#### EXSPC — Extended Space Enable

For a description of the extended space enable bit, see [9.5.3 Extended Space Operation](#). This bit is cleared by reset.

- 1 = Extended space enabled
- 0 = Extended space disabled

#### BASE — Baseband Enable

When set, the BASE bit disables the carrier generator and forces the carrier output high for generation of baseband protocols. When BASE is clear, the carrier generator is enabled and the carrier output toggles at the frequency determined by values stored in the carrier data registers. See [9.5.1 Time Mode](#). This bit is cleared by reset. This bit is not double buffered and should not be written to during a transmission.

- 1 = Baseband enabled
- 0 = Baseband disabled

#### MODE — Mode Select

For a description of CMT operation in time mode, see [9.5.1 Time Mode](#). For a description of CMT operation in FSK mode, see [9.5.2 FSK Mode](#). This bit is cleared by reset. This bit is not double buffered and should not be written to during a transmission.

- 1 = CMT operates in FSK mode.
- 0 = CMT operates in time mode.

#### IE — Interrupt Enable

A CPU interrupt will be requested when EOC is set if IE was previously set. If IE is clear, EOC will not request a CPU interrupt.

- 1 = CPU interrupt enabled
- 0 = CPU interrupt disabled

### MCGEN — Modulator and Carrier Generator Enable

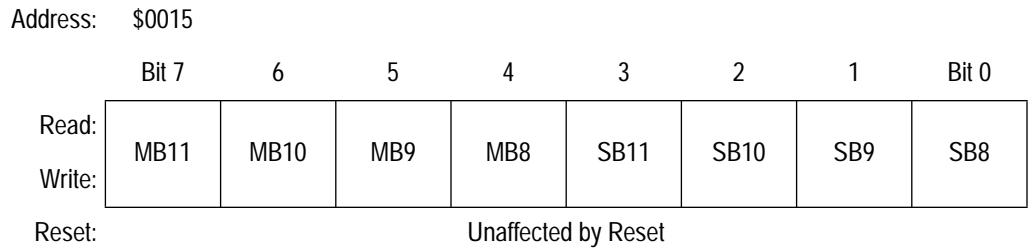
Setting MCGEN will initialize the carrier generator and modulator and will enable all clocks. Once enabled, the carrier generator and modulator will function continuously. When MCGEN is cleared, the current modulator cycle will be allowed to expire before all carrier and modulator clocks are disabled (to save power) and the modulator output is forced low. The user should initialize all data and control registers before enabling the system to prevent spurious operation. This bit is cleared by reset.

1 = Modulator and carrier generator enabled

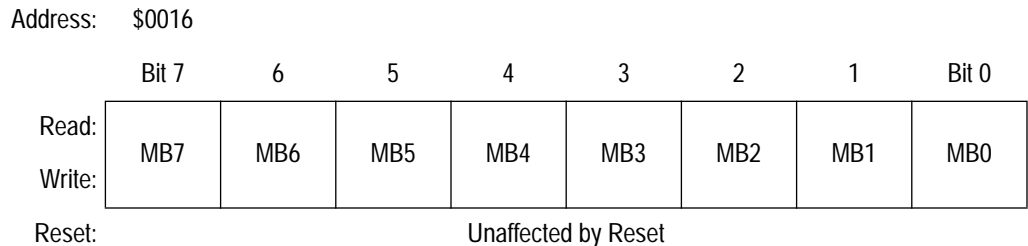
0 = Modulator and carrier generator disabled

### 9.5.4 Modulator Period Data Registers (MDR1, MDR2, and MDR3)

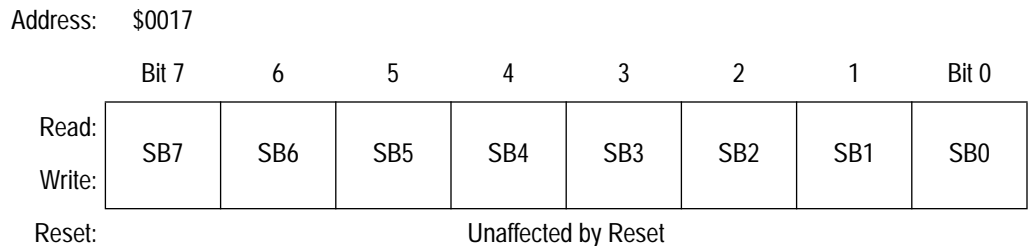
The 12-bit MBUFF and SBUFF registers are accessed through three 8-bit registers: MDR1, MDR2, and MDR3. MDR2 and MDR3 contain the least significant eight bits of MBUFF and SBUFF respectively. MDR1 contains the two most significant nibbles of MBUFF and SBUFF. In many applications, periods greater than those obtained by eight bits will not be required. Dividing the registers in this manner allows the user to clear MDR1 and generate 8-bit periods with just two data writes.



**Figure 9-11. Modulator Period Data Register MDR1**



**Figure 9-12. Modulator Period Data Register MDR2**



**Figure 9-13. Modulator Period Data Register MDR3**



## Section 10. Instruction Set

### 10.1 Contents

10.2	Introduction . . . . .	86
10.3	Addressing Modes . . . . .	86
10.3.1	Inherent . . . . .	87
10.3.2	Immediate . . . . .	87
10.3.3	Direct . . . . .	87
10.3.4	Extended . . . . .	87
10.3.5	Indexed, No Offset . . . . .	88
10.3.6	Indexed, 8-Bit Offset . . . . .	88
10.3.7	Indexed, 16-Bit Offset . . . . .	88
10.3.8	Relative . . . . .	89
10.4	Instruction Types . . . . .	89
10.4.1	Register/Memory Instructions . . . . .	90
10.4.2	Read-Modify-Write Instructions . . . . .	91
10.4.3	Jump/Branch Instructions . . . . .	92
10.4.4	Bit Manipulation Instructions . . . . .	94
10.4.5	Control Instructions . . . . .	95
10.5	Instruction Set Summary . . . . .	96

## 10.2 Introduction

The MCU instruction set has 62 instructions and uses eight addressing modes. The instructions include all those of the M146805 CMOS Family plus one more: the unsigned multiply (MUL) instruction. The MUL instruction allows unsigned multiplication of the contents of the accumulator (A) and the index register (X). The high-order product is stored in the index register, and the low-order product is stored in the accumulator.

## 10.3 Addressing Modes

The CPU uses eight addressing modes for flexibility in accessing data. The addressing modes provide eight different ways for the CPU to find the data required to execute an instruction. The eight addressing modes are:

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

### 10.3.1 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 operand address and are one byte long.

### 10.3.2 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 operand address and are two bytes long. The opcode is the first byte, and the immediate data value is the second byte.

### 10.3.3 Direct

Direct instructions can access any of the first 256 memory locations 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.

### 10.3.4 Extended

Extended instructions use three bytes and can 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.

## 10.3.5 Indexed, No Offset

Indexed instructions with no offset are 1-byte instructions that can access data with variable addresses within the first 256 memory locations. The index register contains the low byte of the effective 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.

## 10.3.6 Indexed, 8-Bit Offset

Indexed, 8-bit offset instructions are 2-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 effective address of the operand. These instructions can access locations \$0000–\$01FE.

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

## 10.3.7 Indexed, 16-Bit Offset

Indexed, 16-bit offset instructions are 3-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 effective 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.

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

As with direct and extended addressing, the Motorola assembler determines the shortest form of indexed addressing.



### 10.3.8 Relative

Relative addressing is only for branch instructions. If the branch condition is true, the CPU finds the effective 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.

## 10.4 Instruction Types

The MCU instructions fall into the following five categories:

- Register/Memory Instructions
- Read-Modify-Write Instructions
- Jump/Branch Instructions
- Bit Manipulation Instructions
- Control Instructions

## 10.4.1 Register/Memory Instructions

These instructions operate on CPU registers and memory locations. Most of them use two operands. One operand is in either the accumulator or the index register. The CPU finds the other operand in memory.

**Table 10-1. Register/Memory Instructions**

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

## 10.4.2 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.

**NOTE:** *Do not use read-modify-write operations on write-only registers.*

**Table 10-2. Read-Modify-Write Instructions**

Instruction	Mnemonic
Arithmetic Shift Left (Same as LSL)	ASL
Arithmetic Shift Right	ASR
Bit Clear	BCLR <sup>(1)</sup>
Bit Set	BSET <sup>(1)</sup>
Clear Register	CLR
Complement (One's Complement)	COM
Decrement	DEC
Increment	INC
Logical Shift Left (Same as ASL)	LSL
Logical Shift Right	LSR
Negate (Two's Complement)	NEG
Rotate Left through Carry Bit	ROL
Rotate Right through Carry Bit	ROR
Test for Negative or Zero	TST <sup>(2)</sup>

1. Unlike other read-modify-write instructions, BCLR and BSET use only direct addressing.
2. TST is an exception to the read-modify-write sequence because it does not write a replacement value.

## 10.4.3 Jump/Branch Instructions

Jump instructions allow the CPU to interrupt the normal sequence of the program counter. The unconditional jump instruction (JMP) and the jump-to-subroutine instruction (JSR) have no register operand. 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.

The BRCLR and BRSET instructions cause a branch based on the state of any readable bit in the first 256 memory locations. These 3-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 effective 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 10-3. Jump and Branch Instructions**

<b>Instruction</b>	<b>Mnemonic</b>
Branch if Carry Bit Clear	BCC
Branch if Carry Bit Set	BCS
Branch if Equal	BEQ
Branch if Half-Carry Bit Clear	BHCC
Branch if Half-Carry Bit Set	BHCS
Branch if Higher	BHI
Branch if Higher or Same	BHS
Branch if $\overline{\text{IRQ}}$ Pin High	BIH
Branch if $\overline{\text{IRQ}}$ Pin 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 Never	BRN
Branch if Bit Set	BRSET
Branch to Subroutine	BSR
Unconditional Jump	JMP
Jump to Subroutine	JSR

## 10.4.4 Bit Manipulation Instructions

The CPU can set or clear any writable bit in the first 256 bytes of memory, which includes I/O registers and on-chip RAM locations. The CPU can also test and branch based on the state of any bit in any of the first 256 memory locations.

**Table 10-4. Bit Manipulation Instructions**

<b>Instruction</b>	<b>Mnemonic</b>
Bit Clear	BCLR
Branch if Bit Clear	BRCLR
Branch if Bit Set	BRSET
Bit Set	BSET

### 10.4.5 Control Instructions

These instructions act on CPU registers and control CPU operation during program execution.

**Table 10-5. Control Instructions**

Instruction	Mnemonic
Clear Carry Bit	CLC
Clear Interrupt Mask	CLI
No Operation	NOP
Reset Stack Pointer	RSP
Return from Interrupt	RTI
Return from Subroutine	RTS
Set Carry Bit	SEC
Set Interrupt Mask	SEI
Stop Oscillator and Enable $\overline{\text{IRQ}}$ Pin	STOP
Software Interrupt	SWI
Transfer Accumulator to Index Register	TAX
Transfer Index Register to Accumulator	TXA
Stop CPU Clock and Enable Interrupts	WAIT

## 10.5 Instruction Set Summary

**Table 10-6. Instruction Set Summary**

Source Form	Operation	Description	Effect on CCR					Address Mode	Opcode	Operand	Cycles
			H	I	N	Z	C				
ADC #opr ADC opr ADC opr ADC opr,X ADC opr,X ADC ,X	Add with Carry	$A \leftarrow (A) + (M) + (C)$	↕x	—	↕x	↕x	↕x	IMM DIR EXT IX2 IX1 IX	A9 B9 C9 D9 E9 F9	ii dd hh ll ee ff ff	2 3 4 5 4 3
ADD #opr ADD opr ADD opr ADD opr,X ADD opr,X ADD ,X	Add without Carry	$A \leftarrow (A) + (M)$	↕x	—	↕x	↕	↕	IMM DIR EXT IX2 IX1 IX	AB BB CB DB EB FB	ii dd hh ll ee ff ff	2 3 4 5 4 3
AND #opr AND opr AND opr AND opr,X AND opr,X AND ,X	Logical AND	$A \leftarrow (A) \wedge (M)$	—	—	↕x	↕	—	IMM DIR EXT IX2 IX1 IX	A4 B4 C4 D4 E4 F4	ii dd hh ll ee ff ff	2 3 4 5 4 3
ASL opr ASLA ASLX ASL opr,X ASL ,X	Arithmetic Shift Left (Same as LSL)		—	—	↕x	↕	↕	DIR INH INH IX1 IX	38 48 58 68 78	dd ff	5 3 3 6 5
ASR opr ASRA ASRX ASR opr,X ASR ,X	Arithmetic Shift Right		—	—	↕x	↕	↕	DIR INH INH IX1 IX	37 47 57 67 77	dd ff	5 3 3 6 5
BCC rel	Branch if Carry Bit Clear	$PC \leftarrow (PC) + 2 + rel ? C = 0$	—	—	—	—	—	REL	24	rr	3
BCLR n opr	Clear Bit n	$M_n \leftarrow 0$	—	—	—	—	—	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	11 13 15 17 19 1B 1D 1F	dd dd dd dd dd dd dd dd	5 5 5 5 5 5 5 5
BCS rel	Branch if Carry Bit Set (Same as BLO)	$PC \leftarrow (PC) + 2 + rel ? C = 1$	—	—	—	—	—	REL	25	rr	3
BEQ rel	Branch if Equal	$PC \leftarrow (PC) + 2 + rel ? Z = 1$	—	—	—	—	—	REL	27	rr	3
BHCC rel	Branch if Half-Carry Bit Clear	$PC \leftarrow (PC) + 2 + rel ? H = 0$	—	—	—	—	—	REL	28	rr	3
BHCS rel	Branch if Half-Carry Bit Set	$PC \leftarrow (PC) + 2 + rel ? H = 1$	—	—	—	—	—	REL	29	rr	3
BHI rel	Branch if Higher	$PC \leftarrow (PC) + 2 + rel ? C \vee Z = 0$	—	—	—	—	—	REL	22	rr	3
BHS rel	Branch if Higher or Same	$PC \leftarrow (PC) + 2 + rel ? C = 0$	—	—	—	—	—	REL	24	rr	3



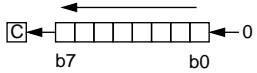
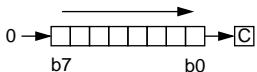
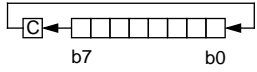
Table 10-6. Instruction Set Summary (Continued)

Source Form	Operation	Description	Effect on CCR					Address Mode	Opcode	Operand	Cycles
			H	I	N	Z	C				
BIH <i>rel</i>	Branch if IRQ Pin High	$PC \leftarrow (PC) + 2 + rel ? IRQ = 1$	—	—	—	—	—	REL	2F	rr	3
BIL <i>rel</i>	Branch if IRQ Pin Low	$PC \leftarrow (PC) + 2 + rel ? IRQ = 0$	—	—	—	—	—	REL	2E	rr	3
BIT # <i>opr</i> BIT <i>opr</i> BIT <i>opr</i> BIT <i>opr</i> , <i>X</i> BIT <i>opr</i> , <i>X</i> BIT , <i>X</i>	Bit Test Accumulator with Memory Byte	$(A) \wedge (M)$	—	—	↕x	↕	—	IMM DIR EXT IX2 IX1 IX	A5 B5 C5 D5 E5 F5	ii dd hh ll ee ff ff	2 3 4 5 4 3
BLO <i>rel</i>	Branch if Lower (Same as BCS)	$PC \leftarrow (PC) + 2 + rel ? C = 1$	—	—	—	—	—	REL	25	rr	3
BLS <i>rel</i>	Branch if Lower or Same	$PC \leftarrow (PC) + 2 + rel ? C \vee Z = 1$	—	—	—	—	—	REL	23	rr	3
BMC <i>rel</i>	Branch if Interrupt Mask Clear	$PC \leftarrow (PC) + 2 + rel ? I = 0$	—	—	—	—	—	REL	2C	rr	3
BMI <i>rel</i>	Branch if Minus	$PC \leftarrow (PC) + 2 + rel ? N = 1$	—	—	—	—	—	REL	2B	rr	3
BMS <i>rel</i>	Branch if Interrupt Mask Set	$PC \leftarrow (PC) + 2 + rel ? I = 1$	—	—	—	—	—	REL	2D	rr	3
BNE <i>rel</i>	Branch if Not Equal	$PC \leftarrow (PC) + 2 + rel ? Z = 0$	—	—	—	—	—	REL	26	rr	3
BPL <i>rel</i>	Branch if Plus	$PC \leftarrow (PC) + 2 + rel ? N = 0$	—	—	—	—	—	REL	2A	rr	3
BRA <i>rel</i>	Branch Always	$PC \leftarrow (PC) + 2 + rel ? 1 = 1$	—	—	—	—	—	REL	20	rr	3
BRCLR <i>n opr rel</i>	Branch if Bit n Clear	$PC \leftarrow (PC) + 2 + rel ? Mn = 0$	—	—	—	—	↕x	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	01 03 05 07 09 0B 0D 0F	dd rr dd rr dd rr dd rr dd rr dd rr dd rr dd rr	5 5 5 5 5 5 5 5
BRN <i>rel</i>	Branch Never	$PC \leftarrow (PC) + 2 + rel ? 1 = 0$	—	—	—	—	—	REL	21	rr	3
BRSET <i>n opr rel</i>	Branch if Bit n Set	$PC \leftarrow (PC) + 2 + rel ? Mn = 1$	—	—	—	—	⊗	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	00 02 04 06 08 0A 0C 0E	dd rr dd rr dd rr dd rr dd rr dd rr dd rr dd rr	5 5 5 5 5 5 5 5
BSET <i>n opr</i>	Set Bit n	$Mn \leftarrow 1$	—	—	—	—	—	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	10 12 14 16 18 1A 1C 1E	dd dd dd dd dd dd dd dd	5 5 5 5 5 5 5 5
BSR <i>rel</i>	Branch to Subroutine	$PC \leftarrow (PC) + 2$ ; push (PCL) $SP \leftarrow (SP) - 1$ ; push (PCH) $SP \leftarrow (SP) - 1$ $PC \leftarrow (PC) + rel$	—	—	—	—	—	REL	AD	rr	6
CLC	Clear Carry Bit	$C \leftarrow 0$	—	—	—	—	0	INH	98		2
CLI	Clear Interrupt Mask	$I \leftarrow 0$	—	0	—	—	—	INH	9A		2

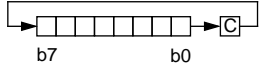
## Table 10-6. Instruction Set Summary (Continued)

Source Form	Operation	Description	Effect on CCR					Address Mode	Opcode	Operand	Cycles
			H	I	N	Z	C				
CLR <i>opr</i> CLRA CLR X CLR <i>opr</i> ,X CLR ,X	Clear Byte	M ← \$00 A ← \$00 X ← \$00 M ← \$00 M ← \$00	—	—	0	1	—	DIR INH INH IX1 IX	3F 4F 5F 6F 7F	dd ff	5 3 3 6 5
CMP # <i>opr</i> CMP <i>opr</i> CMP <i>opr</i> CMP <i>opr</i> ,X CMP <i>opr</i> ,X CMP ,X	Compare Accumulator with Memory Byte	(A) – (M)	—	—	↑x	↑	↑	IMM DIR EXT IX2 IX1 IX	A1 B1 C1 D1 E1 F1	ii dd hh ll ee ff ff	2 3 4 5 4 3
COM <i>opr</i> COMA COM X COM <i>opr</i> ,X COM ,X	Complement Byte (One's Complement)	M ← (M̄) = \$FF – (M) A ← (Ā) = \$FF – (A) X ← (X̄) = \$FF – (X) M ← (M̄) = \$FF – (M) M ← (M̄) = \$FF – (M)	—	—	↑x	↑x	1	DIR INH INH IX1 IX	33 43 53 63 73	dd ff	5 3 3 6 5
CPX # <i>opr</i> CPX <i>opr</i> CPX <i>opr</i> CPX <i>opr</i> ,X CPX <i>opr</i> ,X CPX ,X	Compare Index Register with Memory Byte	(X) – (M)	—	—	↑x	⊗	⊗	IMM DIR EXT IX2 IX1 IX	A3 B3 C3 D3 E3 F3	ii dd hh ll ee ff ff	2 3 4 5 4 3
DEC <i>opr</i> DECA DEC X DEC <i>opr</i> ,X DEC ,X	Decrement Byte	M ← (M) – 1 A ← (A) – 1 X ← (X) – 1 M ← (M) – 1 M ← (M) – 1	—	—	↑x	↑x	—	DIR INH INH IX1 IX	3A 4A 5A 6A 7A	dd ff	5 3 3 6 5
EOR # <i>opr</i> EOR <i>opr</i> EOR <i>opr</i> EOR <i>opr</i> ,X EOR <i>opr</i> ,X EOR ,X	EXCLUSIVE OR Accumulator with Memory Byte	A ← (A) ⊕ (M)	—	—	↑x	↑	—	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 <i>opr</i> INCA INC X INC <i>opr</i> ,X INC ,X	Increment Byte	M ← (M) + 1 A ← (A) + 1 X ← (X) + 1 M ← (M) + 1 M ← (M) + 1	—	—	↑x	↑x	—	DIR INH INH IX1 IX	3C 4C 5C 6C 7C	dd ff	5 3 3 6 5
JMP <i>opr</i> JMP <i>opr</i> JMP <i>opr</i> ,X JMP <i>opr</i> ,X JMP ,X	Unconditional Jump	PC ← Jump Address	—	—	—	—	—	DIR EXT IX2 IX1 IX	BC CC DC EC FC	dd hh ll ee ff ff	2 3 4 3 2

Table 10-6. Instruction Set Summary (Continued)

Source Form	Operation	Description	Effect on CCR					Address Mode	Opcode	Operand	Cycles
			H	I	N	Z	C				
JSR <i>opr</i> JSR <i>opr</i> JSR <i>opr,X</i> JSR <i>opr,X</i> JSR ,X	Jump to Subroutine	PC ← (PC) + n (n = 1, 2, or 3) Push (PCL); SP ← (SP) – 1 Push (PCH); SP ← (SP) – 1 PC ← Effective Address	—	—	—	—	—	DIR EXT IX2 IX1 IX	BD CD DD ED FD	dd hh ll ee ff ff	5 6 7 6 5
LDA # <i>opr</i> LDA <i>opr</i> LDA <i>opr</i> LDA <i>opr,X</i> LDA <i>opr,X</i> LDA ,X	Load Accumulator with Memory Byte	A ← (M)	—	—	↕x	↕	—	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 # <i>opr</i> LDX <i>opr</i> LDX <i>opr</i> LDX <i>opr,X</i> LDX <i>opr,X</i> LDX ,X	Load Index Register with Memory Byte	X ← (M)	—	—	↕x	↕x	—	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 <i>opr</i> LSLA LSLX LSL <i>opr,X</i> LSL ,X	Logical Shift Left (Same as ASL)		—	—	↕x	↕	↕	DIR INH INH IX1 IX	38 48 58 68 78	dd ff	5 3 3 6 5
LSR <i>opr</i> LSRA LSRX LSR <i>opr,X</i> LSR ,X	Logical Shift Right		—	—	0	↕	↕	DIR INH INH IX1 IX	34 44 54 64 74	dd ff	5 3 3 6 5
MUL	Unsigned Multiply	X : A ← (X) × (A)	0	—	—	—	0	INH	42		11
NEG <i>opr</i> NEGA NEGX NEG <i>opr,X</i> NEG ,X	Negate Byte (Two's Complement)	M ← -(M) = \$00 - (M) A ← -(A) = \$00 - (A) X ← -(X) = \$00 - (X) M ← -(M) = \$00 - (M) M ← -(M) = \$00 - (M)	—	—	↕x	↕	↕	DIR INH INH IX1 IX	30 40 50 60 70	dd ff	5 3 3 6 5
NOP	No Operation		—	—	—	—	—	INH	9D		2
ORA # <i>opr</i> ORA <i>opr</i> ORA <i>opr</i> ORA <i>opr,X</i> ORA <i>opr,X</i> ORA ,X	Logical OR Accumulator with Memory	A ← (A) ∨ (M)	—	—	↕x	↕	—	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 <i>opr</i> ROLA ROLX ROL <i>opr,X</i> ROL ,X	Rotate Byte Left through Carry Bit		—	—	↕x	↕	↕	DIR INH INH IX1 IX	39 49 59 69 79	dd ff	5 3 3 6 5

## Table 10-6. Instruction Set Summary (Continued)

Source Form	Operation	Description	Effect on CCR					Address Mode	Opcode	Operand	Cycles
			H	I	N	Z	C				
ROR <i>opr</i> RORA RORX ROR <i>opr,X</i> ROR ,X	Rotate Byte Right through Carry Bit		—	—	↕x	↕	↕	DIR INH INH IX1 IX	36 46 56 66 76	dd  ff	5 3 3 6 5
RSP	Reset Stack Pointer	SP ← \$00FF	—	—	—	—	—	INH	9C		2
RTI	Return from Interrupt	SP ← (SP) + 1; Pull (CCR) SP ← (SP) + 1; Pull (A) SP ← (SP) + 1; Pull (X) SP ← (SP) + 1; Pull (PCH) SP ← (SP) + 1; Pull (PCL)	↕x	↕	↕	↕	↕	INH	80		9
RTS	Return from Subroutine	SP ← (SP) + 1; Pull (PCH) SP ← (SP) + 1; Pull (PCL)	—	—	—	—	—	INH	81		6
SBC # <i>opr</i> SBC <i>opr</i> SBC <i>opr</i> SBC <i>opr,X</i> SBC <i>opr,X</i> SBC ,X	Subtract Memory Byte and Carry Bit from Accumulator	A ← (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 ← 1	—	—	—	—	1	INH	99		2
SEI	Set Interrupt Mask	I ← 1	—	1	—	—	—	INH	9B		2
STA <i>opr</i> STA <i>opr</i> STA <i>opr,X</i> STA <i>opr,X</i> STA ,X	Store Accumulator in Memory	M ← (A)	—	—	↕x	↕	—	DIR EXT IX2 IX1 IX	B7 C7 D7 E7 F7	dd hh ll ee ff ff	4 5 6 5 4
STOP	Stop Oscillator and Enable IRQ Pin		—	0	—	—	—	INH	8E		2
STX <i>opr</i> STX <i>opr</i> STX <i>opr,X</i> STX <i>opr,X</i> STX ,X	Store Index Register In Memory	M ← (X)	—	—	↕x	↕	—	DIR EXT IX2 IX1 IX	BF CF DF EF FF	dd hh ll ee ff ff	4 5 6 5 4
SUB # <i>opr</i> SUB <i>opr</i> SUB <i>opr</i> SUB <i>opr,X</i> SUB <i>opr,X</i> SUB ,X	Subtract Memory Byte from Accumulator	A ← (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 ← (PC) + 1; Push (PCL) SP ← (SP) – 1; Push (PCH) SP ← (SP) – 1; Push (X) SP ← (SP) – 1; Push (A) SP ← (SP) – 1; Push (CCR) SP ← (SP) – 1; I ← 1 PCH ← Interrupt Vector High Byte PCL ← Interrupt Vector Low Byte	—	1	—	—	—	INH	83		10
TAX	Transfer Accumulator to Index Register	X ← (A)	—	—	—	—	—	INH	97		2

**Table 10-6. Instruction Set Summary (Continued)**

Source Form	Operation	Description	Effect on CCR					Address Mode	Opcode	Operand	Cycles	
			H	I	N	Z	C					
TST <i>opr</i> TSTA TSTX TST <i>opr</i> ,X TST ,X	Test Memory Byte for Negative or Zero	(M) – \$00						DIR	3D	dd	4	
							INH	4D		3		
					—	—	↓	↓	INH	5D		3
									IX1	6D	ff	5
									IX	7D		4
TXA	Transfer Index Register to Accumulator	A ← (X)	—	—	—	—	—	INH	9F		2	
WAIT	Stop CPU Clock and Enable Interrupts		—	0x	—	—	—	INH	8F		2	

- |          |   |            |                                      |
|----------|---|------------|--------------------------------------|
| A        | Accumulator   | <i>opr</i> | Operand (one or two bytes)           |
| C        | Carry/borrow flag   | PC         | Program counter                      |
| CCR      | Condition code register   | PCH        | Program counter high byte            |
| dd       | Direct address of operand   | PCL        | Program counter low byte             |
| dd rr    | Direct address of operand and relative offset of branch instruction | REL        | Relative addressing mode             |
| DIR      | Direct addressing mode  | <i>rel</i> | Relative program counter offset byte |
| ee ff    | High and low bytes of offset in indexed, 16-bit offset addressing   | rr         | Relative program counter offset byte |
| EXT      | Extended addressing mode  | SP         | Stack pointer                        |
| ff       | Offset byte in indexed, 8-bit offset addressing                     | X          | Index register                       |
| H        | Half-carry flag   | Z          | Zero flag                            |
| hh ll    | High and low bytes of operand address in extended addressing        | #          | Immediate value                      |
| I        | Interrupt mask  | ^          | Logical AND                          |
| ii       | Immediate operand byte  | ∨          | Logical OR                           |
| IMM      | Immediate addressing mode   | ⊕          | Logical EXCLUSIVE OR                 |
| INH      | Inherent addressing mode  | ( )        | Contents of                          |
| IX       | Indexed, no offset addressing mode                                  | -( )       | Negation (two's complement)          |
| IX1      | Indexed, 8-bit offset addressing mode                               | ←          | Loaded with                          |
| IX2      | Indexed, 16-bit offset addressing mode                              | ?          | If                                   |
| M        | Memory location   | :          | Concatenated with                    |
| N        | Negative flag   | ↓          | Set or cleared                       |
| <i>n</i> | Any bit   | —          | Not affected                         |

**Table 10-7. Opcode Map**

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

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

LSB of Opcode in Hexadecimal

MSB LSB	<b>0</b>
<b>0</b>	BRSET <sup>5</sup> <sub>3</sub> DIR

MSB of Opcode in Hexadecimal  
 Number of Cycles  
 Opcode Mnemonic  
 Number of Bytes/Addressing Mode

## Section 11. Electrical Specifications

### 11.1 Contents

11.2	Introduction . . . . .	103
11.3	Maximum Ratings . . . . .	104
11.4	Operating Range . . . . .	105
11.5	Thermal Characteristics . . . . .	105
11.6	DC Electrical Characteristics (5.0 Vdc). . . . .	106
11.7	DC Electrical Characteristics (2.2 Vdc). . . . .	107
11.8	Control Timing (5.0 Vdc and 2.2 Vdc). . . . .	109

### 11.2 Introduction

This section contains the electrical and timing specifications.

## 11.3 Maximum Ratings

Maximum ratings are the extreme limits to which the MCU can be exposed without permanently damaging it.

The MCU contains circuitry to protect the inputs against damage from high static voltages; however, do not apply voltages higher than those shown in the table below. 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 voltage level, either  $V_{SS}$  or  $V_{DD}$ .

Rating	Symbol	Value	Unit
Supply Voltage	$V_{DD}$	-0.3 to +7.0	V
Burn-In Mode ( $\overline{IRQ}$ Pin Only)	$V_{IN}$	$V_{SS} - 0.3$ to $2 \times V_{DD} + 0.3$	V
Current Drain Per Pin Excluding $V_{DD}$ and $V_{SS}$	I	25	mA
Operating Junction Temperature	$T_J$	+150	°C
Storage Temperature Range	$T_{stg}$	-65 to +150	°C

**NOTE:** *This device is not guaranteed to operate properly at the maximum ratings. Refer to [11.6 DC Electrical Characteristics \(5.0 Vdc\)](#) and [11.7 DC Electrical Characteristics \(2.2 Vdc\)](#) for guaranteed operating conditions.*



## 11.4 Operating Range

Characteristic	Symbol	Value	Unit
Operating Temperature Range MC68HC05RC16 (Standard)	$T_A$	$T_L$ to $T_H$ 0 to +70	$^{\circ}\text{C}$

## 11.5 Thermal Characteristics

Characteristic	Symbol	Value	Unit
Thermal Resistance	$\theta_{JA}$		$^{\circ}\text{C}/\text{W}$
Plastic Dual In-Line Package		60	
Small Outline Intergrated Circuit Package		60	
Plastic Leaded Chip Carrier Package		60	

# Electrical Specifications

## 11.6 DC Electrical Characteristics (5.0 Vdc)

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage $I_{LOAD} = 10.0 \mu A$ $I_{LOAD} = -10.0 \mu A$	$V_{OL}$ $V_{OH}$	— $V_{DD} - 0.1$	— —	0.1 —	V
Output High Voltage ( $I_{LOAD} = -2.0$ mA) Port A, Port B, Port C (1–7) ( $I_{LOAD} = -20$ mA) IRO ( $I_{LOAD} = -4.0$ mA) Port C (Bit 0)	$V_{OH}$	$V_{DD} - 0.8$ $V_{DD} - 0.8$ $V_{DD} - 0.8$	$V_{DD} - 0.2$ $V_{DD} - 0.2$ $V_{DD} - 0.2$	— — —	V
Output Low Voltage ( $I_{LOAD} = 3.0$ mA) Port A, Port B, Port C (1–7) ( $I_{LOAD} = 25.0$ mA) IRO ( $I_{LOAD} = 20.0$ mA) Port C (Bit 0)	$V_{OL}$	— — —	0.2 0.2 0.2	0.4 0.4 0.4	V
Input High Voltage Port A, Port B, Port C, $\overline{IRQ}$ , $\overline{RESET}$ , $\overline{LPRST}$ , OSC1	$V_{IH}$	$0.7 \times V_{DD}$	—	$V_{DD}$	V
Input Low Voltage Port A, Port B, Port C, $\overline{IRQ}$ , $\overline{RESET}$ , $\overline{LPRST}$ , OSC1	$V_{IL}$	$V_{SS}$	—	$0.2 \times V_{DD}$	V
Supply Current (see Notes) Run Wait Stop 25 °C 0 to +70 °C	$I_{DD}$	— — — —	2.3 0.5 0.3 0.3	4.0 1.0 10.0 20.0	mA mA $\mu A$ $\mu A$
I/O Ports Hi-Z Leakage Current Port A, Port B, Port C	$I_{OZ}$	-10	—	10	$\mu A$
Input Current $\overline{RESET}$ , $\overline{LPRST}$ , $\overline{IRQ}$ , OSC1 PB0–PB7 with Pullups Enabled ( $V_{IN} = 0.2 \times V_{DD}$ ) <sup>8</sup> PB0–PB7 with Pullups Enabled ( $V_{IN} = 0.7 \times V_{DD}$ )	$I_{IN}$	-1 -100 -50	— -330 -120	1 -700 -300	$\mu A$
Capacitance Ports (as Input or Output) $\overline{RESET}$ , $\overline{LPRST}$ , $\overline{IRQ}$	$C_{OUT}$ $C_{INT}$	— —	— —	12 8	pF

### NOTES:

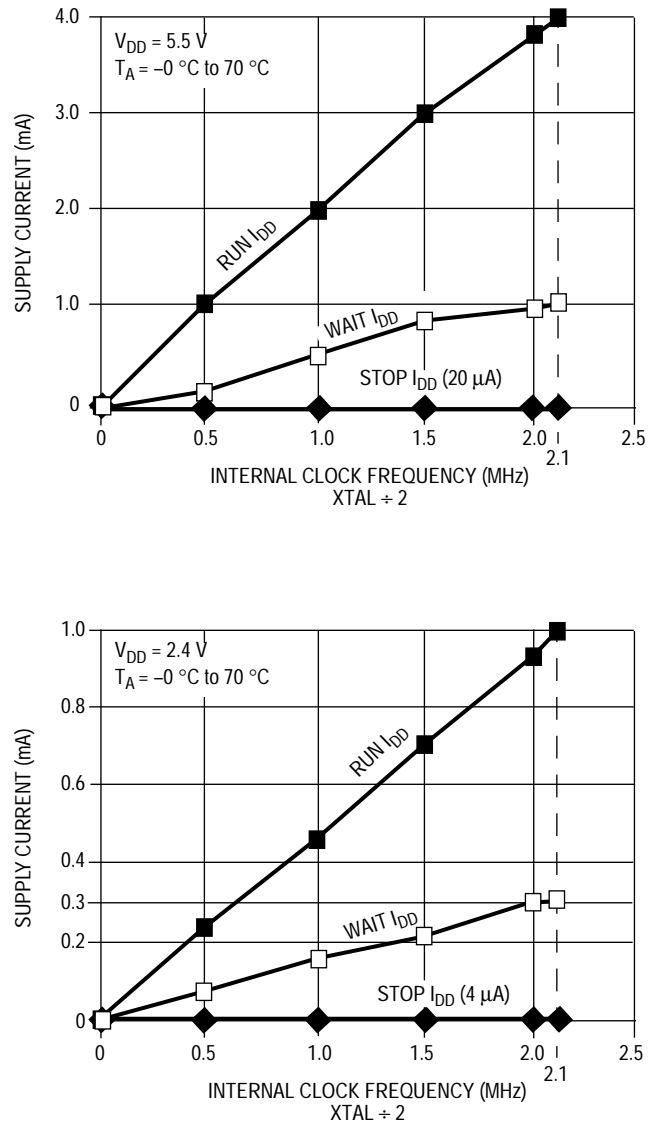
- $V_{DD} = 5.0$  Vdc  $\pm 10\%$ ,  $V_{SS} = 0$  Vdc,  $T_A = 0$  °C to +70 °C, unless otherwise noted
- Typical values at midpoint of voltage range, 25 °C only, represent average measurements.
- Wait  $I_{DD}$ : only core timer active
- Run (Operating)  $I_{DD}$ , wait  $I_{DD}$ : Measured using external square wave clock source ( $f_{OSC} = 4.2$  MHz); all inputs 0.2 V from rail; no dc loads; less than 50 pF on all outputs;  $C_L = 20$  pF on OSC2
- Wait, Stop  $I_{DD}$ : Port A and port C configured as inputs, port B configured as outputs,  $V_{IL} = 0.2$  V,  $V_{IH} = V_{DD} - 0.2$  V
- Stop  $I_{DD}$  is measured with OSC1 =  $V_{SS}$ .
- Wait  $I_{DD}$  is affected linearly by the OSC2 capacitance.
- Pullups are designed to be capable of pulling to  $V_{IH}$  within 1  $\mu s$  for a 100 pF, 4-k $\Omega$  load.

## 11.7 DC Electrical Characteristics (2.2 Vdc)

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage $I_{LOAD} = 10.0 \mu A$ $I_{LOAD} = -10.0 \mu A$	$V_{OL}$ $V_{OH}$	— $V_{DD} - 0.1$	— —	0.1 —	V
Output High Voltage ( $I_{LOAD} = -0.6$ mA) Port A, Port B, Port C (1–7) ( $I_{LOAD} = -8.0$ mA) IRO ( $I_{LOAD} = -1.2$ mA) Port C (Bit 0)	$V_{OH}$	$V_{DD} - 0.3$ $V_{DD} - 0.3$ $V_{DD} - 0.3$	$V_{DD} - 0.1$ $V_{DD} - 0.1$ $V_{DD} - 0.1$	— — —	V
Output Low Voltage ( $I_{LOAD} = 1.0$ mA) Port A, Port B, Port C (1–7) ( $I_{LOAD} = 8.0$ mA) IRO ( $I_{LOAD} = 7.0$ mA) Port C (Bit 0)	$V_{OL}$	— — —	0.1 0.1 0.1	0.3 0.3 0.3	V
Input High Voltage Port A, Port B, Port C, $\overline{IRQ}$ , $\overline{RESET}$ , $\overline{LPRST}$ , OSC1	$V_{IH}$	$0.7 \times V_{DD}$	—	$V_{DD}$	V
Input Low Voltage Port A, Port B, Port C, $\overline{IRQ}$ , $\overline{RESET}$ , $\overline{LPRST}$ , OSC1	$V_{IL}$	$V_{SS}$	—	$0.4 \times V_{DD}$	V
Supply Current (see Notes) Run Wait Stop 25 °C 0 to +70 °C	$I_{DD}$	— — — —	0.3 0.15 0.1 0.1	1.0 0.3 1.0 4.0	mA mA $\mu A$ $\mu A$
I/O Ports Hi-Z Leakage Current Port A, Port B, Port C	$I_{OZ}$	–4	—	4	$\mu A$
Input Current $\overline{RESET}$ , $\overline{LPRST}$ , $\overline{IRQ}$ , OSC1 PB0–PB7 with Pullups Enabled ( $V_{IN} = 0.4 \times V_{DD}$ ) <sup>8</sup> PB0–PB7 with Pullups Enabled ( $V_{IN} = 0.7 \times V_{DD}$ )	$I_{IN}$	–0.4 –25 –15	— –50 –34	0.4 –105 –65	$\mu A$
Capacitance Ports (as Input or Output) $\overline{RESET}$ , $\overline{LPRST}$ , $\overline{IRQ}$	$C_{OUT}$ $C_{INT}$	— —	— —	12 8	pF

NOTES:

- $V_{DD} = 2.2 \text{ Vdc} \pm 10\%$ ,  $V_{SS} = 0 \text{ Vdc}$ ,  $T_A = 0 \text{ }^\circ\text{C}$  to  $+70 \text{ }^\circ\text{C}$ , unless otherwise noted
- Typical values at midpoint of voltage range, 25 °C only, represent average measurements.
- Wait  $I_{DD}$ : only core timer active
- Run (Operating)  $I_{DD}$ , wait  $I_{DD}$ : Measured using external square wave clock source ( $f_{osc} = 4.2 \text{ MHz}$ ); all inputs 0.2 V from rail; no dc loads; less than 50 pF on all outputs;  $C_L = 20 \text{ pF}$  on OSC2
- Wait, Stop  $I_{DD}$ : Port A and port C configured as inputs, port B configured as outputs,  $V_{IL} = 0.2 \text{ V}$ ,  $V_{IH} = V_{DD} - 0.2 \text{ V}$
- Stop  $I_{DD}$  is measured with  $OSC1 = V_{SS}$ .
- Wait  $I_{DD}$  is affected linearly by the OSC2 capacitance.
- Pullups are designed to be capable of pulling to  $V_{IH}$  within 25  $\mu s$  for a 100 pF, 4-k $\Omega$  load.



**Figure 11-1. Maximum Supply Current versus Internal Clock Frequency**

## 11.8 Control Timing (5.0 Vdc and 2.2 Vdc)

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation Crystal External Clock	$f_{osc}$	— dc	4.2 4.2	MHz
Internal Operating Frequency Crystal ( $f_{osc} / 2$ ) External Clock ( $f_{osc} / 2$ )	$f_{op}$	— dc	2.1 2.1	MHz
Cycle Time	$t_{cyc}$	480	—	ns
Crystal Oscillator Startup Time	$t_{OXOV}$	—	100	ms
Stop Recovery Startup Time (Crystal Oscillator)	$t_{ILCH}$	—	100	ms
$\overline{RESET}$ Pulse Width	$t_{RL}$	1.5	—	$t_{cyc}$
Interrupt Pulse Width Low (Edge-Triggered)	$t_{ILIH}$	125	—	ns
Interrupt Pulse Period	$t_{ILIL}$	Note 2	—	$t_{cyc}$
OSC1 Pulse Width	$t_{OH}, t_{OL}$	90	—	ns

NOTES:

- $V_{DD} = 2.0$  to  $5.5$  Vdc,  $V_{SS} = 0$  Vdc,  $T_A = 0$  °C to  $+70$  °C, unless otherwise noted
- The minimum period,  $t_{ILP}$ , should not be less than the number of cycle times it takes to execute the interrupt service routine plus  $19 t_{CYC}$ .



## Section 12. Mechanical Specifications

### 12.1 Contents

12.2	Introduction . . . . .	111
12.3	28-Pin Plastic Dual-In-Line Package (Case 710-02) . . . . .	112
12.4	28-Pin Small Outline Integrated Circuit Package (Case 751F-04) . . . . .	112
12.5	44-Pin Plastic Leaded Chip Carrier Package (Case 777-02) . . . . .	113

### 12.2 Introduction

This section describes the dimensions of the dual-in-line package (DIP), small outline integrated circuit (SOIC), and plastic leaded chip carrier (PLCC) MCU packages.

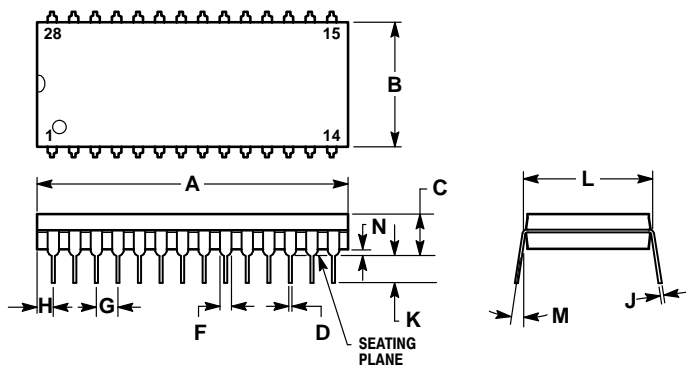
The following figures show the latest packages at the time of this publication. To make sure that you have the latest package specifications, contact one of the following:

- Local Motorola Sales Office
- Motorola Mfax
  - Phone 602-244-6609
  - EMAIL [rmfax0@email.sps.mot.com](mailto:rmfax0@email.sps.mot.com)
- Worldwide Web (wwweb) at <http://design-net.com>

Follow Mfax or wwweb on-line instructions to retrieve the current mechanical specifications.

# Mechanical Specifications

## 12.3 28-Pin Plastic Dual In-Line Package (Case 710-02)

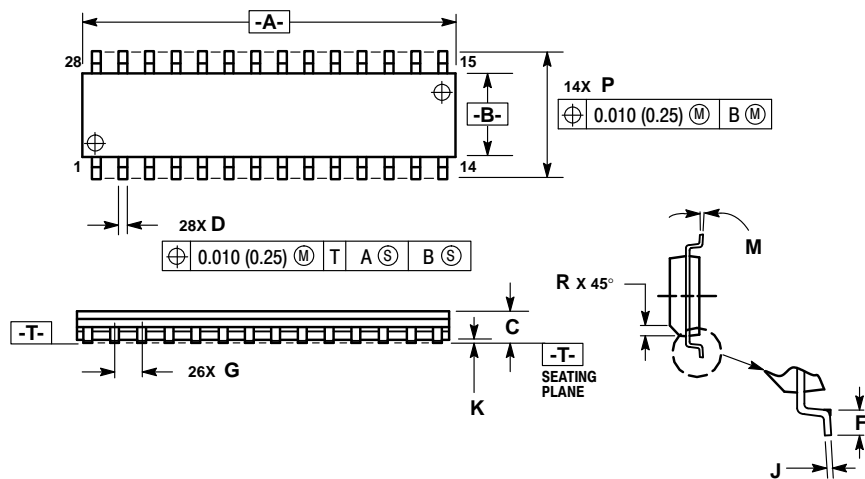


### NOTES:

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.

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

## 12.4 28-Pin Small Outline Integrated Circuit Package (Case 751F-04)



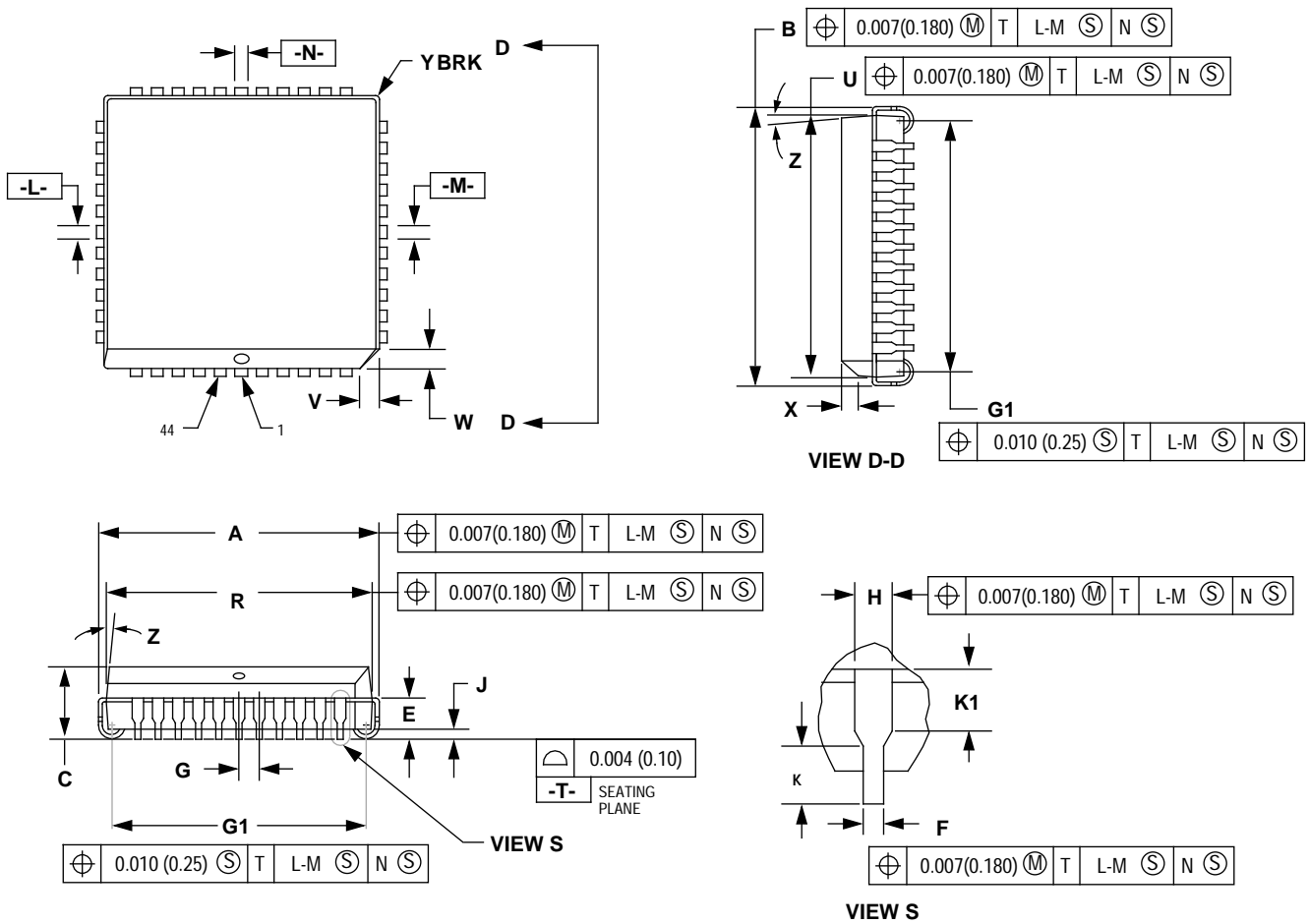
### NOTES:

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. DIMENSION D DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL BE 0.13 (0.005) TOTAL IN EXCESS OF D DIMENSION AT MAXIMUM MATERIAL CONDITION.

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	0.23	0.32	0.009	0.013
K	0.13	0.29	0.005	0.011
M	0°	8°	0°	8°
P	10.05	10.55	0.395	0.415
R	0.25	0.75	0.010	0.029



## 12.5 44-Pin Plastic Leaded Chip Carrier Package (Case 777-02)



NOTES:

- DATUMS -L-, -M-, AND -N- ARE DETERMINED WHERE TOP OF LEAD SHOULDERS EXITS PLASTIC BODY AT MOLD PARTING LINE.
- DIMENSION G1, TRUE POSITION TO BE MEASURED AT DATUM -T-, SEATING PLANE.
- DIMENSION R AND U DO NOT INCLUDE MOLD FLASH. ALLOWABLE MOLD FLASH IS 0.010 (0.25) PER SIDE.
- DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
- CONTROLLING DIMENSION: INCH.
- THE PACKAGE TOP MAY BE SMALLER THAN THE PACKAGE BOTTOM BY UP TO 0.012 (0.300). DIMENSIONS R AND U ARE DETERMINED AT THE OUTERMOST EXTREMES OF THE PLASTIC BODY EXCLUSIVE OF THE MOLD FLASH, THE BAR BURRS, GATE BURRS AND INTERLEAD FLASH, BUT INCLUDING ANY MISMATCH BETWEEN THE TOP AND BOTTOM OF THE PLASTIC BODY.
- DIMENSION H DOES NOT INCLUDE DAMBAR PROTRUSION OR INTRUSION. THE DAMBAR PROTRUSION(S) SHALL NOT CAUSE THE H DIMENSION TO BE GREATER THAN 0.037 (0.940114). THE DAMBAR INTRUSION(S) SHALL NOT CAUSE THE H DIMENSION TO SMALLER THAN 0.025 (0.635).

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



## Section 13. Ordering Information

### 13.1 Contents

13.2	Introduction . . . . .	115
13.3	MCU Ordering Forms . . . . .	115
13.4	Application Program Media. . . . .	116
13.5	ROM Program Verification . . . . .	117
13.6	ROM Verification Units (RVUs). . . . .	118
13.7	MC Order Numbers . . . . .	118

### 13.2 Introduction

This section contains ordering instructions for the MC68HC705RC16.

### 13.3 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 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 [13.4 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 lower-case letters. Then press the return key to start the BBS software.

### 13.4 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 1.4 M)
- MS-DOS<sup>®2</sup> or PC-DOS<sup>™3</sup> 3 1/2-inch diskette (double-sided 720 K or double-sided high-density 1.44 M)
- MS-DOS<sup>®</sup> or PC-DOS<sup>™</sup> 5 1/4-inch diskette (double-sided double-density 360 K or double-sided high-density 1.2 M)

Use positive logic for data and addresses.

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

- Customer name
- Customer part number
- Project or product name
- File name 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.

---

1. Macintosh is a registered trademark of Apple Computer, Inc.

2. MS-DOS is a registered trademark of Microsoft Corporation.

3. PC-DOS is a 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 nonuser ROM locations or leave all nonuser ROM locations blank. Refer to the current MCU ordering form for additional requirements. Motorola may request pattern re-submission if nonuser areas contain any nonzero code.*

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

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 file name of the source code.

## 13.5 ROM Program Verification

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

## 13.6 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 10 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 10 RVUs are free of charge with the minimum order quantity. These units are not to be used for qualification or production. RVUs are not guaranteed by Motorola Quality Assurance.

## 13.7 MC Order Numbers

**Table 13-1** provides information in determining order numbers.

**Table 13-1. MC Order Numbers**

Package Type	Operating Temperature Range	MC Order Number
28-Pin Plastic Dual In-Line Package (DIP)	0 to 70 °C	MC68HC05RC8P MC68HC05RC16P
28-Pin Small Outline Integrated Circuit Package (SOIC)	0 to 70 °C	MC68HC05RC8DW MC68HC05RC16DW
44-Pin Plastic Leaded Chip Carrier (PLCC)	0 to 70 °C	MC68HC05RC8FN MC68HC05RC16FN

## Appendix A. MC68HC05RC8

### A.1 Contents

A.2	Introduction .....	141
A.3	Memory Map .....	141

### A.2 Introduction

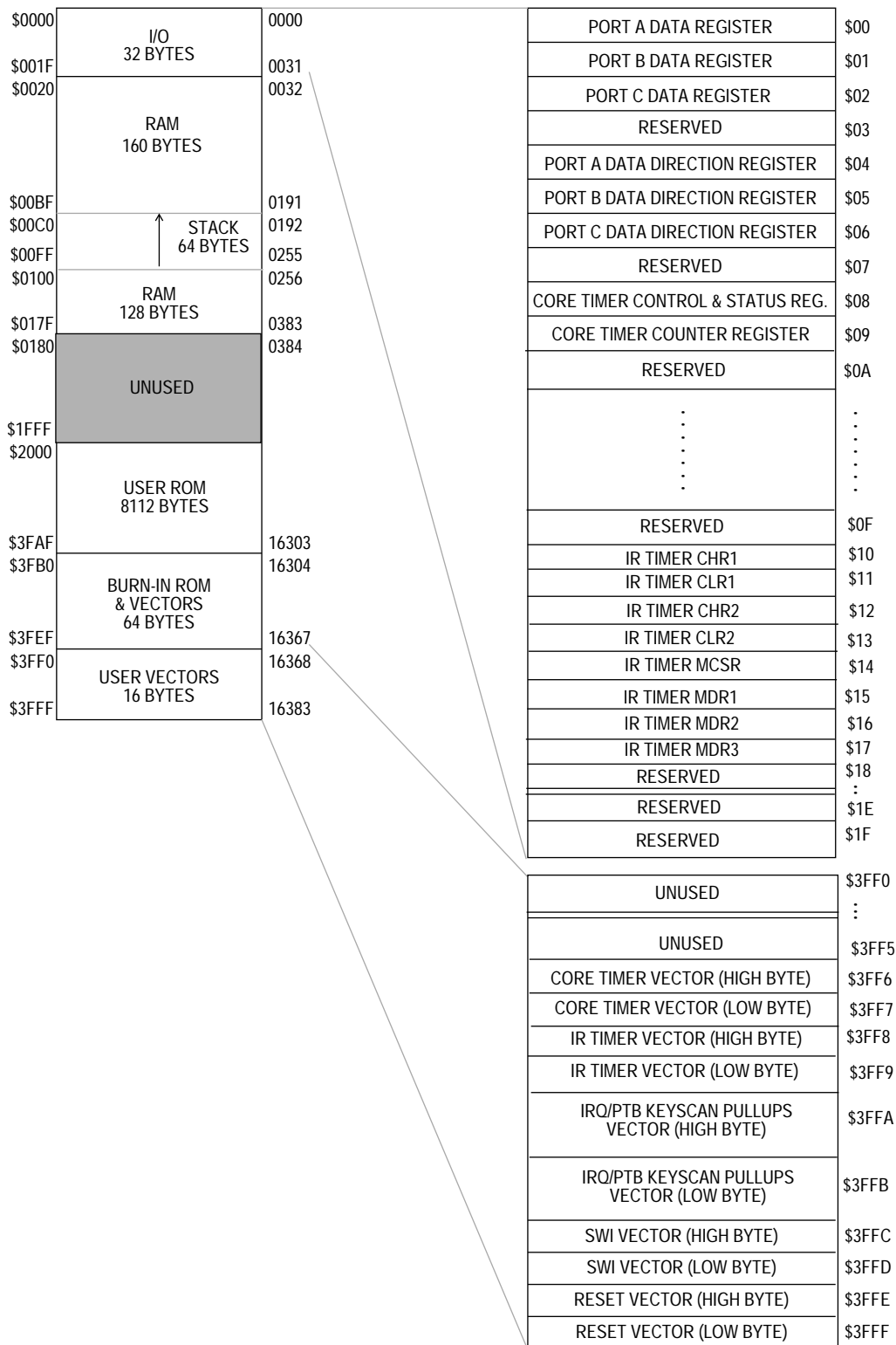
Appendix A introduces the MC68HC05RC8. The technical data applying to the MC68HC05RC16 applies to the MC68HC05RC8 with the exceptions given in this appendix.

### A.3 Memory Map

Both the MC68HC05RC8 and the MC68HC05RC16 have 16-Kbyte memory maps consisting of user ROM, RAM, burn-in ROM, and input/output (I/O). However, the user ROM for the MC68HC05RC8 consists of only 8112 bytes of ROM.

**Figure A-1** shows the MC68HC05RC8 memory map in user mode.


# MC68HC05RC8



**Figure A-1. MC68HC05RC8 Memory Map**





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 which may be provided in Motorola data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. 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.

**How to reach us:**

**USA/EUROPE/Locations Not Listed:** Motorola Literature Distribution; P.O. Box 20912; Phoenix, Arizona 85036. 1-800-441-2447 or 602-303-5454

**MFAX:** RMFAX0@email.sps.mot.com – TOUCHTONE 602-244-6609

**INTERNET:** <http://Design-NET.com>

**JAPAN:** Nippon Motorola Ltd.; Tatsumi-SPD-JLDC, 6F Seibu-Butsuryu-Center, 3-14-2 Tatsumi Koto-Ku, Tokyo 135, Japan.  
03-81-3521-8315

**ASIA/PACIFIC:** Motorola Semiconductors H.K. Ltd.; 8B Tai Ping Industrial Park, 51 Ting Kok Road, Tai Po, N.T., Hong Kong. 852-26629298



**MOTOROLA**

**HC05RC16GRS/D**