68HC05CJ4
68HC705CJ4

SPECIFICATION
(General Release)

©January 3, 1996

CSIC MCU Section
Sendai Design Operations
Nippon Motorola Ltd.
Sendai 981-32 Japan

TABLE OF CONTENTS

Paragraph   **Title**   **Page**

SECTION 1
GENERAL DESCRIPTION

SECTION 2
MEMORY

SECTION 3
CPU CORE

TABLE OF CONTENTS

TABLE OF CONTENTS

Paragraph	**Title**	**Page**

TABLE OF CONTENTS

SECTION 11
TIMER 1

SECTION 12
TIMER 2

SECTION 13
INSTRUCTION SET

TABLE OF CONTENTS

Paragraph	Title	Page

SECTION 14
ELECTRICAL SPECIFICATIONS

SECTION 15
MECHANICAL SPECIFICATIONS

SECTION 16
ORDERING INFORMATION

LIST OF FIGURES

LIST OF FIGURES

LIST OF TABLES

SECTION 1
GENERAL DESCRIPTION

1.1 Introduction

This MC68HC(7)05CJ4 is a 44-pin microcontroller unit (MCU) with highly sophisticated on-chip peripheral functions. The memory map includes nearly 4 Kbytes of user ROM or EPROM and 224 bytes of RAM. The MCU has four ports: A, B, C, and D. Ports A, B, and C are bidirectional, and port D is output only, supporting several multifunction pins. The MC68HC05CJ4 has two timers, a COP watchdog, a serial peripheral interface (SPI), a serial communications interface (SCI), and a slave-only I^{2}C Bus.

1.2 Features

- Low Cost

- M68HC05 Core

- 44-Lead Quad Flat Pack (QFP) Package

- 3856 Bytes of User Mask ROM or EPROM

- 224 Bytes of On-Chip RAM

- 24 Bidirectional I/O Lines, 7 Input-Only Lines

- 15-Stage Multifunctional Timer

- 16-Bit Timer with Input Capture and Output Compare

- COP Watchdog Timer

- Power Saving Stop Mode/Wait Mode

- Asynchronous Serial Communications Interface (SCI)

- Synchronous Serial Peripheral Interface (SPI)

- Slave Only I^{2}C Bus (M-Bus)

- Port C has 10 mA-Per-Pin Drive Capability

Figure 1-1. Block Diagram

$0000	I/O 32 BYTES	0000
$0020	RAM 224 BYTES	0032
$00C0	STACK 64 BYTES	0192
$0100	UNUSED 3840 BYTES	0256
$1000	USER ROM 3840 BYTES	4096
$1F00	SELF-CHECK ROM 240 BYTES	7936
$1FE0	SELF-CHECK VECTORS	8160
$1FF0	USER VECTORS 16 BYTES	8176
$1FFF		8191
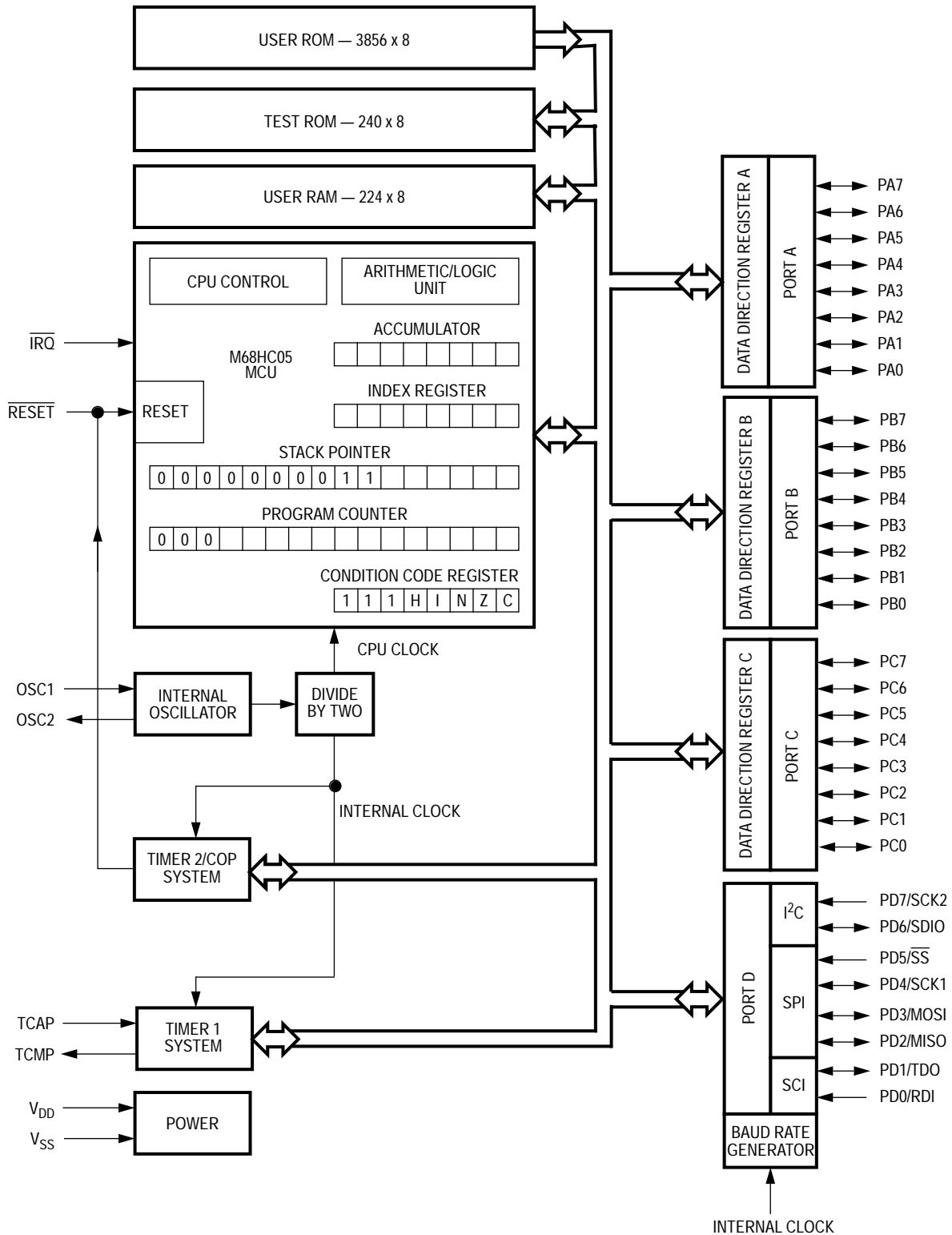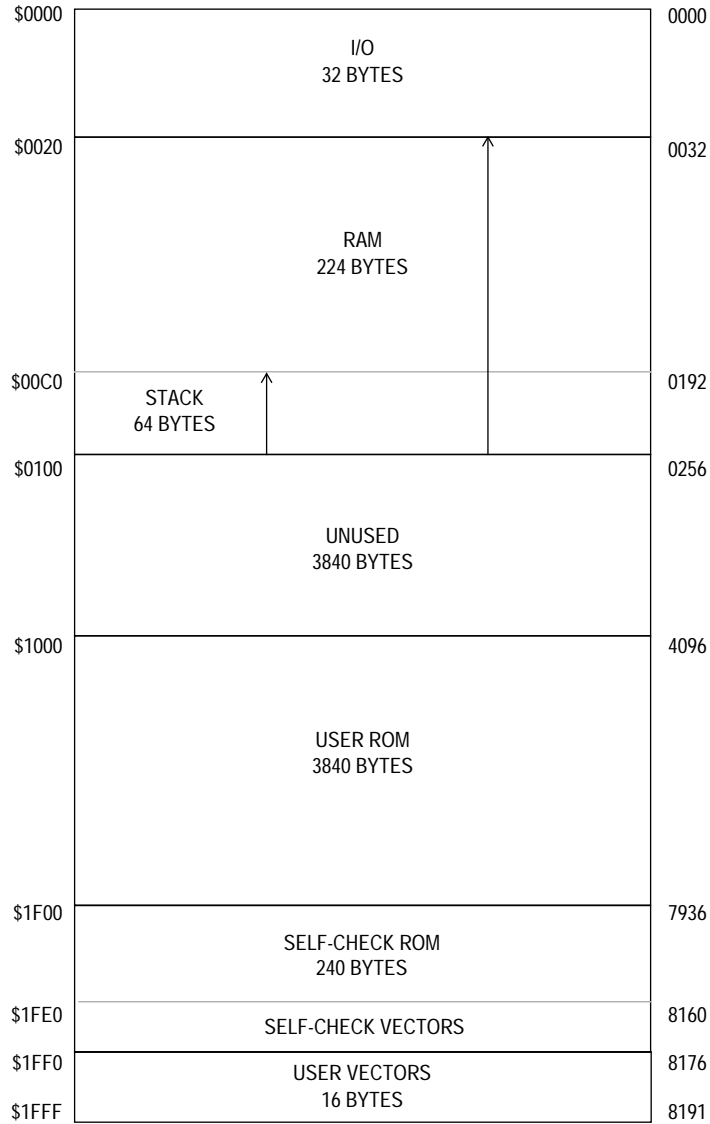
Figure 1-2. **Memory Map**

NOTE

A line over a signal name indicates an active low signal. For example, RESET is active high and $\overline{\text{RESET}}$ is active low. Any reference to voltage, current, or frequency specified in the following sections will refer to the nominal values. The exact values and their tolerance or limits are specified in **SECTION 14 ELECTRICAL SPECIFICATIONS.**
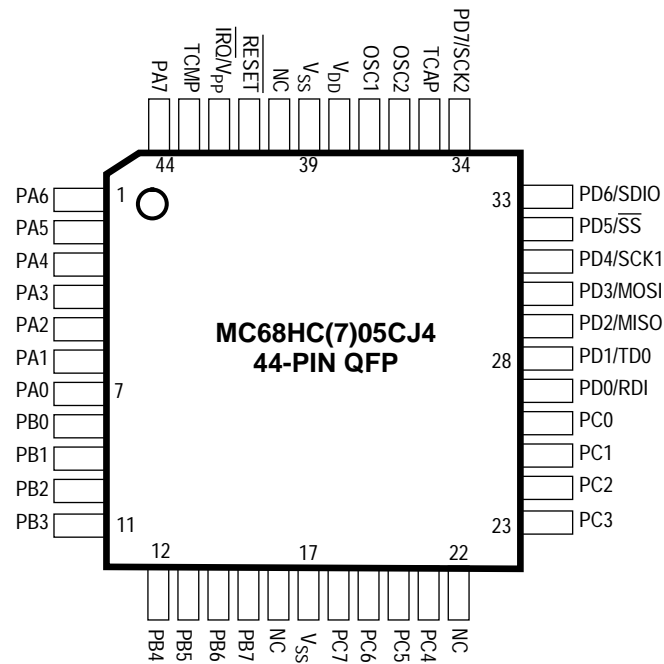
**Figure 1-3. Pin Assignment for Single-Chip Mode
(44-Lead QFP Package)**

1.3 Mask Options

Stop disable is a mask programmable option on MC68HC05CJ4.

There are no mask programmable options on MC68HC705CJ4.

1.4 Signal Description

This section mainly describes signals in the single-chip mode or the user mode. See Figure 1-3.

1.4.1 V_{DD} and V_{SS}

The power is supplied to the microcontroller using V_{DD} and V_{SS}. V_{DD} is the positive supply, and V_{SS} is ground. For more detailed information refer to **14.6 DC Operating Characteristics**.

1.4.2 $\overline{RESET}$

This pin is used to reset the MCU and provide an orderly start-up procedure by pulling $\overline{RESET}$ low. The $\overline{RESET}$ and power-on reset (POR) functions do not

depend on the MCU operation modes and all pins are configured as single-chip mode pins while the RESET pin is low.

1.4.3 Maskable Interrupt Request (IRQ)

This pin has a programmable option that provides two different choices of interrupt triggering sensitivity. The options are:

1. negative edge-sensitive triggering only, or

2. both negative edge-sensitive and level-sensitive triggering.

The MCU completes the current instruction before it responds to the interrupt request. When IRQ goes low for at least one t_{ILIH}, a logic one is latched internally to signify an interrupt has been requested. When the MCU completes its current instruction, the interrupt latch is tested. If the interrupt latch contains a logic one, and the interrupt mask bit (I bit) in the condition code register is clear, the MCU then begins the interrupt sequence. See **SECTION 4 INTERRUPTS** for more information on IRQ programming.

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

The IRQ pin contains an internal Schmitt trigger as part of its input to improve noise immunity.

On the MC68HC705CJ4, this pin becomes IRQ/V_{PP} and has the added function of providing the programming voltage for the on-board EPROM.

1.4.4 OSC1 and OSC2

These pins provide control input for an on-chip clock oscillator circuit. A crystal or an external clock signal connects to these pins providing a system clock.

1.4.5 TCAP and TCMP

These two pins are associated with the 16-bit timer (timer 1). TCAP is an input only to the input capture system, and TCMP is an output only from the output compare system. See **SECTION 11 TIMER 1**.

1.4.6 PA0 through PA7

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 (DDRA) is at $0004. Reset does not affect the data register, but clears the data direction register, thereby returning the port to inputs. Writing a one to a DDRA bit sets the corresponding port bit to output mode.

1.4.7 PB0 through PB7

Port B is an 8-bit bidirectional port which does not share any of its pins with other subsystems. The port B data register is at $0001 and the data direction register (DDRB) is at $0005. Reset does not affect the data register, but clears the data direction register, thereby returning the port to inputs. Writing a one to a DDRB bit sets the corresponding port bit to output mode.

1.4.8 PC0 through PC7

Port C is an 8-bit bidirectional port which does not share any of its pins with other subsystems. The port C data register is at $0002 and the data direction register (DDRC) is at $0006. Reset does not affect the data register, but clears the data direction register, thereby returning the port to inputs. Writing a one to a DDRC bit sets the corresponding port bit to output mode.

1.4.9 PD0 through PD7

Port D is an 8-bit input-only port. PD7:PD6 pins are shared with the I^2C subsystem, PD5:PD2 are shared with the SPI subsystem and PD1:PD0 are shared with the SCI subsystem. The port D data register is at $0003 and there is no data direction register. Reset does not affect the data register.

SECTION 2
MEMORY

2.1 Introduction

The MC68HC05CJ4 contains 3840 mask ROM, 240 bytes of self-check ROM, and 224 bytes of RAM. An additional 16 bytes of mask ROM is provided for user vectors at $1FF0 through $1FFF.

In the MC68HC705CJ4 (EPROM device), ROM and user vector areas are replaced by the EPROM cell, and self-check ROM becomes bootstrap ROM.

Address	Memory		Register	Address
$0000	I/O 32 BYTES		PORT A DATA REGISTER	$00
			PORT B DATA REGISTER	$01
		224 BYTES	PORT C DATA REGISTER	$02
$001F			PORT D DATA REGISTER	$03
$0020	RAM 224 BYTES		PORT A DATA DIRECTION REG	$04
			PORT B DATA DIRECTION REG	$05
			PORT C DATA DIRECTION REG	$06
$00C0	STACK 64 BYTES	FOR EPROM PARTS ONLY	PROGRAM CONTROL REGISTER	$07
$00FF			TIMER STATUS AND CONTROL	$08
$0100			TIMER COUNTER REGISTER	$09
			SPI CONTROL REGISTER	$0A
	UNUSED		SPI STATUS REGISTER	$0B
			SPI DATA REGISTER	$0C
			SCI BAUD RATE REGISTER	$0D
			SCI CONTROL REGISTER 1	$0E
			SCI CONTROL REGISTER 2	$0F
$1000			SCI STATUS REGISTER	$10
			SCI DATA REGISTER	$11
			TIMER CONTROL REGISTER	$12
			TIMER STATUS REGISTER	$13
	MASK ROM 3840 BYTES (EPROM 3840 BYTES)		INPUT CAPTURE REGISTER (H)	$14
			INPUT CAPTURE REGISTER (L)	$15
			OUTPUT COMPARE REGISTER (H)	$16
			OUTPUT COMPARE REGISTER (L)	$17
			TIMER COUNTER (H)	$18
			TIMER COUNTER (L)	$19
			ALTERNATE COUNTER (H)	$1A
$1F00			ALTERNATE COUNTER (L)	$1B
	SELF-TEST ROM (BOOTSTRAP ROM)		MBUS ADD AND DATA REGISTER	$1C
			MBUS CONTROL REGISTER	$1D
$1FDF			MBUS STATUS REGISTER	$1E
$1FE0	RESERVED		RESERVED	$1F
$1FEF				
$1FF0	USER VECTORS			
$1FFF				

Figure 2-1. MC68HC(7)05CJ4 Memory Map

2.2 Summary of Internal Registers and I/O Map

Figure 2-2 explains how to interpret the register figures used in this document.

	B7	B6	B5	B4	B3	B2	B1	B0	
$0012	ICIE	OCIE	TOIE	0	0	0	IEDG	OLVL	XXXX
RESET:	0	0	0	0	0	0	U	0	

REGISTER ADDRESS

BIT 7 IDENTIFIER

BIT NAME (MNEMONIC)

REGISTER NAME

RESET state or condition:
(U = Unaffected by RESET)
(* = Reflects input pin state)
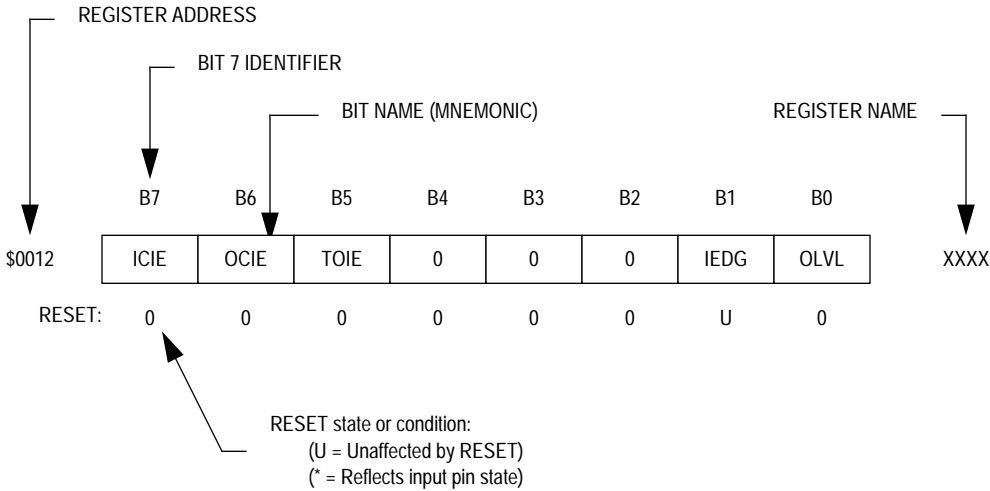
Figure 2-2. Register Description Key

INTERNAL REGISTERS

	Bit 7	6	5	4	3	2	1	Bit 0	
$0000	PA7	PA6	PA5	PA4	PA3	PA2	PA2	PA0	PORTA
$0001	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0	PORTB
$0002	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0	PORTC
$0003	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0	PORTD
$0004	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0	DDRA
$0005	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0	DDRB
$0006	DDRC7	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0	DDRC
$0007									UNUSED
$0008	TOF	RTIF	TOFE	RTIE	IRQS	COPE	RT1	RT0	T2CSR
$0009	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0	T2CR
$000A	SPIE	SPE	DOD	MSTR	CPOL	CPHA	SPR1	SPR0	SPCR
$000B	SPIF	WCOL	0	MODF	0	0	0	0	SPSR
$000C	SPD7	SPD6	SPD5	SPD4	SPD3	SPD2	SPD1	SPD0	SPDR
$000D	0	0	SCP1	SCP0	0	SCR2	SCR1	SCR0	BAUD
$000E	R8	T8	0	M	WAKE	0	0	0	SCCR1
$000F	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK	SCCR2
	Bit 7	6	5	4	3	2	1	Bit 0	

Figure 2-3. I/O Map ($0000:$000F)

INTERNAL REGISTERS

	Bit 7	6	5	4	3	2	1	Bit 0	
$0010	TDRE	TC	RDRF	IDLE	OR	NF	FE	0	SCSR
$0011	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0	SCDR
$0012	ICIE	OCIE	TOIE	0	0	0	IEDG	OLVL	T1CR
$0013	ICF	OCF	TOF	0	0	0	0	0	T1SR
$0014	BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8	ICRH
$0015	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0	ICRL
$0016	BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8	OCRH
$0017	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0	OCRL
$0018	BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8	TRH
$0019	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0	TRL
$001A	BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8	ATRH
$001B	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0	ATRL
$001C	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0	MBADR
$001D	SMIE	SME	T/R	NOACK	0	0	0	CLKR	MBCR
$001E	SMF	SRDF	MACK	0	0	0	0	0	MBSR
$001F	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0	TEST
	Bit 7	6	5	4	3	2	1	Bit 0	

Figure 2-4. I/O Map ($0010:$001F)

2.3 RAM

The 224-byte internal RAM is positioned at $0020 through $00FF in the memory map. All of the memory is positioned in page zero and is accessible by direct addressing mode, but the upper 64 bytes of page zero are used for the CPU stack area. Extreme caution should be taken if the stack area is used for data storage.

The RAM is implemented with static cells and retains its contents during the stop and wait modes.

2.4 Self-Check ROM (MC68HC05CJ4)

Self-check ROM is the 240 bytes of mask ROM positioned at $1F00 through $1FEF. This ROM contains self-check programs and reset/interrupt vectors in the self-check mode.

2.5 Boot ROM (MC68HC705CJ4)

Boot ROM is the 240 bytes of mask ROM positioned at $1F00 through $1FEF. This ROM contains bootstrap programs and reset/interrupt vectors in the bootstrap mode. The programs include:

- EPROM Programming and Verify
- Dumping EPROM Contents
- Reading Program into the Internal RAM
- Executing Program in the Internal RAM

2.6 Mask ROM (MC68HC05CJ4)

The 3840-byte user ROM is positioned at $1000 through $1EFF, and additional 16-byte ROM is located at $1FF0 through $1FFF for user vectors. In this mask ROM device, V_{PP} pin is not used and program control register (PCR) is not implemented.

2.7 EPROM/OTP (MC68HC705CJ4)

The 3840-byte EPROM is positioned at $1000 through $1EFF, and additional 16-byte EPROM is located at $1FF0 through $1FFF for user vectors. The erased state of EPROM is read as $FF and EPROM power is supplied from V_{PP} pin and V_{DD} pin.

The program control register (PCR) is provided for the EPROM programming and testing. This register is only available in special test modes.

SECTION 3
CPU CORE

3.1 Introduction

The MCU contains five registers as shown in the programming model of Figure 3-1. This section describes these registers.
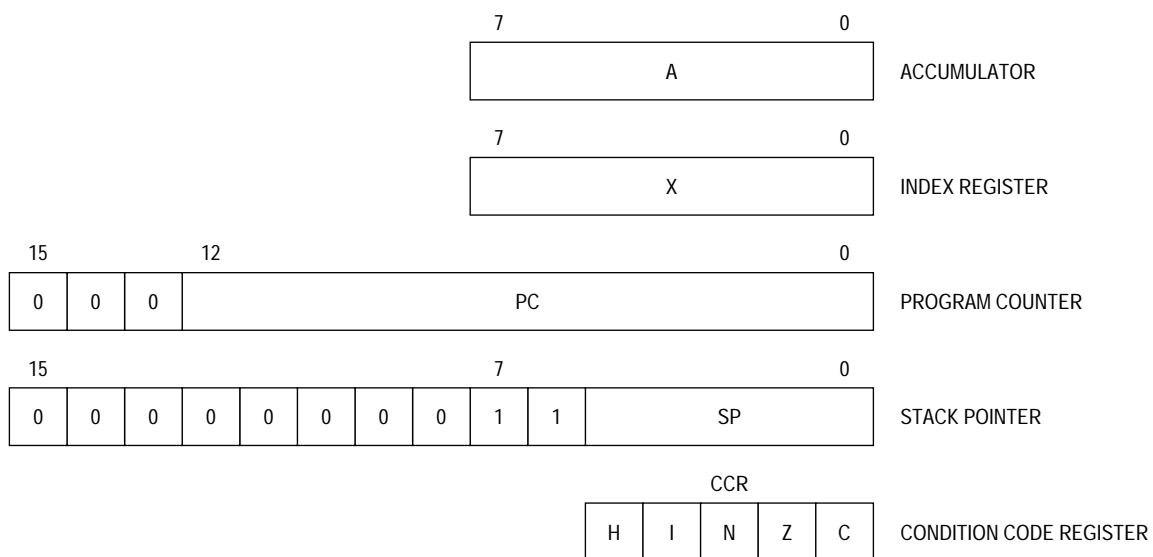
Figure 3-1. Programming Model

3.2 Accumulator (A)

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

3.3 Index Register (X)

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

3.4 Program Counter (PC)

The program counter is a 16-bit register that contains the address of the next byte to be fetched.

3.4.1 Stack Pointer (SP)

The stack pointer 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 10 most significant bits are permanently set to 0000000011. These 10 bits are appended to the six least significant 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. See Figure 3-2 for more information.
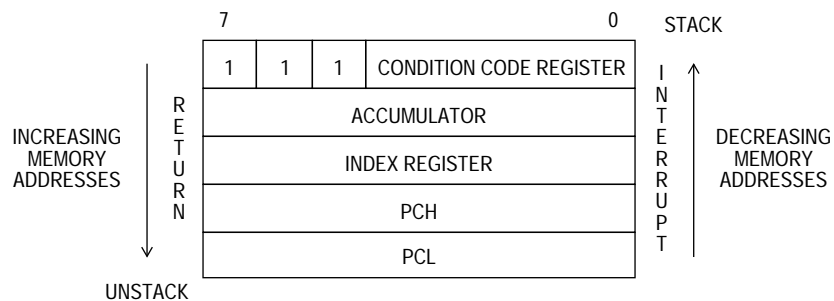
Figure 3-2. Stacking Order

NOTE

Since the stack pointer decrements during pushes, the program counter low (PCL) is stacked first, followed by program counter high (PCH), etc. Pulling from the stack is in the reverse order.

3.4.2 Condition Code Register (CCR)

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

Half Carry (H)

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

Interrupt (I)

When this bit is set, the hardware 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.

Negative (N)

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

Zero (Z)

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

Carry/Borrow (C)

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

SECTION 4
INTERRUPTS

4.1 Introduction

The MCU can be interrupted seven different ways: the six maskable hardware interrupts ($\overline{\text{IRQ}}$, SCI, SPI, I^{2}C, timer 1, and timer 2) and the non-maskable software interrupt instruction (SWI).

Interrupts cause the processor to save register contents on the stack and to set the interrupt mask (I bit) to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack and normal processing to resume.

Unlike RESET, hardware interrupts do not cause the current instruction execution to be halted, but are considered pending until the current instruction is complete.

NOTE

The current instruction is the one already fetched and being operated on.

When the current instruction is complete, the processor checks all pending hardware interrupts. If interrupts are not masked (CCR I bit clear) and if the corresponding interrupt enable bit is set, the processor proceeds with interrupt processing; otherwise, the next instruction is fetched and executed.

The SWI is executed the same as any other instruction, regardless of the I-bit state.

See Table 4-1 for more information.

Table 4-1. Vector Address for Interrupts and Reset

Register	Flag Name	Interrupts	CPU Interrupt	Vector Address
N/A	N/A	Reset	RESET	$1FFE–$1FFF
N/A	N/A	Software	SWI	$1FFC–$1FFD
N/A	N/A	External Interrupt	IRQ	$1FFA–$1FFB
TSR	ICF	Timer Input Capture	TIMER 1	$1FF8–$1FF9
TSR	OCF	Timer Output Compare	TIMER 1	$1FF8–$1FF9
TSR	TOF1	Timer 1 Overflow	TIMER 1	$1FF8–$1FF9
SCSR	TDRE	Transmit Buffer Empty	SCI	$1FF6–$1FF7
SCSR	TC	Transmit Complete	SCI	$1FF6–$1FF7
SCSR	RDRF	Receiver Buffer Full	SCI	$1FF6–$1FF7
SCSR	IDLE	Idle Line Detect	SCI	$1FF6–$1FF7
SCSR	OR	Overrun	SCI	$1FF6–$1FF7
SPSR	SPIF	Transfer Complete	SPI	$1FF4–$1FF5
SPSR	MODF	Mode Fault	SPI	$1FF4–$1FF5
MBSR	SMF	Slave M-Bus Flag	IIC	$1FF2–$1FF3
TCSR	TOF2	Timer 2 Overflow	TIMER 2	$1FF0–$1FF1
TCSR	RTI	Real Time Interrupt	TIMER 2	$1FF0–$1FF1

4.2 Hardware Controlled Interrupt Sequence

The following three functions ($\overline{\text{RESET}}$, STOP, and WAIT) are not in the strictest sense an interrupt; however, they are acted upon in a similar manner. Flowcharts for hardware interrupts are shown in Figure 4-1**.** A discussion is provided below.

1. $\overline{\text{RESET}}$ — A low input on the $\overline{\text{RESET}}$ input pin causes the program to vector to its starting address which is specified by the contents of memory locations $1FFE and $1FFF. The I bit in the condition code register is also set. Much of the MCU is configured to a known state during this type of reset.

2. STOP — The STOP instruction causes the oscillator to be turned off and the processor to "sleep" until an external interrupt ($\overline{\text{IRQ}}$) or reset occurs.

3. WAIT — The WAIT instruction causes all processor clocks to stop, but leaves the timer clocks running. This "rest" state of the processor can be cleared by reset, or any hardware interrupt. There are no special wait vectors for these individual interrupts.

4.3 Software Interrupt (SWI)

The SWI is an executable instruction and a non-maskable interrupt: It is executed regardless of the state of the I bit in the CCR. If the I bit is zero (interrupts enabled), SWI executes after interrupts which were pending when the SWI was fetched, but before interrupts generated after the SWI was fetched. The interrupt service routine address is specified by the contents of memory locations $1FFC and $1FFD.

4.4 External Interrupt

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 IRQ. It is then synchronized internally and serviced as specified by the contents of $1FFA and $1FFB.

Either a level-sensitive and edge-sensitive trigger, or an edge-sensitive-only trigger operation is selectable by writing to bit 3 of the T2CSR register.

NOTE

The internal interrupt latch is cleared in the first part of the interrupt service routine; therefore, one external interrupt pulse could be latched and serviced as soon as the I bit is cleared.

4.5 Timer 1 Interrupt

There are three different timer interrupt flags that cause a timer interrupt whenever they are set and enabled. The interrupt flags are in the timer status register (TSR), and the enable bits are in the timer control register (TCR). Any of these interrupts will vector to the same interrupt service routine, located at the address specified by the contents of memory location $1FF8 and $1FF9.

4.6 SCI Interrupt

There are five different SCI interrupt flags that cause an SCI interrupt whenever they are set and enabled. The interrupt flags are in the SCI status register (SCSR), and the enable bits are in the SCI control register 2 (SCCR2). Any of these interrupts will vector to the same interrupt service routine, located at the address specified by the contents of memory location $1FF6 and $1FF7.

4.7 SPI Interrupt

There are two different SPI interrupt flags that cause an SPI interrupt whenever they are set and enabled. The interrupt flags are in the SPI status register (SPSR), and the enable bits are in the SPI control register (SPCR). Either of these interrupts will vector to the same interrupt service routine, located at the address specified by the contents of memory location $1FF4 and $1FF5.

4.8 I^{2}C Interrupt

There is one I^{2}C interrupt flag that causes an I^{2}C interrupt whenever it is set and enabled. The interrupt flag is in the I^{2}C status register (MBSR), and the enable bits are in the I^{2}C control register (MBCR). This interrupt will vector to the same interrupt service routine, located at the address specified by the contents of memory location $1FF2 and $1FF3.

4.9 Timer 2 Interrupt

There are two different timer 2 interrupt flags that cause a timer interrupt whenever they are set and enabled. The interrupt flags and enables are in the timer status and control register (TCSR). Either of these interrupts will vector to the same interrupt service routine, located at the address specified by the contents of memory location $1FF0 and $1FF1.
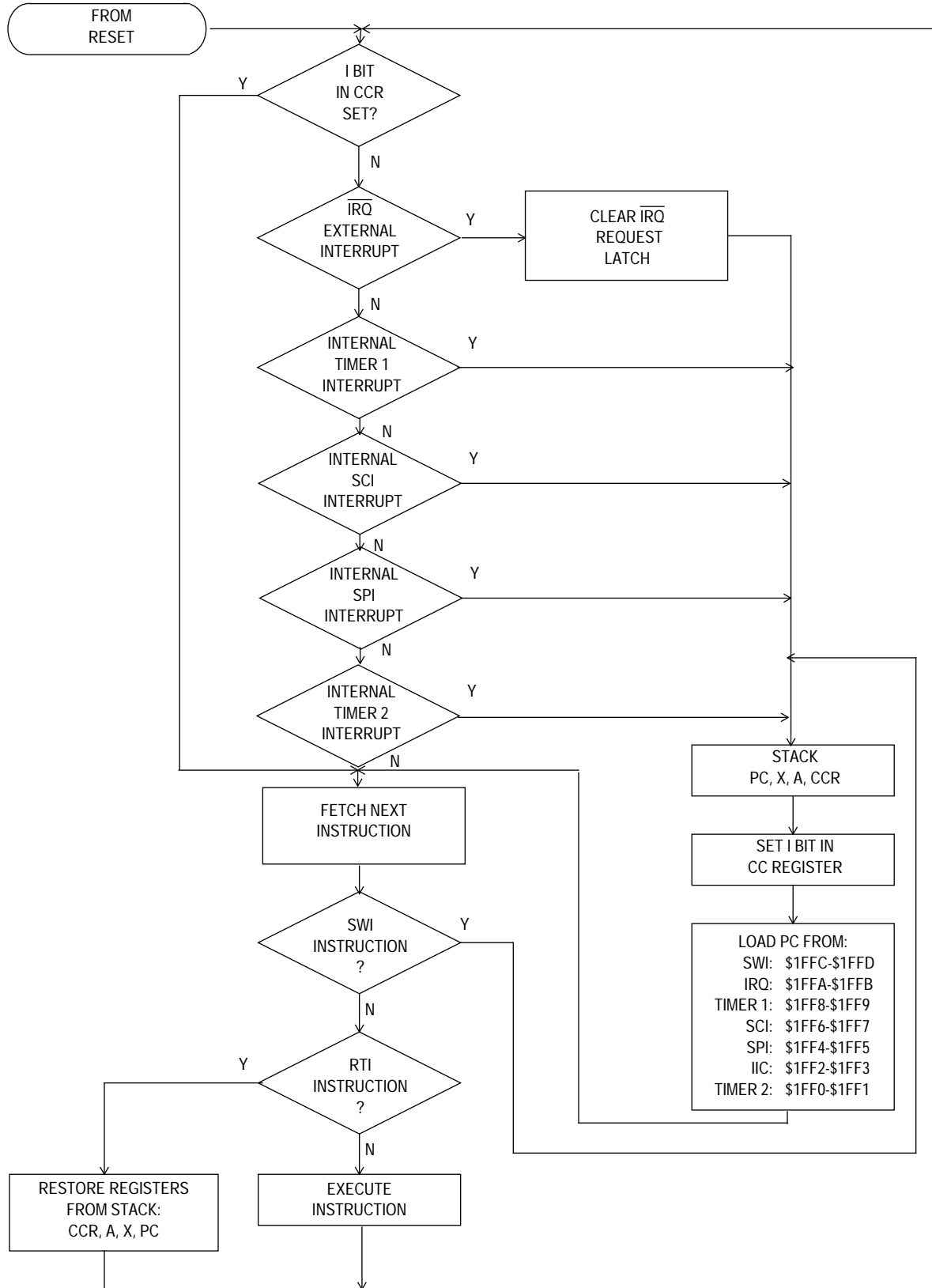
Figure 4-1. Interrupt Flowchart

SECTION 5
RESETS

5.1 Introduction

The MCU can be reset two ways: by the initial power-on reset function or by an active low input to the $\overline{\text{RESET}}$ pin. See Figure 5-1.

5.2 Power-On Reset (POR)

An internal reset is generated on power-up to allow the internal clock generator to stabilize. The power-on reset is strictly for power turn-on conditions and should not be used to detect a drop in the power supply voltage. There is a 4064 internal processor clock cycle (t_{cyc}) oscillator stabilization delay after the oscillator becomes active.

If the $\overline{\text{RESET}}$ pin is low at the end of this 4064 cycle delay, the MCU will remain in the reset condition until $\overline{\text{RESET}}$ goes high.

5.3 $\overline{\text{RESET}}$ Pin

The MCU is reset when a logic zero is applied to the $\overline{\text{RESET}}$ input for a period of one and one-half machine cycles (t_{cyc}).

5.4 Computer Operating Properly (COP) Reset

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. Because the internal $\overline{\text{RESET}}$ signal is used, the MCU comes out of a COP reset in the same operating mode it was in when the COP time-out was generated.

The COP reset function is a software programmable option.

Refer to **12.4 COP Watchdog Reset** for more information on the COP.
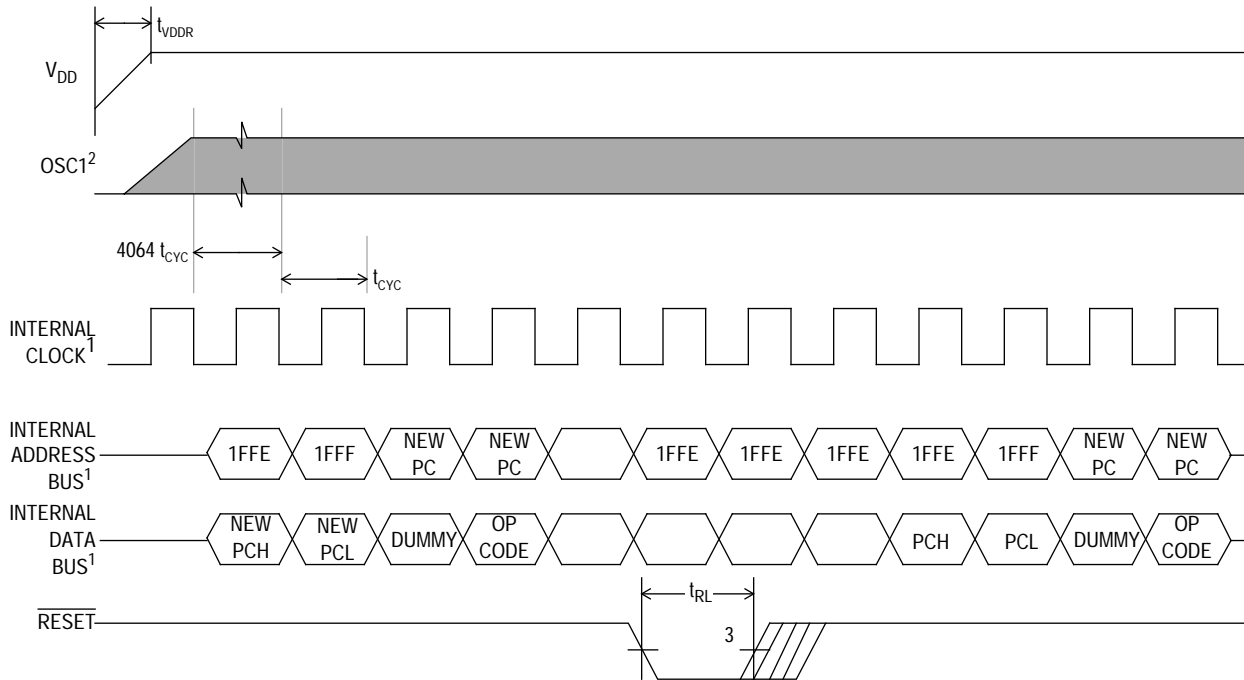
NOTES:
  1. Internal timing signal and bus information not available externally.
  2. OSC1 line is not meant to represent frequency. It is only used to represent time.
  3. The next rising edge of the internal processor clock following the rising edge of $\overline{\text{RESET}}$ initiates the reset sequence.

Figure 5-1. Power-On Reset and $\overline{\text{RESET}}$

SECTION 6
MODES OF OPERATION

6.1 Introduction

The MC68HC05CJ4 has the following operating modes: single-chip mode (SCM) and self-check mode. In the MC68HC705CJ4, the self-check mode becomes bootstrap mode.

The single-chip mode allows maximum use of pins for on-chip peripheral functions.

The self-check capability of MC68HC05CJ4 provides an internal check to determine if the device is functional.

The bootstrap loader mode is provided for EPROM programming, dumping EPROM contents, and reading programs into the internal RAM and executing it. This is a very versatile mode because essentially on the special purpose program that is bootloaded into the internal RAM has no limitations.

This section also provides a description of the low-power modes.

6.2 Mode Entry

The mode entry is done at the rising edge of the $\overline{\text{RESET}}$ pin. Once the device enters one of the four modes, the mode can be changed only by external reset not software.

At the rising edge of the $\overline{\text{RESET}}$ pin, the device latches the states of the $\overline{\text{IRQ}}$, PB6 and PB7 pins and places itself in the specified mode. While the $\overline{\text{RESET}}$ pin is low, all pins are configured as single-chip mode. Table 6-1 shows the states of $\overline{\text{IRQ}}$ and PB6:7 pins for each mode. See Figure 6-1.

Table 6-1. Mode Select Summary

Mode	$\overline{\text{RESET}}$	$\overline{\text{IRQ}}$	PB6	PB7
Single-Chip Mode	⌐	L or H	X	X
Self-Check/Bootstrap Mode	⌐	V_{TST}	H	X

$V_{TST} = 1.8 \times V_{DD}$
$H = V_{DD}$
$L = GND$
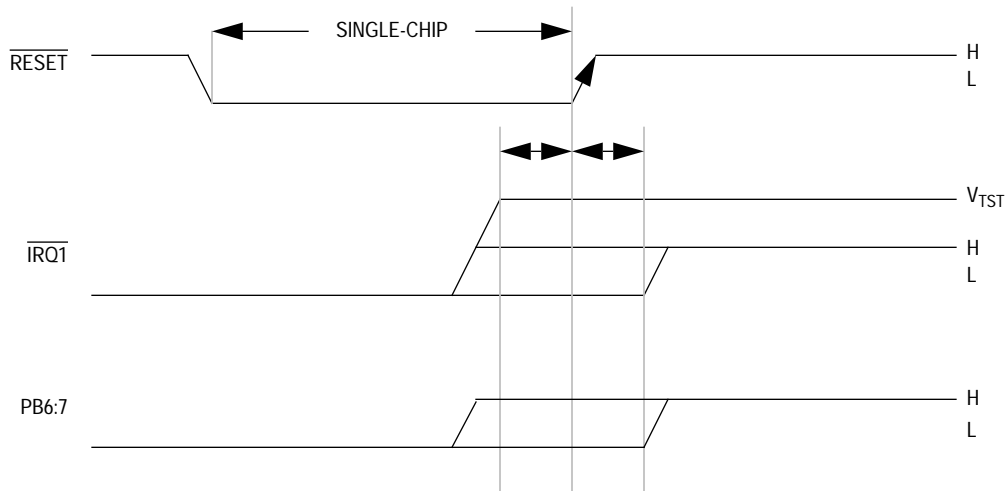
* V_{TST} = 1.8 x V_{DD}

Figure 6-1. MC68HC(7)05CJ4 Mode Entry Diagram

6.3 Single-Chip Mode (SCM)

In this mode, all address and data bus activity occurs within the MCU so no external pins are required for these functions. The single-chip mode allows the maximum number of I/O pins for on-chip peripheral functions, port A through port D.

6.4 Self-Check/Bootstrap Mode

In this mode, the reset vector is fetched from a 240-byte internal self-check (bootstrap for MC68HC705CJ4) ROM at $1F00:$1FEF. The self-check ROM contains a self-check program to test the functions of internal modules. The bootstrap ROM contains a small program which reads a program into the internal RAM and then passes control to that program at location $0020, or executes EPROM programming sequence and dumps EPROM contents.

Since these modes are not normal user modes, all of the privileged control bits are accessible. This allows the self-check/bootstrap mode to be used for self testing the device.

6.5 Low-Power Modes

The following sub-sections describe the low-power modes. Figure 6-2 provides a timing diagram of the stop recovery mode and Figure 6-3 a flowchart for the stop and wait modes.

6.5.1 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 the stop mode, the TCSR (timer 2) bits are altered to remove any pending timer interrupt request and to disable any further timer interrupts. The I bit in the CCR is cleared to enable external interrupts. All other registers and memory remain unaltered. All input/output lines remain unchanged. The processor can be brought out of the stop mode only by an external interrupt or reset.

6.5.2 Stop Recovery

The processor can be brought out of the stop mode only by an external interrupt or RESET.

NOTES:
1. Represents the internal gating of the OSC1 pin.
2. $\overline{IRQ}$ pin edge-sensitive software option.
3. $\overline{IRQ}$ pin level and edge-sensitive software option.
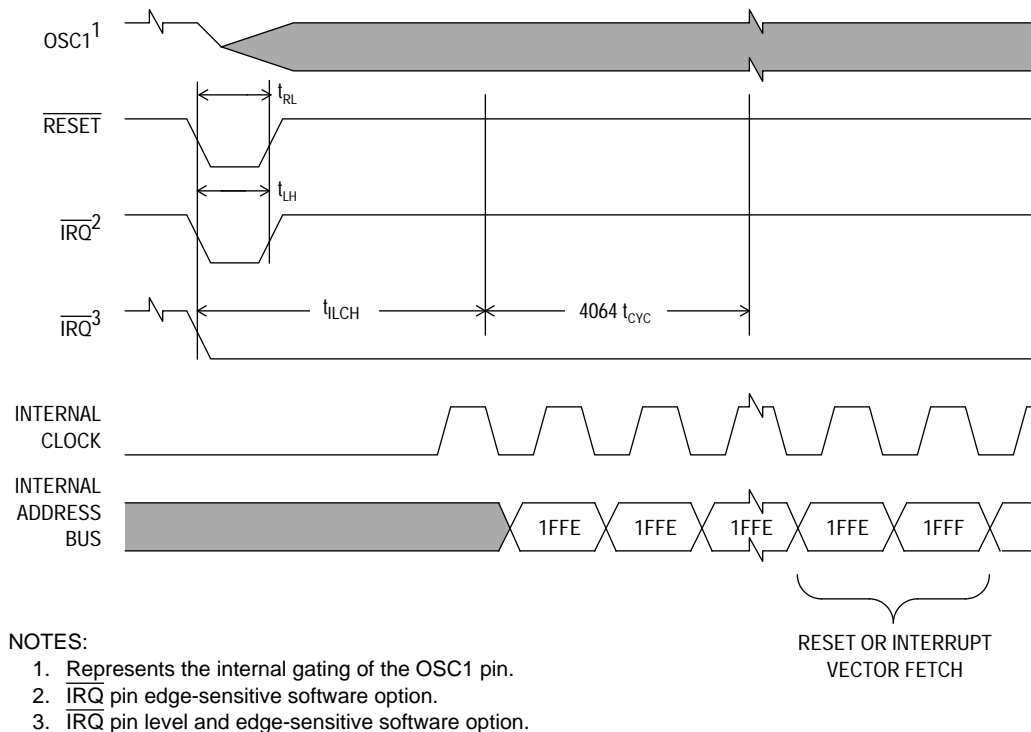
Figure 6-2. Stop Recovery Timing Diagram

6.5.3 Wait Mode

The WAIT instruction places the MCU in a low-power consumption mode, but the wait mode consumes more power than the stop mode. All CPU action is suspended, but the timers and the oscillator remain active. Any interrupt or reset will cause the MCU to exit the wait mode.

During the wait mode, the I bit in the CCR is cleared to enable interrupts. All other registers, memory, and input/output lines remain in their previous state. The timers may be enabled to allow a periodic exit from the wait mode.
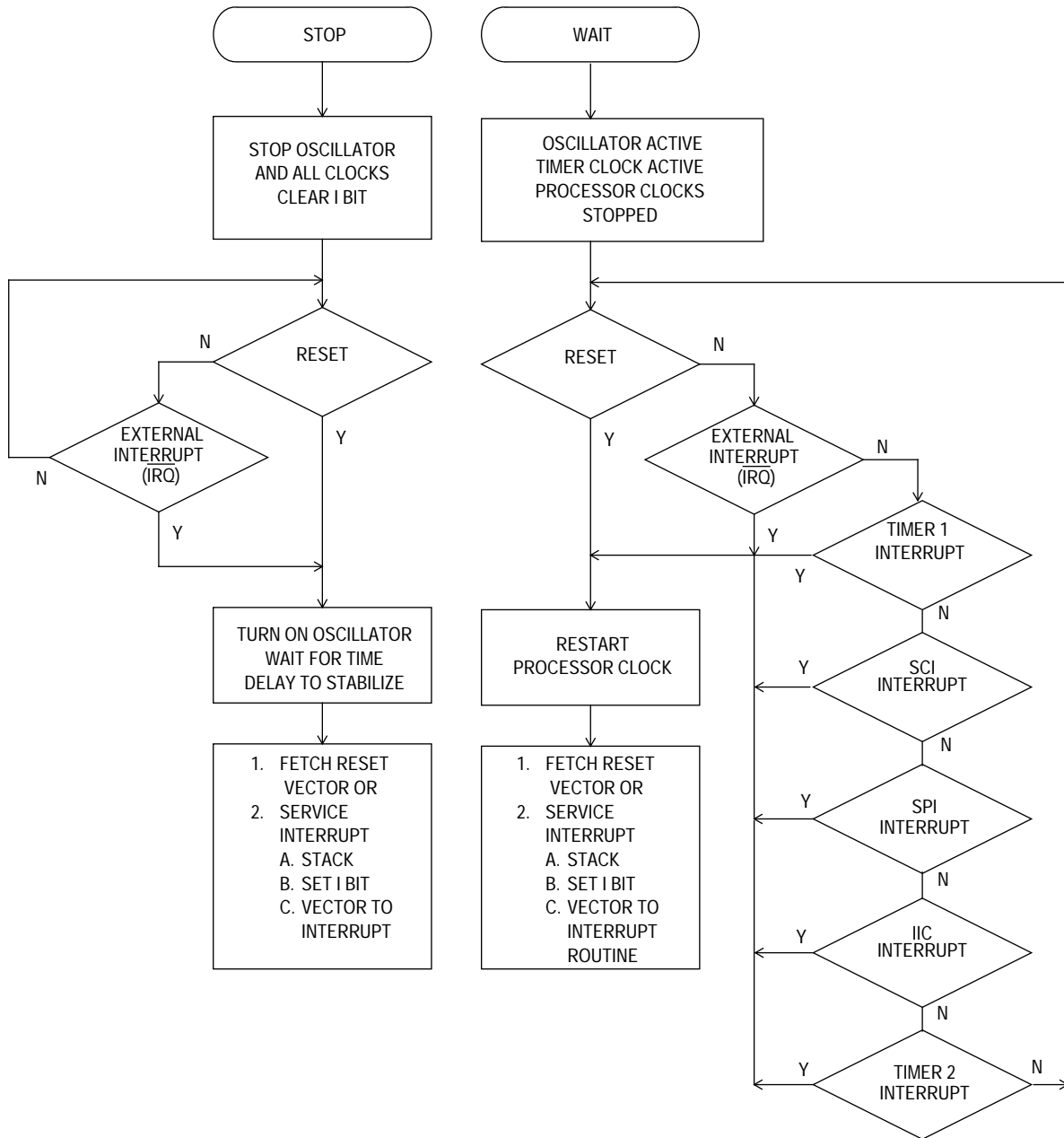
Figure 6-3. Stop/Wait Mode Flowcharts

SECTION 7
INPUT/OUTPUT PORTS

7.1 Introduction

In user mode there are 32 lines arranged as four 8-bit ports. Most of these port pins are programmable as either inputs or outputs under software control of the data direction registers, though some are input only.

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.

7.2 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 registers, but clears the data direction registers, thereby returning the ports to inputs. Writing a one to a DDR bit sets the corresponding port bit to output mode.

7.3 Port B

Port B is an 8-bit bidirectional port. The port B data register is at $0001 and the data direction register (DDR) is at $0005. Reset does not affect the data registers, but clears the data direction registers, thereby returning the ports to inputs. Writing a one to a DDR bit sets the corresponding port pin to output mode.

7.4 Port C

Port C is an 8-bit bidirectional port. The port C data register is at $0002 and the data direction register (DDR) is at $0006. Reset does not affect the data registers, but clears the data direction registers, thereby returning the ports to inputs. Writing a one to a DDR bit sets the corresponding port bit to output mode. PC7 has a high current sink and source capability.

GENERAL RELEASE SPECIFICATION

7.5 Port D

Port D is an 8-bit fixed input port. Four of its pins are shared with the SPI subsystem, two are shared with the I^{2}C subsystem and two more are shared with the SCI subsystem. During reset, all eight bits become valid input ports because all special function output drivers associated with the SCI, I^{2}C and SPI subsystems are disabled.

7.6 Input/Output Programming

Port pins for ports A, B, and C 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). Ports A, B, and C have an associated DDR. Any I/O port pin is configured as an output if its corresponding DDR bit is set to a logic one. A pin is configured as an input if its corresponding DDR bit is cleared to a logic zero.

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. For further information see Table 7-1 and Figure 7-1.

Table 7-1. I/O Pin Functions

R/$\overline{\text{W}}$	DDR	I/O Pin Function
0	0	The I/O pin is in input mode. Data is written into the output data latch.
0	1	Data is written into the output data latch and output to the I/O pin.
1	0	The state of the I/O pin is read.
1	1	The I/O pin is in an output mode. The output data latch is read.
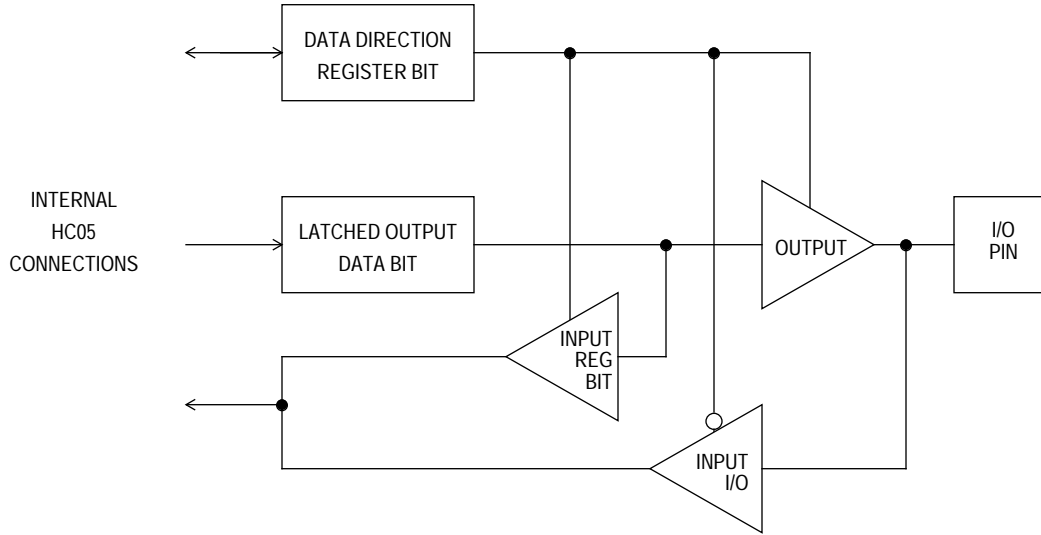
Figure 7-1. I/O Circuitry

SECTION 8
SERIAL COMMUNICATIONS INTERFACE

8.1 Introduction

This is an NRZ asynchronous communication system with internal baud rate generation circuitry. It can be configured for eight or nine data bits. The baud rate generator is programmable to allow flexibility in choosing baud rates. There is a receiver wake-up feature, an idle line detect feature and various error detection features. Two port D pins provide the external interface for the transmitted data (TXD) and the received data (RXD).

8.2 Transmit Operation

This transmitter includes a transmit serial shift register and a parallel transmit data register (forming a double buffered system). The transmit serial shift register is internal to the transmit logic and may not be read or written directly by the CPU. The output of this serial shifter is connected to the TXD pin (port D bit 1) as long as the transmitter is enabled (TE control bit set or TE clear but a transmit operation being completed). If the transmitter is enabled (TE = 1), TC will be set at the end of a frame provided the transmit data register is empty (TDRE = 1), there is no pending preamble, and no pending break.

The transmitter can operate in either of two formats as specified by the M control bit in the SCCR1 register. When M is zero, one start bit, eight data bits, and one stop bit is the selected mode. When M is one, one start bit, nine data bits, and one stop bit is the selected mode. The most common standard word format for NRZ serial communication is one start bit (logic zero or space) followed by eight data bits (LSB first) and one stop bit (logic one or mark). If the nine data bit mode is selected, software control (and overhead) of the ninth bit may be used to support a number of special formats. The ninth data bit is positioned as two bits in the SCCR1 register (R8 and T8). It remains unchanged after transmission and may be used again without having to re-write it. Some examples of its use include:

- start, eight data, two stop bits

- start, eight data, parity, one stop
  w/ odd, even, mark, or space parity

- start, seven data, parity, two stop bits
  w/ odd, even, mark, or space parity

- start, eight data, address/control, one stop bit
  where address/control bit identifies special command words

When the transmit logic is enabled by writing a one to the TE control bit in the SCCR2 register, a check is made to determine whether or not the transmit serial shift register is empty. If it was empty (indicated by TC = 1) a preamble word of all ones and no start bit is transmitted and normal data transmission begins. If the shifter was not empty (TC = 0) then normal shifting continues until the current word is shifted out including the stop bit and then a preamble is transmitted and normal data transmission continues.

Data to be transmitted originates from the CPU and enters the serial transmit logic when it is written to the data register (SCDR). If 9-bit data is to be transmitted, the T8 bit should be initialized before writing to the SCDR. Writes to the SCDR register access the write-only TDR and reads access the read-only RDR. Before writing to the TDR, the program should read the SCSR status register. If the TDRE bit in SCSR is clear, useful data is in the TDR and writes to TDR would erroneously overwrite this information. If the TDRE bit is set it indicates that the TDR is empty and a subsequent write to TDR will fill the TDR and automatically clear TDRE. As soon as the data in the serial shifter has finished shifting out a check is made to see if there is a new byte of data in the TDR. If TDRE = 0, then data is transferred from the TDR to the transmit shifter and TDRE becomes set automatically (optionally causing an interrupt). This transfer from TDR to the shifter is synchronized with the baud rate clock. If TDRE = 1 when the shifter becomes available and there is no preamble pending, then an idle condition will be entered in which the TXD pin will remain high.

Messages may be separated by a serial preamble of 10 (11 if nine data bit format is specified) bit times of marks (ones). To force this separation preamble with minimum idle line time the following sequence is used:

1. Write last byte of first message to TDR.

2. Wait for TDRE to go high, indicating the last byte has been transferred to the shifter.

3. Clear Transmit Enable (TE) and then set TE back to one. This queues a preamble to immediately follow the transmission of the last character of the first message (including stop bit).

4. Write first byte of second message to TDR.

In this sequence, if the first byte of the second message is not transferred to the TDR prior to the finish of the preamble transmission, then the transmit data line will simply mark idle until the TDR is finally written. Also, if the last byte of the first message finishes shifting out (including stop bit) and TE is clear then the TC bit will

go high and transmission will be considered to be complete. The TXD pin also will revert to being a general purpose input line.

When the TE control bit is cleared (and left clear), the transmit logic gives up control of the port D pin in the following controlled manner. If no data is being shifted out and the transmitter is in an idle state (TC = 1), then when TE is cleared the port D pin immediately reverts to being a general purpose input line. If a transmission was still in progress (TC = 0), the character in the transmit shift register will continue to be shifted out normally including any stop bit. When this character is finished, the TXD pin reverts to being a general purpose input line regardless of whether any data is pending in the TDR (TDRE is a don't care). In order to avoid accidentally cutting off the transmitter before the last character in a message, the software should always wait for TDRE to go high following the last write to TDR before clearing TE.

Another transmit-related SCI function is the send break function. This function is used to abort transmissions by sending a minimum of 10 (11 if nine data bit format is specified) bit times of space (logic zeros). The break function is invoked by writing a one to the SBK bit in the SCCR2 register. If SBK is set while a transmission is in progress, then the character in the transmit shift register will be finished normally (including stop bit) before the break function begins. The logic zero state on the TXD line will be a minimum of 10 (11) bit times but will continue indefinitely as long as the SBK bit remains set. To guarantee the minimum break time, SBK should just be toggled quickly to a one and back to zero. After a break period, at least one bit time of mark (logic one) will be transmitted to guarantee recognition of a subsequent start bit.

8.3 Receive Operation

The receive logic in the SCI is divided into two major sections. The first section is a front end that synchronizes to the asynchronous receive data and evaluates the logic sense of each bit in the serial stream. The second section controls the functional operation and the interface to the CPU.

8.3.1 Receiver Front End

Due to the asynchronous nature of this serial communication system, the receiver has a significant section of logic dedicated to gaining bit time synchronization with the incoming data and evaluating the logic sense of each bit time in the serial data stream. An understanding of this receiver front end logic is required before the functional operation of the remaining receive logic can be explained. This receiver front end logic uses a clock that is 16 times the baud rate frequency as a sampling clock (The sampling clock is called the RT clock in the following discussions.). In the remainder of this discussion, one RT is understood to be 1/16 of a bit time.

A few definitions will help in understanding the following discussions.

Bit Time

A bit time is the period of time required to serially transmit or receive one bit of data and is equal to one cycle of the baud rate frequency.

Mark/Space

Standard NRZ format is sometimes called mark/space format where a mark is a bit time of logic one and a space is a bit time of logic zero.

Start Bit

A start bit is a bit time of logic zero which indicates the beginning of a data frame. A start bit must begin with a one to zero transition and is preceded by at least one bit time of logic one.

Stop Bit

A stop bit is one bit time of logic one which indicates the end of a data frame.

Frame

A frame consists of a start bit followed by a specified number of data or information bits terminated by a stop bit. The number of data or information bits depends on the format specified and must agree between the transmitting device and the receiving device. The most common frame format is one start bit followed by eight data bits (LSB first) terminated by one stop bit for a total of ten bit times in the frame. This device is also capable of operating in a nine data bit format as described elsewhere in this specification.

When the receiver is first enabled, by writing a one to the RE bit in the SCCR2 register, the receiver front end logic begins an asynchronous search for a start bit. The goal of this search is to gain synchronization with a frame. This bit time synchronization is done at the beginning of each frame so that small differences in the baud rate of the receiver and transmitter are not cumulative. The circuit also resynchronizes on all one-to-zero transitions in the serial data stream.

The sequence of events in searching for a start bit is as follows:

1. Sample RXD input during each RT period and maintain these samples in a serial pipeline.
    * if RXD is already low, do not begin the start bit search - go to step 1.
    * if RXD is high, look for two more high RT samples before beginning the start bit search. Stay in step 1 until a total of three consecutive high RT samples have been received.

2. RXD was low during this RT period and high during the previous three periods. Consider this sample to be time "RT1 and the official start of the start bit search.

3. Ignore the sample at RT2 and sample RXD at RT3. Call this the first test sample.

4. Ignore the sample at RT4 and sample RXD at RT5. Call this the second test sample. If test samples at RT3 and RT5 were both high, assume that the low detected at RT1 was noise and go back to step 1 else continue.

5. Ignore the sample at RT6 and sample RXD at RT7. Call this the third test sample. If any two of the test samples at RT3, RT5, or RT7 were high assume that the low detected at RT1 was noise and go back to step 1 else continue.

6. A valid start bit has been found and synchronization has been achieved. From this point on until the end of the frame the RT clock will increment to RT16 and start over at RT1 where RT1 is considered to be the beginning of a bit time and RT16 is considered to be the end of a bit time.

NOTE

Any one-to-zero transition will re-synchronize the RT clock and may shift the assumed location of RT1.

Upon detection of a valid start bit, synchronization has been established and locked to the internal 16 times baud rate clock. The incoming data is no longer considered asynchronous until after reception of the stop bit, at which time a new asynchronous search for a start bit is initiated.

During each bit time of the frame, including the start and stop bit times, three logic sense samples are taken at RT8, RT9, and RT10. The logic sense of the bit time is taken to be the sense of the majority of these three samples (except the start bit, which is assumed to be logic sense zero regardless of the RT8 – RT10 samples). Note that during the start bit time, samples were also taken at RT1, RT3, RT5, and RT7 and the samples taken at RT8, RT9, and RT10 during the start bit time are only used for possible setting of a working "noise flag". There is a working "noise flag" associated with each received frame which implies whether or not the information in the frame is likely to be in error. If the noise flag is clear, it implies good information, and, if this flag is set, it implies that the data may (but probably does not) contain errors. At the beginning of a frame, the working noise flag starts out at zero. If any of the samples taken at RT3, RT5, or RT7 during the start bit search was a high, the working noise flag will be set. Also, if the samples taken at RT8, RT9, and RT10 during any one bit time (including the start bit, all data bits, optional ninth bit, and stop bit) do not all three agree, the working noise flag will be set.

The following figures demonstrate the operation of the start bit search and synchronization procedure. The logic sense sampling at RT8, RT9, and RT10 assumes that the probability of noise affecting three consecutive RT samples is acceptably low, so the examples will not consider any such cases.

Example 1 shows the ideal case with no noise present.

* = RESTART RT CLOCK
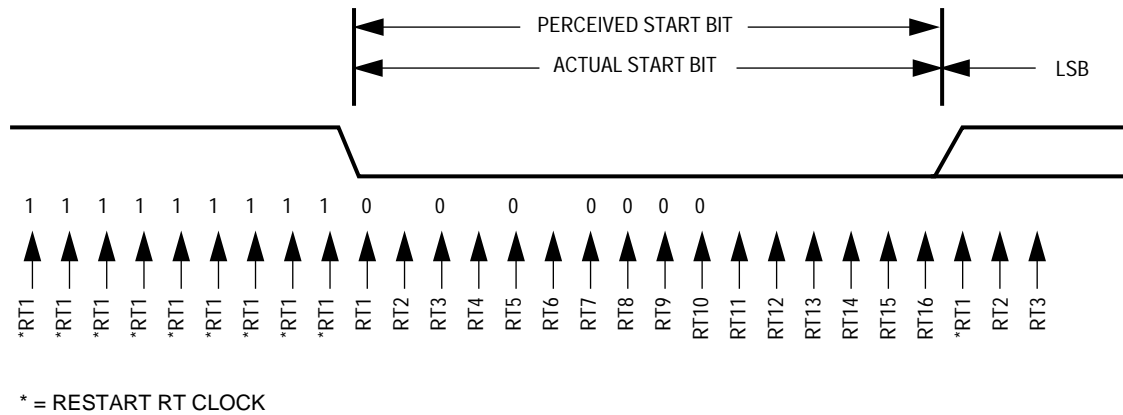
Figure 8-1. Start Search Example 1

Example 2 shows the start bit search and resynchronization process being restarted because the first low detected was determined to e noise rather than being the beginning of a start bit time. Since the noise was before the start bit was found, it would not cause the working noise flag to be set.

* = RESTART RT CLOCK
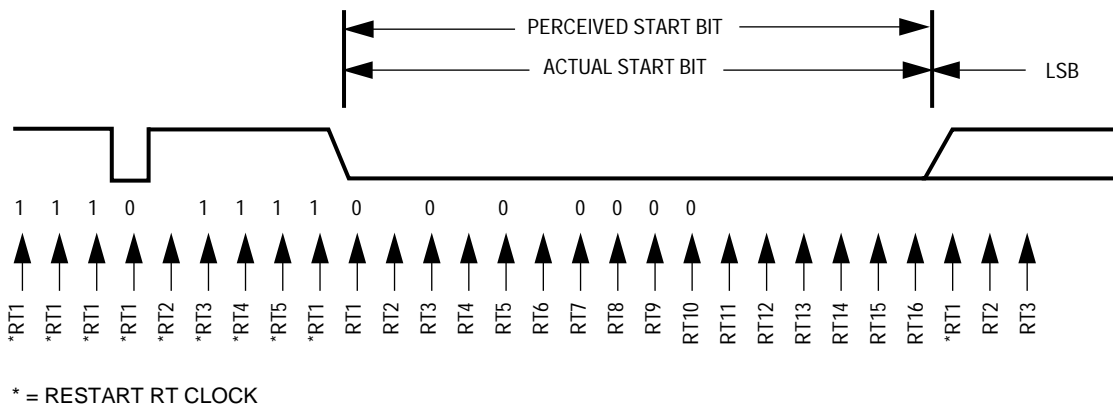
Figure 8-2. Start Search Example 2

In example 3, noise is perceived as the beginning of a start bit although the test sample at RT3 was high setting the working noise flag. Even though this shows improper alignment of perceived bit time boundaries to actual bit time boundaries the logic sense samples taken at RT8, RT9, and RT10 will fall well within the correct actual bit time and data recovery should still be good.

PERCEIVED START BIT

ACTUAL START BIT   LSB

1 1 1 1 1 0   1   0   0   0   0   0   :

*RT1 *RT1 *RT1 *RT1 *RT1 *RT1 RT2 RT3 RT4 RT5 RT6 RT7 RT8 RT9 RT10 RT11 RT12 RT13 RT14 RT15 RT16 *RT1 RT2 RT3 RT4 RT5 RT6 RT7

* = RESTART RT CLOCK
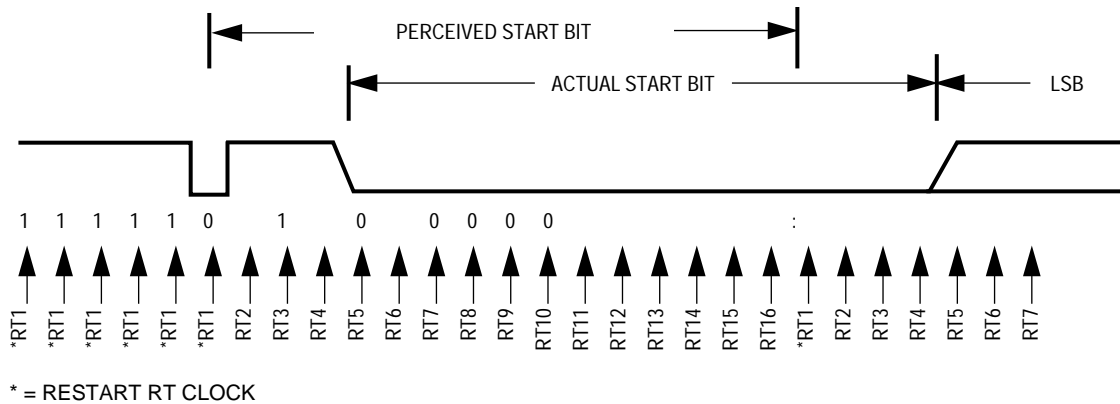
Figure 8-3. Start Search Example 3

In example 4, a large burst of noise is perceived as the beginning of a start bit, although the test sample at RT5 was high, setting the working noise flag. Even though this shows a worst case alignment of perceived bit time boundaries to actual bit time boundaries, the logic sense samples taken at RT8, RT9, and RT10 will fall within the correct actual bit time, and data recovery should still be successful.

PERCEIVED START BIT

ACTUAL START BIT   LSB

1 1 1 0   0   1   0   0   0   0   :

*RT1 *RT1 *RT1 *RT1 RT2 RT3 RT4 RT5 RT6 RT7 RT8 RT9 RT10 RT11 RT12 RT13 RT14 RT15 RT16 *RT1 RT2 RT3 RT4 RT5 RT6 RT7 RT8 RT9

* = RESTART RT CLOCK
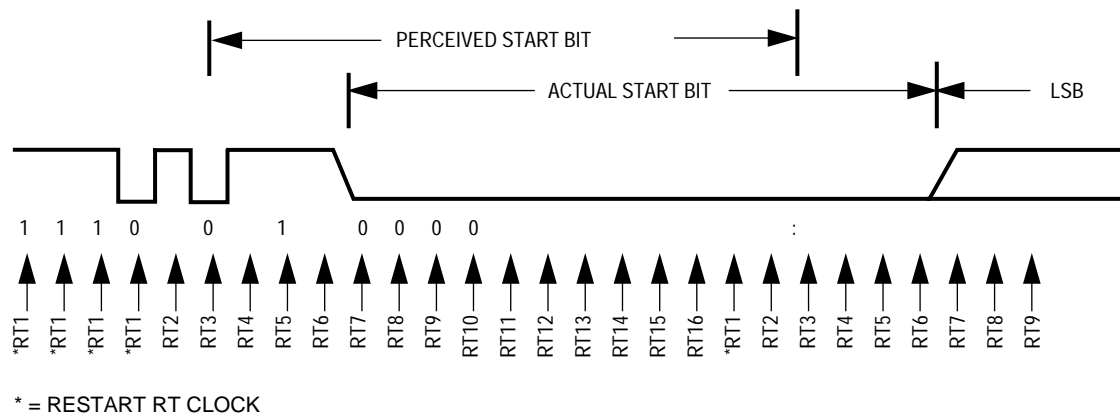
Figure 8-4. Start Search Example 4

Example 5 shows the effect of noise early within the start bit time. Although this noise does not affect proper synchronization with the start bit time, it does set the working noise flag.

Figure 8-5. Start Search Example 5

In example 6, a large burst of noise near the beginning of the start bit caused the start bit search to be restarted. During the RT period after the restart at RT7, a low was sensed but the preceding three RT samples were not all high so no falling edge could be recognized. Since the circuit cannot locate the start bit, the frame will be received as a Framing Error, improperly received, or missed entirely depending on the data in the frame and when the start bit search logic synchronized on what it thought was a start bit. In the narrative prior to the examples, it was stated that any burst of noise that could cause three consecutive RT samples to sense the incorrect data level was so improbable that it would not be considered. Example 6 shows two separate short duration noise incidents which are so specifically positioned within the frame that this case could be ignored on similar grounds of very low probability.

Figure 8-6. Start Search Example 6

In example 7, the majority of the logic sense samples taken at RT8, RT9, and RT10 during the start bit time were high, suggesting that the logic sense should be one but recall that, for the start bit time only, the logic sense indicated by the RT8, RT9, and RT10 samples is overridden to zero.
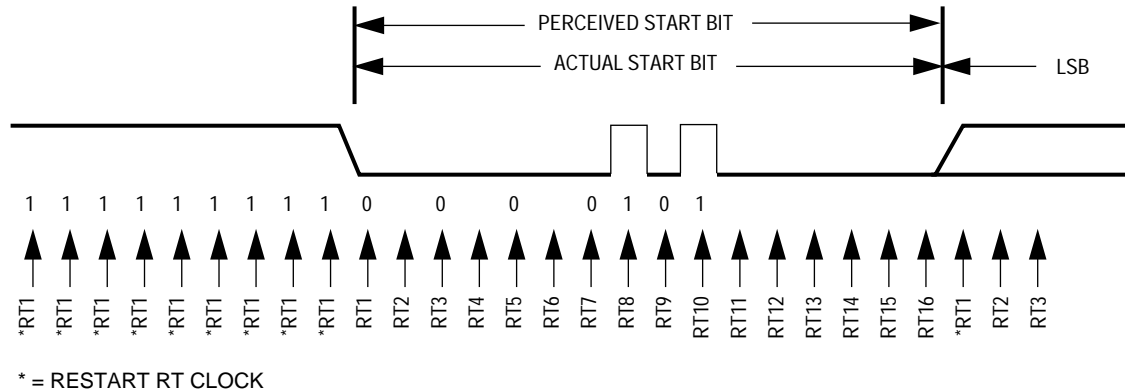
Figure 8-7. Start Search Example 7

To understand why start bit times do not obey the majority sampling of RT8, RT9, and RT10, consider what would happen if the start bit noise shown in example 7 occurred at a time when the sum of logic sense one bit times at the end of the preceding frame and the number of logic sense one bit times at the beginning of this frame was greater than the number of bits in a frame. The wake-up logic would erroneously wake up a receiver that did not force the start bit to be seen as a zero. Also, consider that in order for the start bit to have been recognized, at least three of the samples taken at RT1, RT3, RT5, and RT7 had to be lows. Even if two of the samples taken at RT8, RT9, and RT10 were high, then there would still be a majority of all samples taken during the start bit time in favor of a logic zero decision.

8.3.2 Receiver Functional Operation

This receiver includes a receive serial shift register and a parallel receive data register (This is called a double buffered system because after a complete character is shifted in serially it is immediately moved to a parallel register so that the subsequent character can be shifted in without requiring the CPU to immediately service the first character.). The receive serial shift register is internal to the receive logic and may not be read or written directly by the CPU. The input of this serial shifter is connected to the majority sampling logic of the front end.

The receiver can operate in either of two formats as specified by the M control bit in the SCCR1 register. The most common standard word format for NRZ serial communication is one start bit (logic zero or space) followed by eight data bits (LSB first) followed by one stop bit (logic one or mark). In addition to this standard format this circuit provides hardware for a nine data bit format as follows: one start bit, eight data bits (LSB first), ninth data bit, and one stop bit. If the nine data bit mode is selected, software control (and overhead) of the ninth bit may be used to support a number of special formats. The ninth data bit is positioned as R8 in the SCCR1 register. Some examples of its use include:

- start, eight data, two stop bits

- start, eight data, parity, one stop
  w/ odd, even, mark, or space parity

- start, seven data, parity, two stop bits
  w/ odd, even, mark, or space parity

- start, eight data, address/control, one stop bit
  where address/control bit identifies special command words

The receive logic is enabled when the receive enable (RE) bit in the SCCR2 register is set to one. When RE is zero the receive logic is initialized and most of the receiver front end logic is disabled. The receiver front end logic drives a state machine (running off the RT clock) and provides the derived logic level for each bit time. This state machine controls when the front end logic is to sample the RXD pin and controls when data is to be passed to the receive serial shift register. Data is shifted into the receive serial shift register according to the most recent synchronization of the RT clock. From this point on in the discussion, data reception may be considered to be synchronous.

The logic sense of each bit in the frame is determined from the majority of three samples taken near the middle of the bit time except the start bit which is forced to be shifted in as a zero regardless of the result of the majority sampling logic (see the discussion of the receiver front end logic). The next eight bits shifted in are the basic data byte (LSB is shifted in first). The next bit shifted in depends on the mode selected by the M bit in SCCR2. If the nine data bit format is selected, the next bit received after the MSB is the ninth data bit. It will be transferred to its appropriate

position in R8 at the same time the lower 8 bits of data are transferred from the receive serial shift register to SCDR.

The last bit to be shifted in for each frame is the stop bit, which should be a logic one. If a logic zero bit time is sensed where a stop bit was expected, it is called a Framing Error. The framing error is usually caused by mismatched baud rates between the transmitter and receiver, or by noise that caused the start bit to be missed so that the frame was synchronized incorrectly. It should be noted that it is possible for a framing error to go undetected because there is a chance that the data sampled where the stop bit was expected could have been a logic one anyway.

When the stop bit is received, the frame is considered to be complete and the received character in the receive serial shift register is normally transferred in parallel to the receive data register (RDR). Several other actions occur at the same time this transfer is taking place. The abnormal case where the RDRF flag in the SCSR register is set at the time the transfer was to occur is called an overrun because a new data character was received from the serial line before a previously received character was serviced by the CPU. No transfer to RDR is allowed while RDRF or OR is set. Parallel transfers and associated actions to status bits occur at a time that will not interfere with CPU access to the affected registers.

All status flags associated with a serially received frame are simultaneously set. When a complete frame has been serially received either the receive data register full (RDRF) or the over run (OR) status flag always will be set. When the receive interrupt enable (RIE) control bit in the SCCR2 register is set to one, a hardware interrupt request will result if either RDRF or OR is set indicating reception of a new frame of data. The receive status bits noise flag (NF) and framing error (FE) do not separately cause hardware interrupts because they are never set without being accompanied by RDRF. The NF and FE flags are always associated with the data in the RDR so they never get set with OR but do get set with RDRF if the associated frame had the corresponding error(s).

An automatic clearing mechanism is associated with the receiver status bits. The mechanism involves reading the SCSR register followed by reading the RDR register (SCDR — read). When the RDR is read, any of the receive status bits (RDRF, IDLE, OR, NF, or FE) that were set when the SCSR register was read will automatically be cleared.

8.3.3 Idle Line Detect

The receive logic hardware includes the ability to detect an idle line. This can be useful in a system to indicate when one message was finished and another was about to be started. An idle line is defined as a minimum of 10 (11 if nine data bit format is selected) bit times of continuous logic ones on the RXD line. During a normal message, there is no idle time between frames, so even if all information bits in a frame were logic ones the start bits would ensure that at least one logic zero bit time occurred for each frame and IDLE would not get set.

When the RXD line goes idle for the minimum required time, the IDLE status bit in SCSR gets set. If the Idle Line Interrupt Enable (ILIE) bit in the SCCR1 register is set then a hardware interrupt sequence also will be requested when IDLE gets set. The IDLE status bit is cleared by reading the SCSR register (with IDLE set) followed by reading the RDR register. IDLE will not be set again until after at least one character is received. This prevents an RXD line that remains idle for a long period of time from causing several interrupts.

8.3.4 Receiver Wakeup

The receiver logic hardware also supports a receiver wake-up function intended for systems having more than one receiver. With this function, the transmitting device directs messages to an individual receiver or group of receivers by passing addressing information as the initial byte(s) of each message. Receivers not addressed invoke the receiver wake up function which puts these receivers in a dormant state for the remainder of the unwanted message. This eliminates any further software overhead to service the remaining characters of the unwanted message and thus improves performance.

The receiver is placed in wake-up mode by writing a one to the receiver wake-up (RWU) bit in the SCCR2 register. While RWU is set, all of the receive status flags (RDRF, IDLE, OR, NF, and FE) are inhibited (cannot be set). Note that specification of receiver wake-up mode inhibits the use of the idle line detect function. Although RWU may be cleared by a write to SCCR, it is normally left alone by software and gets cleared automatically by hardware with one of the two methods explained in the following paragraphs.

The SCI offers a choice of two methods for waking up receivers from the dormant state. The first method is called idle line wake up. The second method is called address mark wake up and operates by using the MSB to differentiate between address information (MSB set) and data information (MSB clear). The WAKE control bit in the SCCR1 control register is used to select which method of automatic hardware wake up is to be used. If the WAKE bit is clear, Idle Line wake up is specified and if WAKE is set to one, address mark wake up is selected.

8.3.5 Idle Line Wakeup

To use this receiver wake up method in an actual system, a software device addressing scheme is established to allow the transmitting device(s) to direct messages to individual receivers or groups of receivers. This addressing scheme is purely a software device and may take any form as long as all transmitting or receiving devices are programmed to understand the same scheme. The addressing information is usually the first frame(s) in a message so that uninterested receivers are burdened with only these minimum addressing frames. All receivers are awake (RWU bit is clear) when each message begins and as soon as a receiver determines that the message is not intended for it, it sets its RWU bit which inhibits flag setting until the RXD line goes idle at the end of the message. As soon as an idle line is detected (in hardware) while the RWU bit is set, the receiver hardware will force the RWU bit to zero so that the first frame in the next message can be received. This method of receiver wake up requires that a minimum of one frame time of idle line be imposed between messages and that no idle time is allowed between frames within a message.

8.3.6 Address Mark Wakeup

In this method of receiver wake up, all serial frames consist of seven (or eight) information bits plus an MSB which is used to indicate an address frame (when set to one — "mark"). The first frame(s) of each message are addressing frames and all receivers in the system evaluate these marked address frames to determine whether or not the subsequent message is intended for a particular receiver. As soon as a receiver determines that a message is not intended for it, it invokes the receiver wake-up function (by setting RWU) so that no additional software overhead is required for the rest of the message. When the next message begins, its first frame will have the ninth bit set, which will automatically clear the RWU bit and enable normal frame reception. The first frame whose ninth bit is set (address marked) also will be the first frame to be received after wake up because RWU gets cleared before the stop bit for that frame is serially received. This method allows messages to include idle times, unlike the first wake-up method, but efficiency is lost due to the extra bit time (address bit) which is required in all frames.

8.4 SCI Register Descriptions

The following subsections provide a description of the SCI registers.

8.4.1 SCI Baud Rate Control Register

The value in this register determines the baud rate of the SCI. SCP1:0 set the prescaler and SCR2:0 set the final divider chain.The desired baud rate is determined by:

baud = bus clock divided by 16 divided by (SCP1:SCP0) divided by (SCR2:SCR0).

	Bit 7	6	5	4	3	2	1	Bit 0
$000D Read:	0	0	SCP1	SCP0	0	SCR2	SCR1	SCR0
BAUD Write:								
Reset:	0	0	0	0	0	U	U	U

= Unimplemented   U = Unaffected
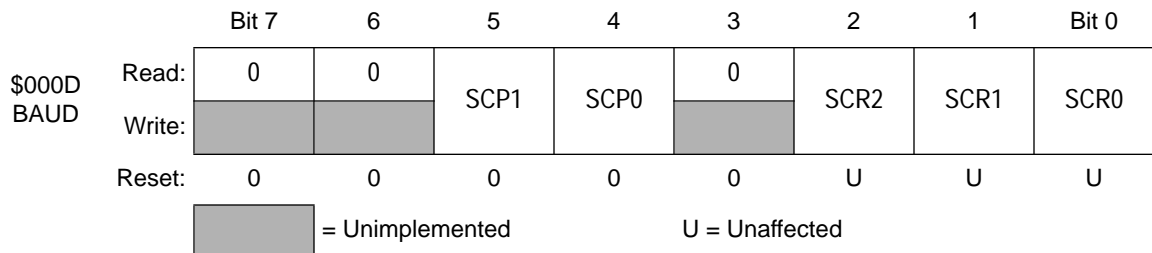
Figure 8-8. SCI Baud Rate Control Register

Read: anytime
Write: anytime

SCP1:SCP0 — SCI Prescaler

These bits control the prescaler as shown in Table 8-1.

Table 8-1. SCP1:SCP0 Select

SCP Bit		E*/16 Divided by	Crystal Frequency (f_{osc}) MHz				
1	**0**		**4.194304**	**4.0**	**2.4576**	**2.0**	**1.8432**
0	0	1	131.072 kHz	125.000 kHz	76.80 kHz	62.50 kHz	57.60 kHz
0	1	3	43.691 kHz	41.666 kHz	25.60 kHz	20.833 kHz	19.20 kHz
1	0	4	32.768 kHz	31.250 kHz	19.20 kHz	15.625 kHz	14.40 kHz
1	1	13	10.082 kHz	9600 kHz	5.907 kHz	4800 kHz	4430 Hz

* E is the Internal Bus Clock (OSC/2)

SCR2:SCR0 — SCI Rate Select

These three bits can be used to divide further the output of the prescaler. See Table 8-2.

Table 8-2. SCR2:SCR0 Select

SCR Bit			Prescaler Divided by	Representative Highest Prescaler Baud Rate Output				
2	**1**	**0**		**131.072 kHz**	**32.768 kHz**	**76.80 kHz**	**19.20 kHz**	**9600 Hz**
0	0	0	1	131.072 kHz	32.768 kHz	76.80 kHz	19.20 kHz	9600 Hz
0	0	1	2	65.536 kHz	16.384 kHz	38.40 kHz	9600 Hz	4800 Hz
0	1	0	4	32.768 kHz	8.192 kHz	19.20 kHz	4800 Hz	2400 Hz
0	1	1	8	16.384 kHz	4.096 kHz	9600 Hz	2400 Hz	1200 Hz
1	0	0	16	8.192 kHz	2.048 kHz	4800 Hz	1200 Hz	600 Hz
1	0	1	32	4.096 kHz	1.024 kHz	2400 Hz	600 Hz	300 Hz
1	1	0	64	2.048 kHz	512 Hz	1200 Hz	300 Hz	150 Hz
1	1	1	128	1.024 kHz	256 Hz	600 Hz	150 Hz	75 Hz

8.4.2 SCI Control Register 1 (SCCR1)

		Bit 7	6	5	4	3	2	1	Bit 0
$000E SCCR1	Read: Write:	R8	T8	0	M	WAKE	0	0	0
	Reset:	U	U	0	U	U	0	0	0

         ▨ = Unimplemented   U = Unaffected
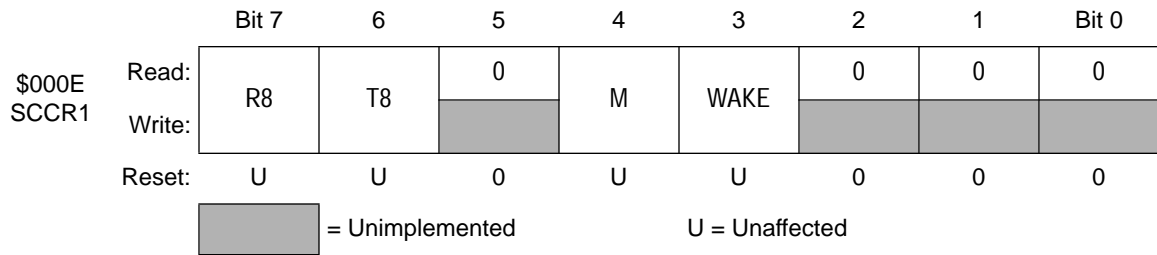
Figure 8-9. SCI Control Register 1

Read: anytime

Write: anytime

R8 — Receiver Bit 8

    This bit provides storage for the ninth bit in the receive data byte (if M = 1).

T8 — Transmit Bit 8

    This bit provides storage for the ninth bit in the transmit data byte (if M = 1).

M — Mode (select character format)

    Read: anytime

    Write: anytime
        0 = 1 start, 8 data, 1 stop bit
        1 = 1 start, 8 data, ninth data, 1 stop bit

WAKE — Wake Up by Address Mark/Idle

    Read: anytime

    Write: anytime
        0 = wake up by IDLE line recognition
        1 = wake up by address mark (8th or 9th -last- data bit set)

8.4.3 SCI Control Register 2 (SCCR2)

		Bit 7	6	5	4	3	2	1	Bit 0
$000F SCCR2	Read: Write:	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
	Reset:	0	0	0	0	0	0	0	0

Figure 8-10. SCI Control Register 2

Read: anytime

Write: anytime

TIE — Transmit Interrupt Enable

  When this control bit is set to a one, an SCI interrupt will be requested whenever the TDRE status flag is set.

TCIE — Transmit Complete Interrupt Enable

  When this control bit is set to a one, an SCI interrupt will be requested whenever the TC status flag is set.

RIE — Receiver Interrupt Enable

  When this control bit is set to a one, an SCI interrupt will be requested whenever the RDRF status flag or the OR status flag is set.

ILIE — Idle Line Interrupt Enable

  When this control bit is set to a one, an SCI interrupt will be requested whenever the IDLE status flag is set.

TE — Transmitter Enable

  When this control bit is set, the SCI transmit logic is enabled and the TXD pin (port D bit 1) is dedicated to the transmitter. The TE bit can be used to queue an idle preamble.

RE — Receiver Enable

  This control bit (when set) enables the SCI receive circuitry.

RW — Receiver Wake-up Control

  When set this control bit enables the wake-up function and inhibits further receiver interrupts. Normally, hardware wakes the receiver by automatically clearing this bit.

SBK — Send Break

To generate a break code (at least 10 or 11 contiguous zeros), a one is written into this bit. As long as SBK remains set to a one, the transmitter will send zeros. If SBK is toggled on and off, the transmitter will send only 10 (or 11) zeros and then revert to mark idle or sending data.

8.4.4 SCI Status Register (SCSR)

		Bit 7	6	5	4	3	2	1	Bit 0
$0010 SCSR	Read:	TDRE	TC	RDRF	IDLE	OR	NF	FE	0
	Write:								
	Reset:	1	1	0	0	0	0	0	0

= Unimplemented

Figure 8-11. SCI Status Register

The bits in this register are set by various conditions in the SCI hardware and are automatically cleared by special acknowledge sequences. The receive-related flag bits in SCSR (RDRF, IDLE, OR, NF, and FE) are all cleared by a read of the SCSR register followed by a read of the Transmit/Receive Data Register. However, only those bits which were set when SCSR was read will be cleared by the subsequent read of the Transmit/Receive Data Register. The transmit-related bits in SCSR (TDRE and TC) are cleared by a read of the SCSR register followed by a write to the Transmit/Receive Data Register.

Read: anytime (used in auto clearing mechanism)

Write: has no meaning or effect

TDRE — Transmit Data Register Empty Flag

This bit is set when the byte in the transmit data register is transferred to the serial shift register. New data will not be transmitted unless the SCSR register is read before writing to the transmit data register. Reset sets this bit.

TC — Transmit Complete Flag

This bit is set to indicate that the SCI transmitter has no meaningful information to transmit (no data in shifter, no preamble, no break). When TC is set, the serial line will go idle (continuous mark). Reset sets this bit.

RDRF — Receive Data Register Full Flag

This bit is set when the contents of the receiver serial shift register is transferred to the receiver data register.

IDLE — Idle Line Detected Flag

This bit is set when a receiver idle line is detected (the receipt of a minimum of 10/11 consecutive ones). This bit will not be set by the idle line condition when the RWU bit is set. Once cleared, IDLE will not be set again until after RDRF has been set (until after the line has been active and becomes idle again).

OR — Over-Run Error Flag

This bit is set when a new byte is ready to be transferred from the receiver shift register to the receiver data register and the receive data register is already full (RDRF bit is set). Data transfer is inhibited until this bit is cleared.

NF — Noise Error Flag

This bit is set if there is noise on a "valid" start bit, any of the data bits, or on the stop bit. The NF bit is set during the same cycle as the RDRF bit but does not get set in the case of an over run (OR).

FE — Framing Error Flag

This bit is set when the word boundaries in the bit stream are not synchronized with the receiver bit counter (generated by the reception of a logic zero bit where a stop bit was expected). The FE bit reflects the status of the byte in the receive data register and the transfer from the receive shifter to the receive data register is inhibited in the case of over run. The FE bit is set during the same cycle as the RDRF bit but does not get set in the case of an over run (OR). The framing error flag inhibits further transfer of data into the RDR until it is cleared.

PF — Parity Error Flag

This bit is set when the received data's parity does not match its parity bit. This feature is active only when parity is enabled. The type of parity tested for is determined by the PT (parity type) bit in SCCR1.

8.4.5 SCI Data Register (SCDR)

		Bit 7	6	5	4	3	2	1	Bit 0
$0011 SCDR	Read: Write:	SCD7	SCD6	SCD5	SCD4	SCD3	SCD2	SCD1	SCD0

Unaffected by Reset
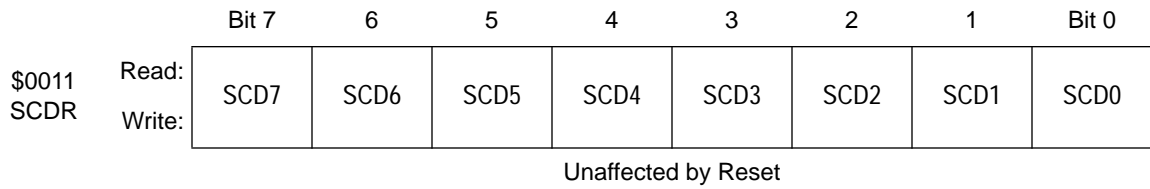
Figure 8-12. SCI Data Register

SCD7:SCD0 Serial Data Bits 7 to 0

Read: reads access the 8 bits of the read-only SCI receive data register (RDR)

Write: writes access the 8 bits of the write-only SCI transmit data register (TDR)

Reset: does not affect this address

SECTION 9
SERIAL PERIPHERAL INTERFACE

9.1 Introduction

The term "serial peripheral" refers to the fact that this interface requires separate wires (signals) for data and clock. In this format, data does not contain an explicit clock. The SPI scheme may be used to interconnect microcomputers located at a short distance (usually within a single "black box" or on the same PC card). This may comprise a system of one microcomputer and several slaves or may be a system of microcomputers, each having the capability of either master or slave.

A practical system may include:

1. MISO master in slave out

2. MOSI master out slave in

3. SCK1 serial clock

4. SS(n) slave select(s)

This device also includes a programmable direction of data to select either MSB or LSB first format.

9.2 Signal Description

The following subsections provide a description of the SPI signals.

9.2.1 Master In Slave Out (MISO)

In slave mode, MISO is the signal which is used to present data from a slave device to the bus. The MISO pin will be placed in the high-impedance state when ever a slave device is "not" selected (SS = 1) by the bus master. See Figure 9-2 for more information. Four possible timing relationships may be chosen by use of the control bits (CPOL) and (CPHA). The slave device and a master device must be programmed to be in similar timing modes for proper data transfer.

In the master mode (control bit MSTR = 1), the function of MOSI and MISO are inverted within the device. Therefore, the MISO pin becomes the data input pin for a device which is in the master mode. (Also, the function of SCK1 switches from being an input for system clock to one of outputting the system clock.)

9.2.2 Serial Data In (MOSI)

In slave mode, MOSI is the signal used to receive data from some master device. Figure 9-2 shows the serial clock and data timing relationship.

It should be noted that when a master device transmits data to a second device via the MOSI line, the slave device (if it has the capability) will respond by sending data into the MISO pin of the master device. This implies full duplex transmission with both data out and data in synchronized to the same clock signal which is provided by the master. Moreover, the SAME shift register is used for data out and data in. Thus, the byte transmitted is replaced by the byte received, removing the need for separate status bits for XMIT EMPTY and REC FULL. A single status bit, SPIF, is used to signify I/O operation complete.

In the master mode, as noted above, the functions of MOSI and MISO are inverted. The MOSI pin becomes the data output pin when the device is in the master mode. When a transfer of data is not taking place with a slave device the master drives the MOSI line high. The master always allows the data onto the MOSI pin a half-cycle before the clock edge (SCK1) needed for the slave to latch the data internally.

9.2.3 Serial Clock In/Out (SCK1)

In slave mode, the serial clock is used to move data both in and out of the device through its MOSI and MISO pins. The master and slave devices are capable of exchanging a byte of information during a sequence of eight clock pulses if wired to do so. In the slave mode, the SCK1 pin becomes an input being sent from the master device for the external clock. In this case SCK1 is asynchronous to the slave device's phase 1-2 clocks and read/write control, therefore synchronization must take place prior to transmission and after reception. This must be done on the byte level. The type of clock and its relationship with the data is controlled by bits CPOL and CPHA. Reference Figure 9-2. The clock rate control bits SPR1 and SPR0 have no function while the part is in the slave mode.

In master mode, the clock is generated within the master device by a circuit driven from the bus clock. The clock rate is selected by bits (SPR1, SPR0) in the control register. The SCK1 pin on the master device becomes a fixed output providing the system clock to an enabled slave or slaves. The clock is used by the master to latch incoming slave data on the MISO pin and shift out data to the slave device on the MOSI pin. The master and slave must be operated in the same timing mode. The type of clock and its relationship with the data is controlled by bits CPOL and CPHA. Reference Figure 9-2.

9.2.4 Slave Select ($\overline{SS}$)

In slave mode, the slave select ($\overline{SS}$ input) is generated by the master (parallel port may be used) and used to "enable one" of several slaves to accept and/or return data or "enable several" slaves to accept data. To insure a data byte transfer, the $\overline{SS}$ signal must be low prior to occurrence of SCK1 and must not become high until after the 8th (last) SCK1 cycle. Figure 9-2 shows the clock (SCK1) and data relationship. Depending on the state of the CPHA control bit, the $\overline{SS}$ pin pulled low: (1) allows the first bit of data onto the MISO system line for transfer and (2) prevents the slave from reading or writing the data register. A further description of the effect of the ($\overline{SS}$) pin and (CPHA) control bit on the I/O data register is given in the description of the (WCOL) status flag. The (WCOL) flag warns the slave if it has had a conflict between a transmission and a write of the data register. A high level on $\overline{SS}$ forces MISO to the high-impedance state. Also, SCK1 and MOSI are ignored by the disabled slave.

In master mode, slave select ($\overline{SS}$) input is monitored to assure that it stays false (high). If slave select becomes true, the device immediately exits the master mode and becomes a slave (MSTR = 0). Also, control bit (SPE) is forced to a zero causing all SPI system pins to be inputs. An interrupt flag (MODF) is set warning the device that the above events have occurred. The significance of this is that a collision has occurred; that is, two devices have both become masters. This is normally the result of software error, although some systems may allow the default master to "knock all other masters off the bus" if an erroneous bus state is detected. This is, of course, a catastrophic event and it is the responsibility of the default master to completely "clean up" the system.

9.3 SPI Registers

The following subsections describe the SPI registers.

9.3.1 SPI Control Register (SPCR)

		Bit 7	6	5	4	3	2	1	Bit 0
$000A SPCR	Read: Write:	SPIE	SPE	DOD	MSTR	CPOL	CPHA	SPR1	SPR0
	Reset:	0	0	0	0	U	U	U	U

U = Unaffected

Figure 9-1. SPI Control Register

Read: anytime

Write: anytime

SPIE — SPI Interrupt Enable

When this bit is set to a one a hardware interrupt sequence is requested each time the SPIF or MODF status flag is set. SPI interrupts are inhibited if this bit is clear or if the I bit in the condition code register is one.

SPE — SPI System Enable

When the SPE bit is set the port D bits 2, 3, 4, and 5 are dedicated to the SPI function.

DOD — Direction Of Data

This bit determines the direction of data flow in or out of the serial shift register. When set, data is transferred LSB first. When cleared (the default state), data is transferred MSB first.

NOTE

Figure 9-2 assumes a value of zero for this bit.

MSTR — Master/Slave Mode Select
       0 = Slave mode
       1 = Master mode

CPOL — Clock Polarity

CPHA — Clock Phase

These two bits are used to specify the clock format to be used in SPI operations. Refer to Figure 9-2.

SPR1, SPR0 — SPI Clock (SCK1) Rate Select Bits

These bits are used to specify the SPI clock rate.

Table 9-1. SPI Clock Rates

SPR1	SPR0	E Clock Divided-By	Frequency at E = 2 MHz (Baud Rate)
0	0	2	1.0 MHz
0	1	4	500 kHz
1	0	16	125 kHz
1	1	32	62.5 kHz
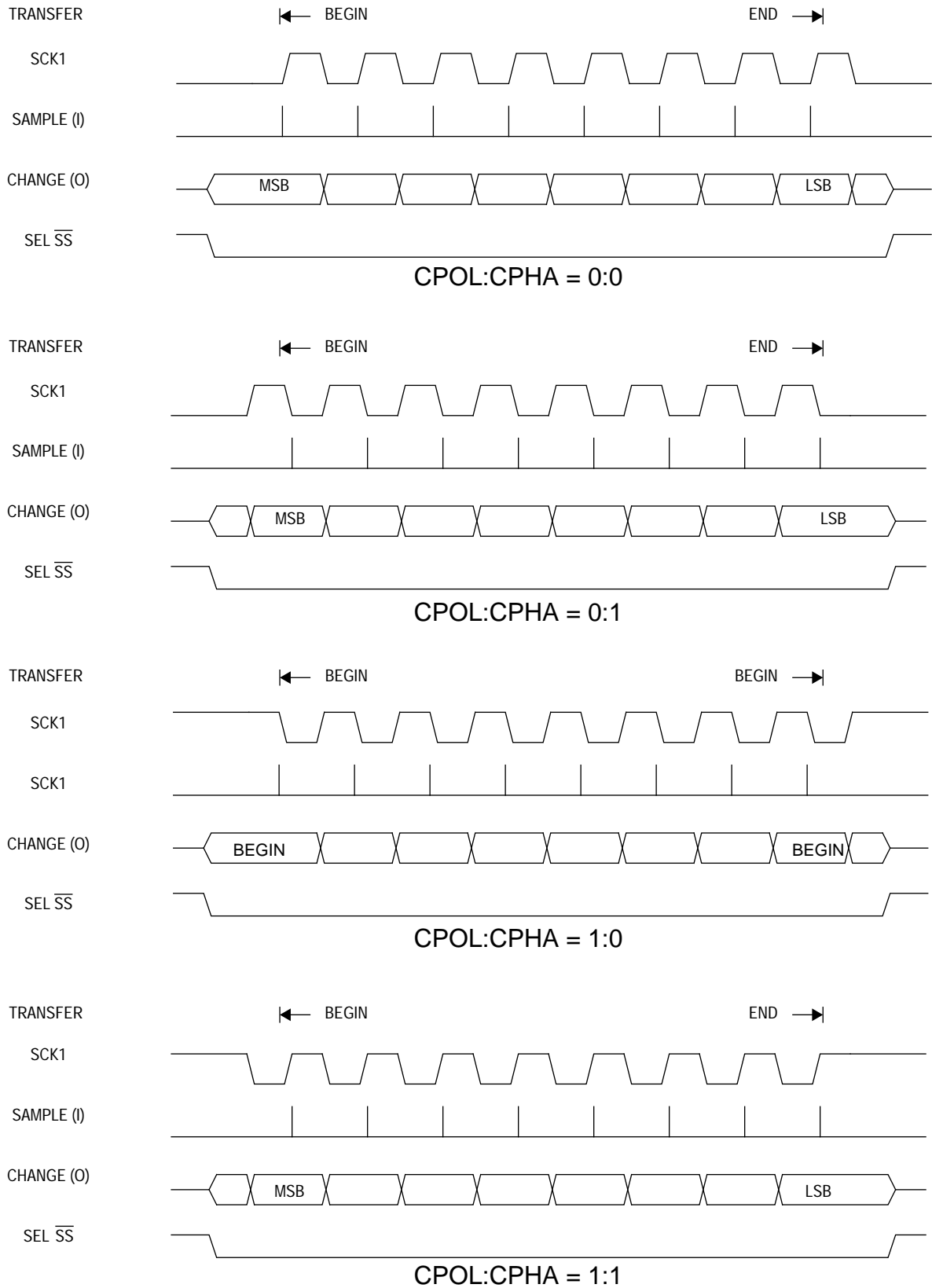
Figure 9-2. SPI Clock/Data Relationships

9.3.2 SPI Status Register (SPSR)

		Bit 7	6	5	4	3	2	1	Bit 0
$000D SPSR	Read:	SPIF	WCOL	0	MODF	0	0	0	0
	Write:								
	Reset:	0	0	0	0	0	0	0	0
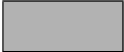
     = Unimplemented

Figure 9-3. SPI Status Register

Read: anytime

Write: has no meaning or effect

SPIF — SPI Interrupt Request

 SPIF is set after the eighth SCK1 cycle in a data transfer and it is cleared by reading the SPSR register (with SPIF set) followed by an access (read or write) to the SPI data register.

WCOL — Write Collision Status Flag

 This error status flag is used to indicate that a serial transfer was in progress when the MCU tried to write new data into the SPDR data register. The MCU write is disabled to avoid writing over the data being transmitted. No interrupt is generated because the error status flag can be read upon completion of the transfer that was in progress at the time of the error. This flag is automatically cleared by a read of the SPSR (with WCOL set) followed by an access (read or write) to the SPDR register.

MODF — SPI Mode Error Interrupt Status Flag

 This bit is set automatically by SPI hardware if the MSTR control bit is set to one and the slave select input pin becomes zero. This condition is not permitted in normal operation. This flag is automatically cleared by a read of the SPSR (with MODF set) followed by a write to the SPCR register.

9.3.3 SPI Data Register (SPDR)

		Bit 7	6	5	4	3	2	1	Bit 0
$000D SPSR	Read:	SPD7	SPD6	SPD5	SPD4	SPD3	SPD2	SPD1	SPD0
	Write:								
	Reset:				Unaffected by Reset				
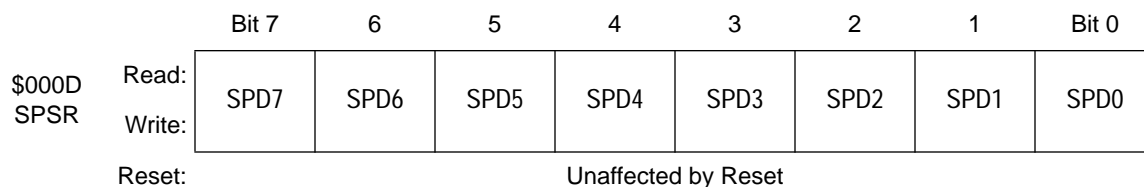
Figure 9-4. SPI Control Register

Read: anytime (normally only after SPIF flag set)

Write: anytime (see WCOL write collision flag)

Reset: does not affect this register

This 8-bit register is both the input and output register for SPI data. In the SPI system the 8-bit data register in the master and the 8-bit data register in the slave are linked by the MOSI and MISO wires to form a distributed 16-bit register. When a data transfer operation is performed, this 16-bit register is serially shifted eight bit positions by the SCK1 clock from the master so the data is effectively exchanged between the master and the slave. Note that some slave devices are very simple and either accept data from the master without returning data to the master or pass data to the master without requiring data from the master.

When writing the SPDR, the data is written directly into the shift register and always shifted out in the same direction. To affect a change in the direction of data transfer, the data is loaded into the shift register in reverse order. For this reason, the DOD bit must be written before data is loaded into the shift register

When reading the SPDR, a read data buffer is actually accessed. This buffer contains the last data byte received by the SPI, and is updated during the cycle that SPIF is set (reception complete).

SECTION 10
SLAVE-ONLY M-BUS

10.1 Introduction

Motorola bus (M-bus for short) is a two-wire bidirectional bus which provides a simple, efficient method of providing data exchange between devices. It is fully compatible to the I^2c bus standard.This device will contain a reduced version of the M-bus supporting only the slave mode. Slave address recognition is provided in hardware with the value of the address being initialized by the user. Also included is automatic generation of the acknowledge for both address and data, general call address recognition, and automatic clock stretching.

10.2 Operation of SOMB and Ninth-Bit Detector

The SOMB consists of an 8-bit synchronous shift register, a start/stop detect circuit, a ninth-bit detector, and acknowledge detector. The Serial DAta line (SDA) and Serial CLock line (SCL) share external connections with two bits of port D. These bits are not enabled during serial operation. The operation of the SOMB is best described in phases as shown in the following sub-sections.

10.2.1 After Reset

- After the MCU has completed the reset operation, the SOMB is in the following condition:

- SME = 0. The SOMB is disabled and the parallel I/O are enabled. By definition, SMIE and SMF also = 0.

- Data = ?. The contents of the data register is unknown.

To enable the SOMB, set SME and SMIE. This will allow the CPU to be interrupted upon reception of the slave address and/or serial data.

10.2.2 First Reception

The first reception must be proceeded by the start condition. By definition, the first reception consists of the slave address (seven bits) and the R/W bit. When the start condition is detected, the SCL input is enabled and the slave's address is transferred from the address register into the shift register. On the first clock pulse the external serial data (MSB of the master address) is shifted into the shift register,

and the first bit of the slave's address **(already in the shifter)** is shifted out. These two address bits are compared and if a match occurs, a counter is incremented. If a match does not occur, the counter is **not** incremented. This continues for the duration of the seven bits of address. After the **eighth** clock pulse, the shift register stops shifting. The SMF bit is set after the **ninth** clock pulse.

10.2.3 After the First Reception

When SMF is set, an interrupt request is generated to the CPU based on the following conditions:

- A START condition was recently detected.
- The address match counter equals seven
- SMIE is set

If the address match counter is less than seven, this means that the slave is not being addressed. In that case, the start condition and SMF are cleared and the SCL input is disabled. The SOMB will ignore all further clock pulses until a subsequent start condition is detected.

If a start condition was not detected on this transmission, then the address match counter is ignored and the data is treated as data.

If the address is matched and the start was detected, the acknowledge is accomplished by forcing the SDA line low (in hardware) and enabling the ninth-bit detector on the clock line. When the ninth clock pulse is detected on the clock line, the SDA line is released automatically and the SCL line is forced low to stretch the clock. *When the serial service routine is completed*, *the user* **must** *release the clock line by writing a '0' to CLKR.* This serves as a mechanism for releasing the clock.

NOTE

The M-bus address/data register (MBADR) now has the received data and bit zero contains the R/W bit from the master. This should be checked in software and the T/R bit in the slave should be set accordingly.

The SOMB is now enabled and prepared to receive further data. It will stay in this mode until another start condition is detected, at which time this process will begin again.

10.2.4 Subsequent Receptions

After the first reception, The SMF bit should be cleared by reading the status register followed by reading or writing the data register, This action clears the serial interrupt and enables the circuit to receive additional data. Receiving and acknowledging the data is accomplished in the same way as the first reception. Stretching the clock is done automatically after every transmission and must be released by the user each time. If the T/R bit is set, data is transmitted, if it is cleared data is received. The data register should be serviced before releasing the clock line in order to avoid data collision.

10.2.5 Acknowledgment

After address detection and data reception, an acknowledge is returned to the master during the ninth clock pulse. By setting the NOACK bit in the slave M-bus control register, the acknowledge function will be disabled for the following transmissions.

During master read, an acknowledge is sent from the receiving master. The status of the acknowledge could be read from the MACK bit in the slave M-bus status register during the service routine. This bit is used for detecting the master's acknowledge and end of transmission. By definition, the end of transmission is signaled from the master by the absence of the master acknowledge. In this case, **the slave must clear the T/R** bit to release the SDA line. If the slave device does not release the SDA line, the master may not be able to generate the stop condition.

10.2.6 Stop Condition

The system is capable of detecting the stop condition. This is defined as the rising edge of the data line while the clock line is high. When this condition is detected, the system is brought to the state before the start condition.

10.2.7 General Call Address Detect

During address detect cycle (receiving cycle after the start detect), If the resulting address is $00, a condition of address match is generated. The user must determine whether this address match is a slave address match or a general call address match by reading the MBADR (slave MBUS address/data register).

10.3 Slave M-Bus Control Register (MBCR)

		Bit 7	6	5	4	3	2	1	Bit 0
$001D MBCR	Read:	SMIE	SME	T/R	NOACK	0	0	0	0/1
	Write:								CLKR
	Reset:	0	0	0	0	0	0	0	0/1

    = Unimplemented

Figure 10-1. M-Bus Control Register

SMIE — Slave M-Bus Interrupt Enable

When set, this bit enables the SOMB interrupt. There is only one source of interrupt which is from the SMF bit. If SMIE is set when SMF is set, an interrupt request is generated to the CPU. This bit is readable and writable and is cleared by reset.

SME — Slave M-Bus Enable

When set, this bit enables the SOMB. When clear, the circuit is disabled and all other status and control bits are cleared. If SME is cleared during a transmission, the transmission is aborted and all circuits are reset. This bit is readable and writable and is cleared by reset.

T/R — Transmit/Receive

When set, the SOMB is configured to transmit data. Upon reception of the SCL from the master, the data in the shift register is transmitted out MSB first. After the eighth clock pulse is received the ninth-bit detector is enabled. The SDA line is **not** driven low in this case, but following the ninth clock bit the SCL line **is** and SMF is set (an interrupt is generated if SMIE is set). The purpose of stretching the clock is to allow the device to complete its service routine. **The user must release the SCL line to allow additional transmissions.**

This bit is readable and writable and is cleared by reset.

NOACK — No Acknowledge

This bit, when set, will disable the acknowledge for the following transmissions. This bit is used in multiple byte data transmission to terminate further transmissions when invalid data was sent from the master. The master, with no acknowledge being sent, will abort the transmission. This bit is readable and writable, and is cleared by reset.

SLAVE-ONLY M-BUS

CLKR — Clock Release

This bit is used to release the SCL line after a complete transmission. This bit can never be set, and writing a logic zero to it releases the clock line to allow further communications.

NOTE

This bit reads a logic zero on the MC68HC705CJ4 and a logic one on the MC68HC05CJ4. Special care must be taken on the MC68HC705CJ4 as bit read-modify-write instructions may release the clock line.

The other bits in this register (1-3) are not implemented and always read as zero.

10.4 Slave M-Bus Status Register (MBSR)

		Bit 7	6	5	4	3	2	1	Bit 0
$001E	Read:	SMF	STDF	MACK	0	0	0	0	0
MBSR	Write:								
	Reset:	0	0	0	0	0	0	0	0
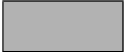
= Unimplemented

Figure 10-2. M-Bus Status Register

SMF — Slave M-Bus Flag

This bit is set after the **ninth** clock of a valid transmission. A valid transmission can be either a reception of a valid address following a start condition or transmission/reception of data after the a valid address is recognized. If SMIE is also set, an interrupt will be generated. This bit is cleared by first reading the MBSR with SMF set, followed by reading or writing the M-bus address/data register. SMF is also cleared when SME is cleared or by reset.

STDF — Start Detect Flag

This bit is set when a start condition is detected. It is provided as a means for the user to distinguish between an address and a data transmission. No interrupt is associated with this bit. Clearing STDF is done by first reading the MBSR with STDF set, followed by reading or writing the M-bus address/data register. STDF is also cleared when SME is cleared or by reset.

MACK — Master Acknowledge

This bit shows the status to the acknowledge during the master read (slave write) operation. A Master ACKnowledge happens during the ninth-bit of a slave transmission, when the slave releases the SDA line and the master holds it low. When an acknowledge is sent by the master, MACK will be set (logic one): If the master does not return any acknowledge, MACK will be cleared. This provides a mean to detect if a master is signalling an end of transmission to the slave. No interrupt is associated with this bit. Clearing MACK is done by first reading the MBSR with MACK set, followed by reading or writing the M-bus data register. MACK is also cleared when SME is cleared or by reset

---

NOTE

The other bits in this register (0–4) are not implemented and always read as zero.

10.5 M-Bus Address/Data Register (MBADR)

First access:

		Bit 7	6	5	4	3	2	1	Bit 0
$001C MBADR	Read:	MSR7	MSR6	MSR5	MSR4	MSR3	MSR2	MSR1	MSR0
	Write:	MBA7	MBA6	MBA5	MBA4	MBA3	MBA2	MBA1	MBR/W
	Reset:	U	U	U	U	U	U	U	U

U = Unaffected

Figure 10-3. M-Bus Address/Data Register

Subsequent accesses:

		Bit 7	6	5	4	3	2	1	Bit 0
$001C MBADR	Read:	MSR7	MSR6	MSR5	MSR4	MSR3	MSR2	MSR1	MSR0
	Write:	MBD7	MBD6	MBD5	MBD4	MBD3	MBD2	MBD1	MBD0
	Reset:	U	U	U	U	U	U	U	U

U = Unaffected

Figure 10-4. M-Bus Data Address/Register

This register has a dual purpose. The initial write to the MBADR must be the slave address. This data is actually stored in a buffer for use after a start condition has been detected. Subsequent writes go directly into serial shift register for transmission. The data register is not buffered. This means that reading or writing the MBADR during a transmission can corrupt the data. Reads of the MBADR always give back the contents of the shift register, which is undefined until after the first reception.

The address match is done for bits 1:7. Bit 0 is not compared and contains the R/W bit when read.

Reset has no effect on the data register, but the address buffer is cleared to $00 and MUST be initialized by the user.

10.6 Hardware Flowchart of the SOMB
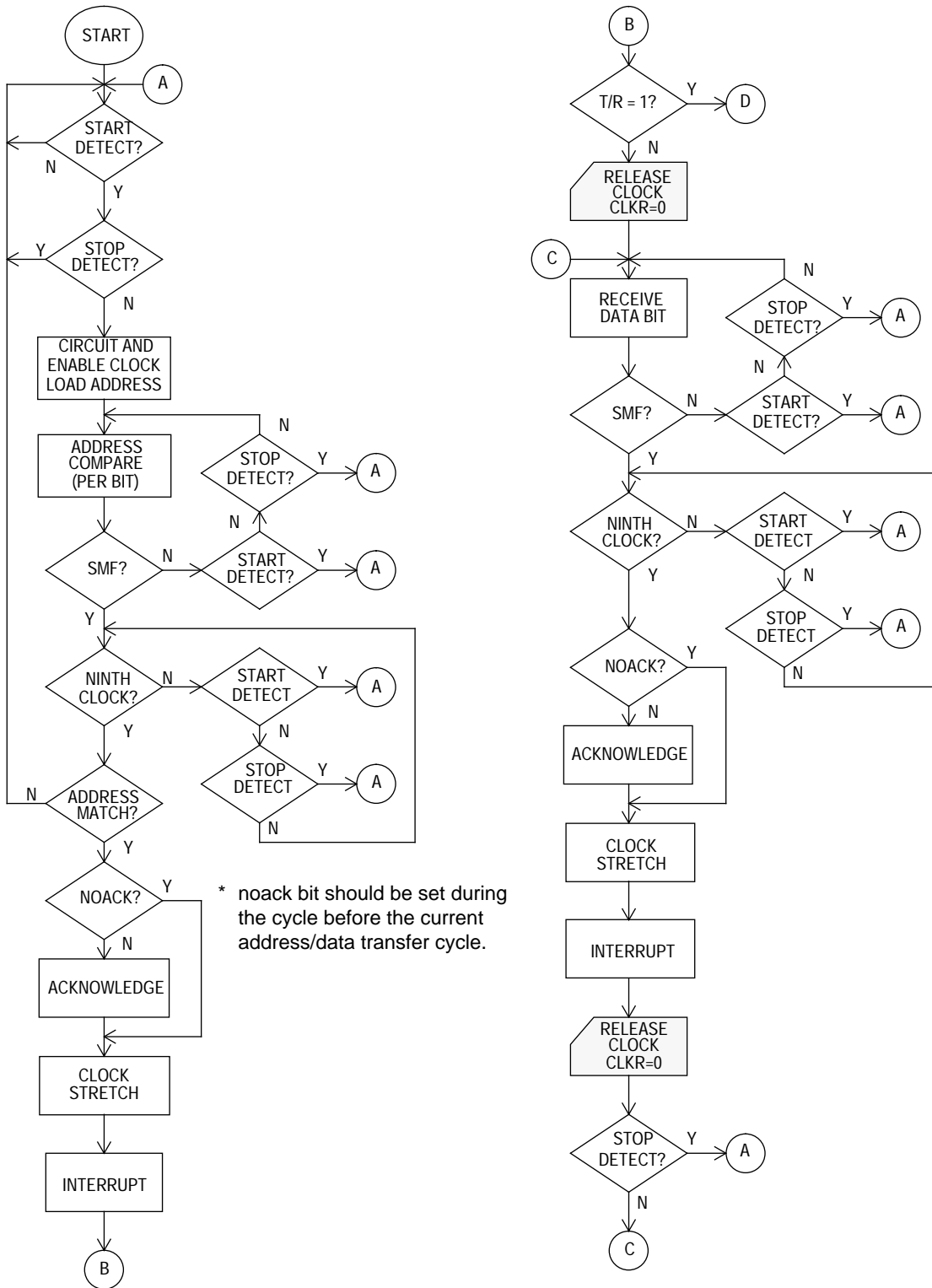
Figure 10-5. SOMB Flowchart (Sheet 1 of 2)

* noack bit should be set during the cycle before the current address/data transfer cycle.
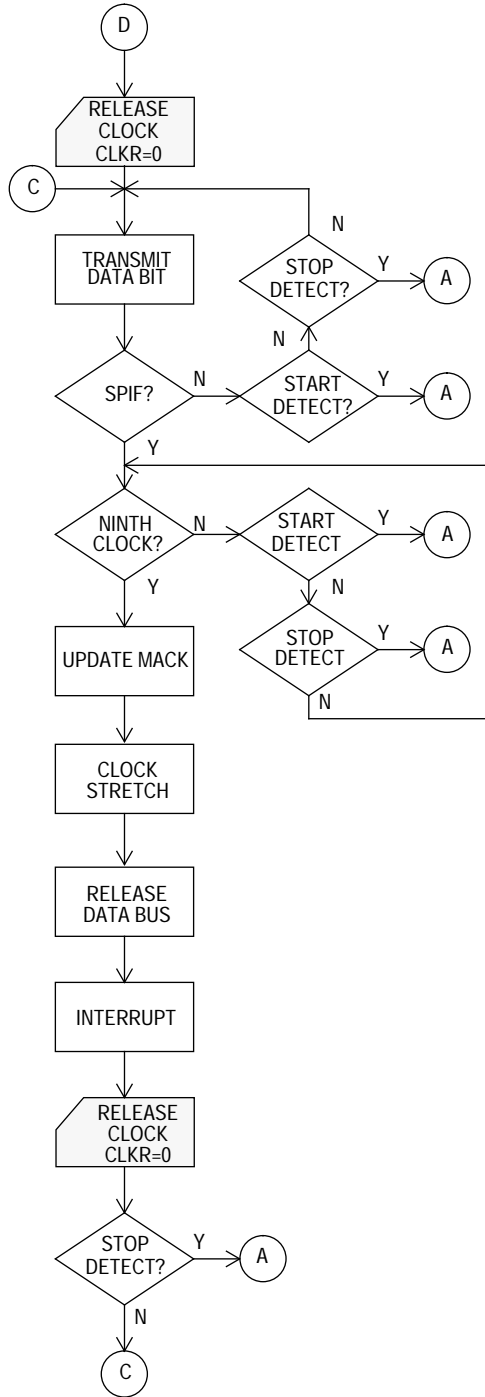
Figure 10-4. SOMB Flowchart (Sheet 2 of 2)

NOTE

Shaded area indicates an action of software.

10.7 SOMB Timing Diagrams

Refer to Figure 10-6, Figure 10-7, and Figure 10-8 for SOMB timing.
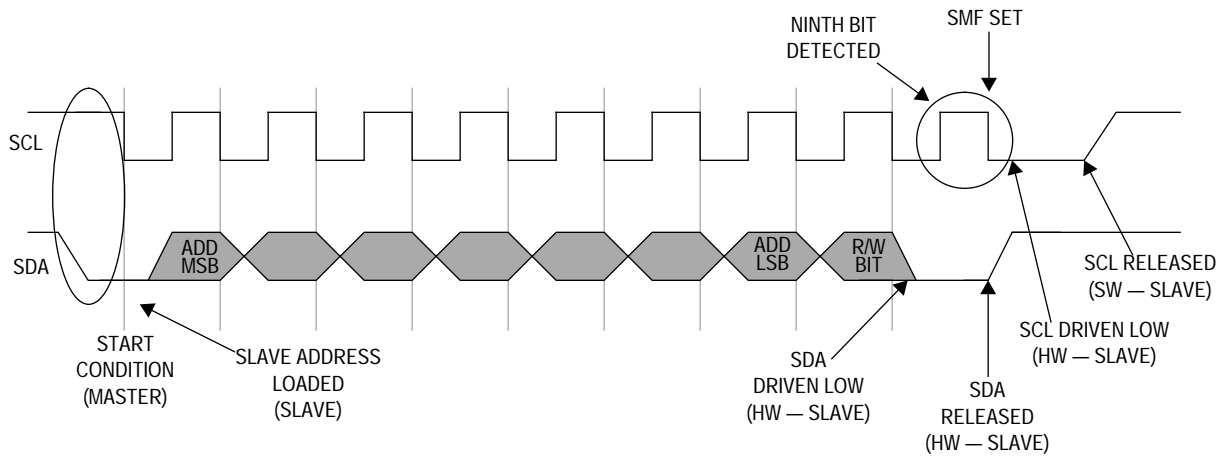
Figure 10-6. First Reception Timing
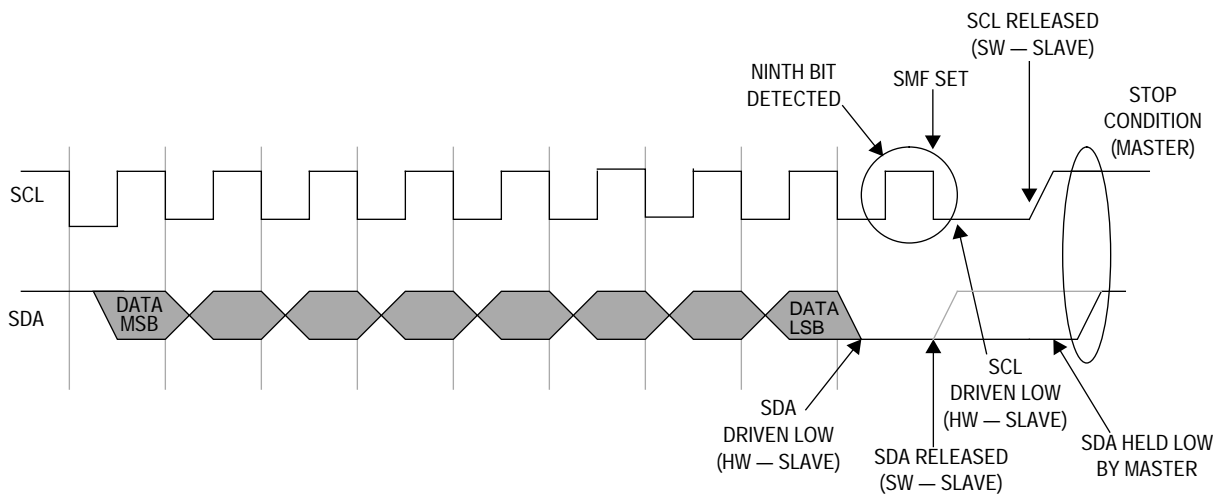
Figure 10-7. Additional Receptions Timing
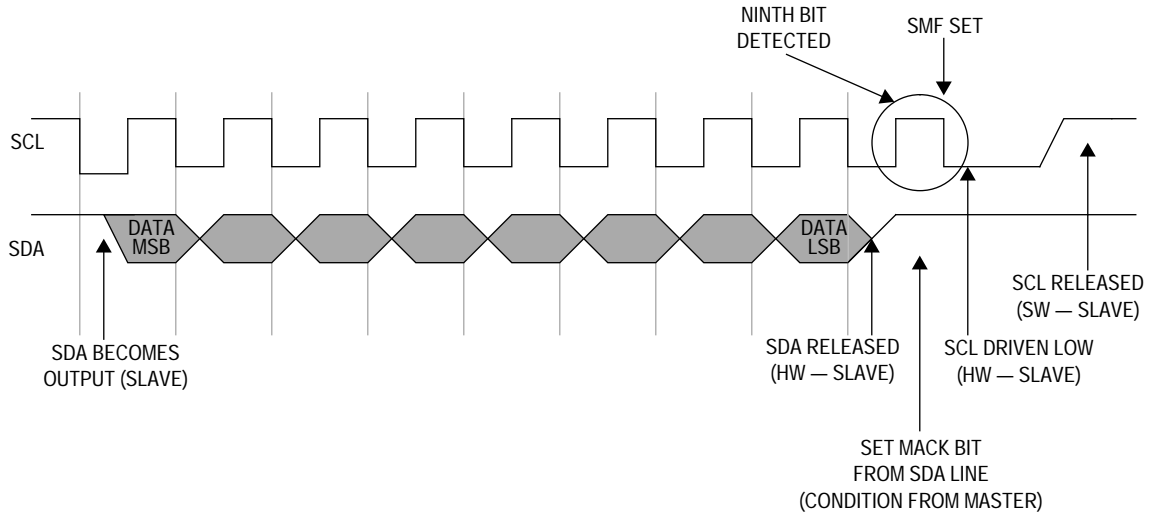
Figure 10-8. Transmissions (Master Read) Timing

SECTION 11
TIMER 1

11.1 Introduction

This timer can be used for many purposes, including input waveform measurements while simultaneously generating an output compare interrupt. Pulse widths can vary from several microseconds to many seconds. Refer to Figure 11-1.

Because the timer has a 16-bit architecture, each specific functional segment (capability) is represented by two registers. These registers contain the high and low byte of that functional segment. Generally, accessing the low byte of a specific timer function allows full control of that function; however, an access of the high byte inhibits that specific timer function until the low byte is also accessed.

NOTE

The I bit in the CCR should be set while manipulating both the high and low byte register of a specific timer function to ensure that an interrupt does not occur.
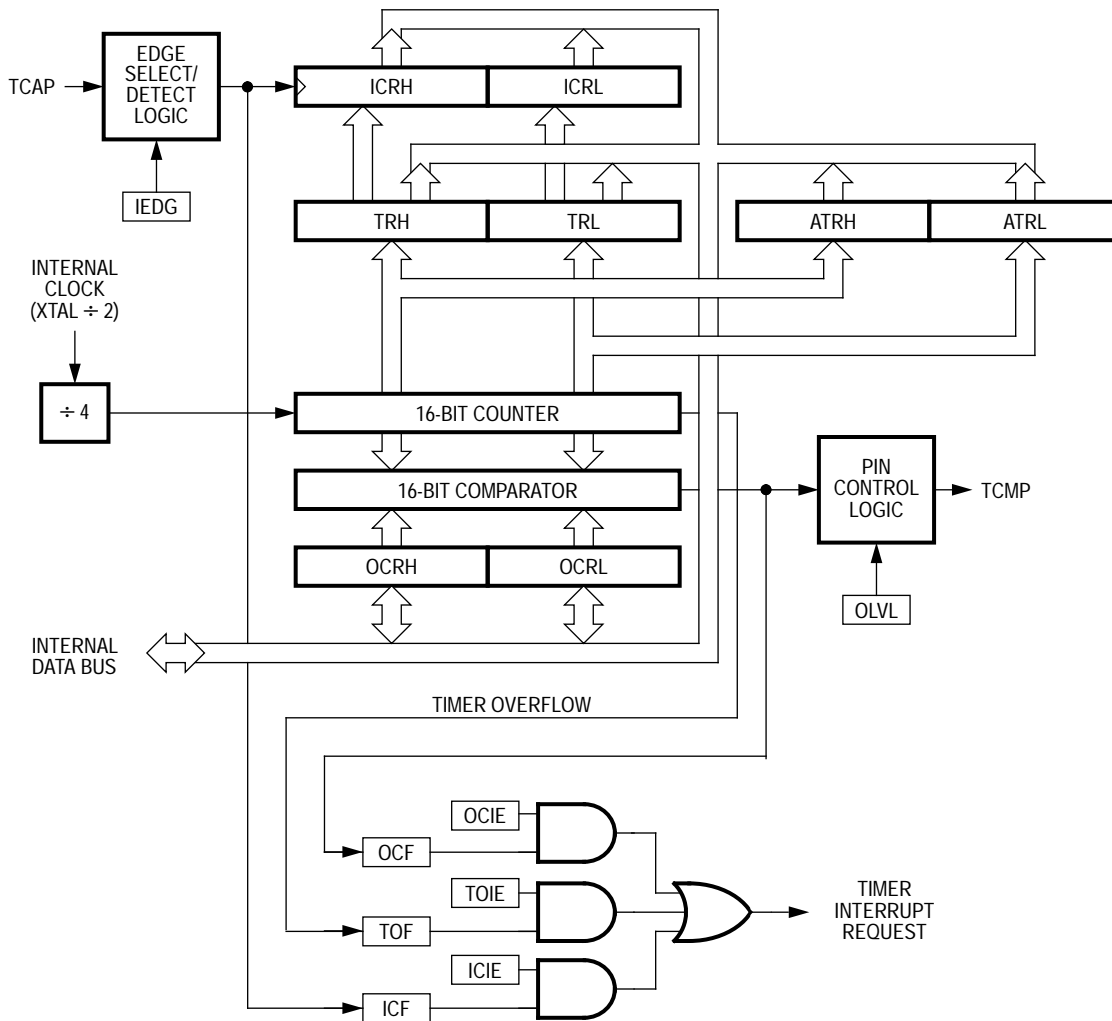
Figure 11-1. Timer 1 Block Diagram

11.2 Counter

The key element in the programmable timer is a 16-bit, free-running counter or counter register, preceded by a prescaler that divides the internal processor clock by four. The prescaler gives the timer a resolution of 2.0 microseconds if the internal bus clock is 2.0 MHz. The counter is incremented during the low portion of the internal bus clock. Software can read the counter at anytime without affecting its value.

The double-byte, free-running counter can be read from either of two locations, $18:$19 (counter register) or $1A:$1B (counter alternate register). A read from only the least significant byte (LSB) of the free-running counter ($19, $1B) receives the count value at the time of the read. If a read of the free-running counter or counter alternate register first addresses the most significant byte (MSB) ($18, $1A), the LSB ($19, $1B) is transferred to a buffer. This buffer value remains fixed after the

first MSB read, even if the user reads the MSB several times. This buffer is accessed when reading the free-running counter or counter alternate register LSB ($19 or $1B) and, thus, completes a read sequence of the total counter value. In reading either the free-running counter or counter alternate register, if the MSB is read, the LSB must also be read to complete the sequence.

The counter alternate register differs from the counter register in one respect: A read of the counter register MSB can clear the timer overflow flag (TOF). Therefore, the counter alternate register can be read at anytime without the possibility of missing timer overflow interrupts due to clearing of the TOF.

The free-running counter is configured to $FFFC during reset and is always a read-only register. During a power-on reset, the counter is also preset to $FFFC and begins running after the oscillator start-up delay. Because the free-running counter is 16 bits preceded by a fixed divided-by-four prescaler, the value in the free-running counter repeats every 262,144 internal bus clock cycles. When the counter rolls over from $FFFF to $0000, the TOF bit is set. An interrupt can also be enabled when counter rollover occurs by setting its interrupt enable bit (TOIE).

11.3 Output Compare Register

The 16-bit output compare register is made up of two 8-bit registers at locations $16 (MSB) and $17 (LSB). The output compare register is used for several purposes, such as indicating when a period of time has elapsed. All bits are readable and writable and are not altered by the timer hardware or reset. If the compare function is not needed, the two bytes of the output compare register can be used as storage locations.

The output compare register contents are compared with the contents of the free-running counter continually, and if a match is found, the corresponding output compare flag (OCF) bit is set and the corresponding output level (OLVL) bit is clocked to an output level register. The output compare register values and the output level bit should be changed after each successful comparison to establish a new elapsed timeout. An interrupt can also accompany a successful output compare provided the corresponding interrupt enable bit (OCIE) is set.

After a processor write cycle to the output compare register containing the MSB ($16), the output compare function is inhibited until the LSB ($17) is also written. The user must write both bytes (locations) if the MSB is written first. A write made only to the LSB ($17) will not inhibit the compare function. The free-running counter is updated every four internal bus clock cycles. The minimum time required to update the output compare register is a function of the program rather than the internal hardware.

The processor can write to either byte of the output compare register without affecting the other byte. The output level (OLVL) bit is clocked to the output level register regardless of whether the output compare flag (OCF) is set or clear.
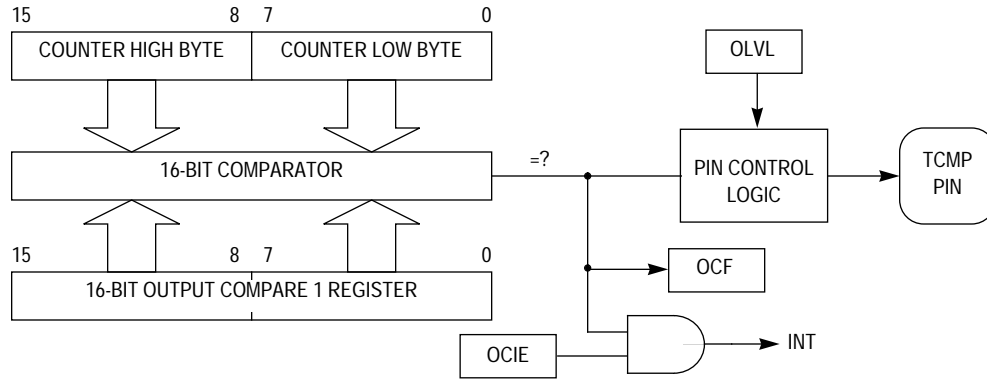
Figure 11-2. Timer 1 Output Compare Operation

Because neither the output compare flag (OCF bit) or output compare register is affected by reset, care must be exercised initializing the output compare function with software. The following procedure is recommended:

Write to the high byte of the output compare register to inhibit further compares until the low byte is written.

Read the timer status register to clear the OCF bit if it is already set.

Write to the low byte of the output compare register to enable the output compare function.

The purpose of this procedure is to prevent the OCF bit from being set between the writes to the high and low halves of the 16-bit output compare register. A software example follows:

```
B7      16      STA     OCRH        inhibit output compare
B6      13      LDA     T1SR        clear OCF bit if set
BF      17      STX     OCRL        ready for next compare
```

11.4 Input Capture Register

Two 8-bit registers, which make up the 16-bit input capture register, are read-only and are used to latch the value of the free-running counter after the corresponding input capture edge detector senses a defined transition. The level transition which triggers the counter transfer is defined by the corresponding input edge bit (IEDG). Reset does not affect the contents of the input capture register.

The result obtained by an input capture will be one more than the value of the free-running counter on the rising edge of the internal bus clock preceding the external transition. This delay is required for internal synchronization. Resolution is one count of the free-running counter, which is four internal bus clock cycles.

The free-running counter contents are transferred to the input capture register on each proper signal transition regardless of whether the input capture flag (ICF) is

set or clear. The input capture register always contains the free-running counter value that corresponds to the most recent input capture.

After a read of the input capture register ($14) MSB, the counter transfer is inhibited until the LSB ($15) is also read. This characteristic causes the time used in the input capture software routine and its interaction with the main program to determine the minimum pulse period.

A read of the input capture register LSB ($15) does not inhibit the free-running counter transfer since they occur on opposite edges of the internal bus clock.
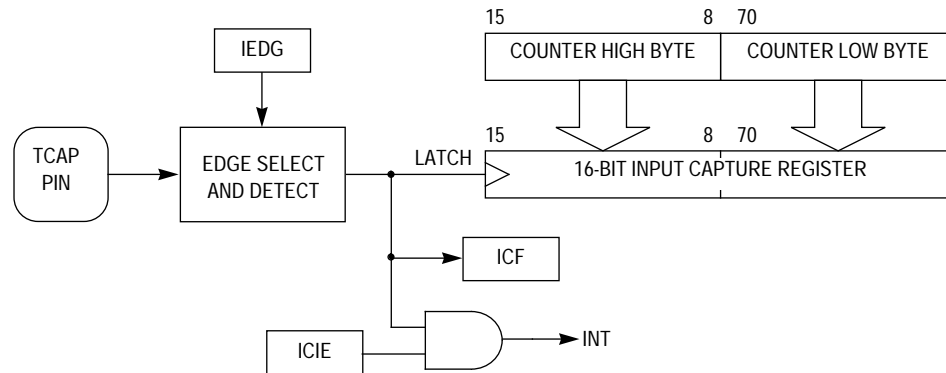
Figure 11-3. Timer 1 Input Capture Operation

11.5 Timer 1 Control Register (T1CR)

The TCR is a read/write register containing five control bits. Three bits enable interrupts associated with the timer status register flags ICF, OCF, and TOF.

		Bit 7	6	5	4	3	2	1	Bit 0
$0012 T1CR	Read:	ICIE	OCIE	TOIE	0	0	0	IEDG	OLVL
	Write:								
	Reset:	0	0	0	0	0	0	U	U

= Unimplemented   U = Unaffected
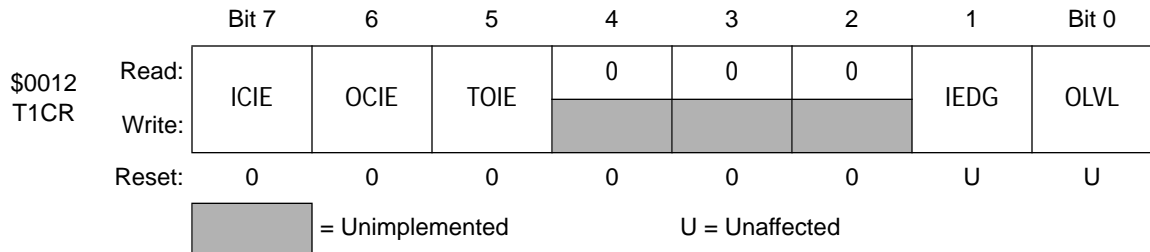
Figure 11-4. Timer 1 Control Register

ICIE — Input Capture Interrupt Enable
    1 = Interrupt enabled
    0 = Interrupt disabled

OCE — Output Compare Interrupt Enable
    1 = Interrupt enabled
    0 = Interrupt disabled

TOIE — Timer Overflow Interrupt Enable
    1 = Interrupt enabled
    0 = Interrupt disabled

BITS 2:4 — Not used

    Always read zero

IEDG — Input Edge

    Value of input edge determines which level transition on TCAP pin will trigger free-running counter transfer to the input capture register Reset does not affect the IEDG bit.
    1 = Positive edge
    0 = Negative edge

OLVL — Output Level

    Value of output level is clocked into output level register by the next successful output compare and will appear on the TCMP pin
    1 = High output
    0 = Low output

11.6 Timer 1 Status Register (T1SR)

The T1SR is a read-only register containing three status flag bits.

		Bit 7	6	5	4	3	2	1	Bit 0
$0013	Read:	ICF	OCF	TOF	0	0	0	0	0
T1SR	Write:								
	Reset:	X	X	X	0	0	0	0	0

☐ = Unimplemented   X = Undefined

Figure 11-5. Timer 1 Status Register

ICF — Input Capture Flag
   1 = Flag set when selected polarity edge is sensed by input capture edge detector
   0 = Flag cleared when T1SR and input capture low register ($15) are accessed

OCF — Output Compare Flag
   1 = Flag set when output compare register contents match the free-running counter contents
   0 = Flag cleared when T1SR and output compare low register ($17) are accessed

TOF — Timer Overflow Flag
   1 = Flag set when free-running counter transition from $FFFF to $0000 occurs
   0 = Flag cleared when T1SR and counter low register ($19) are accessed

Bits 0:4 — Not used and always read as zero

NOTE

Status of ICF, OCF, and TOF flag bits are undefined after reset.

Accessing the timer 1 status register satisfies the first condition required to clear status bits. The remaining step is to access the register corresponding to the status bit.

A problem can occur when using the timer overflow function and reading the free-running counter at random times to measure an elapsed time. Without incorporating the proper precautions into software, the timer overflow flag could unintentionally be cleared if:

1. The timer status register is read or written when TOF is set, and

2. The LSB of the free-running counter is read but not for the purpose of servicing the flag.

The counter alternate register at address $1A and $1B contains the same value as the free-running counter (at address $18 and $19); therefore, this alternate register can be read at anytime without affecting the timer overflow flag in the timer status register.

11.7 Timer 1 During Wait Mode

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

11.8 Timer 1 During Stop Mode

In the stop mode, the timer stops counting and holds the last count value if STOP is exited by an interrupt. If reset is used, the counter is forced to $FFFC. During stop, if at least one valid input capture edge occurs at the TCAP pin, the input capture detect circuit is armed. This does not set any timer flags nor wake up the MCU, but when the MCU does wake up, there is an active input capture flag and data from the first valid edge that occurred during the stop mode. If reset is used to exit stop mode, then no input capture flag or data remains, even if a valid input capture edge occurred.

SECTION 12
TIMER 2

12.1 Introduction

The timer for this device is a 15-stage multi-functional ripple counter. Features include:

- Timer Over Flow (TOF)

- Real Time Interrupt (RTI)

- Computer Operating Properly (COP) Watchdog.

As seen in Figure 12-1, the timer begins with a fixed divide by four prescaler. This signal 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 timer 2 counter register (T2CR) at address $09. A timer overflow function is implemented on the last stage of this counter, giving a possible interrupt at the rate of E/1024. Two additional stages produce the POR function at E/4064. The timer counter bypass circuitry is at this point in the timer chain. This circuit is followed by two more stages, with the resulting clock (E/16384) driving the real time interrupt circuit. The RTI circuit consists of three divider stages with a 1-of-4 selector. The output of the RTI circuit is further divided by eight to drive the software configurable COP watchdog timer circuit. The RTI rate selector bits, and the RTI and TOF enable bits and flags are located in the timer control and status register at location $08.
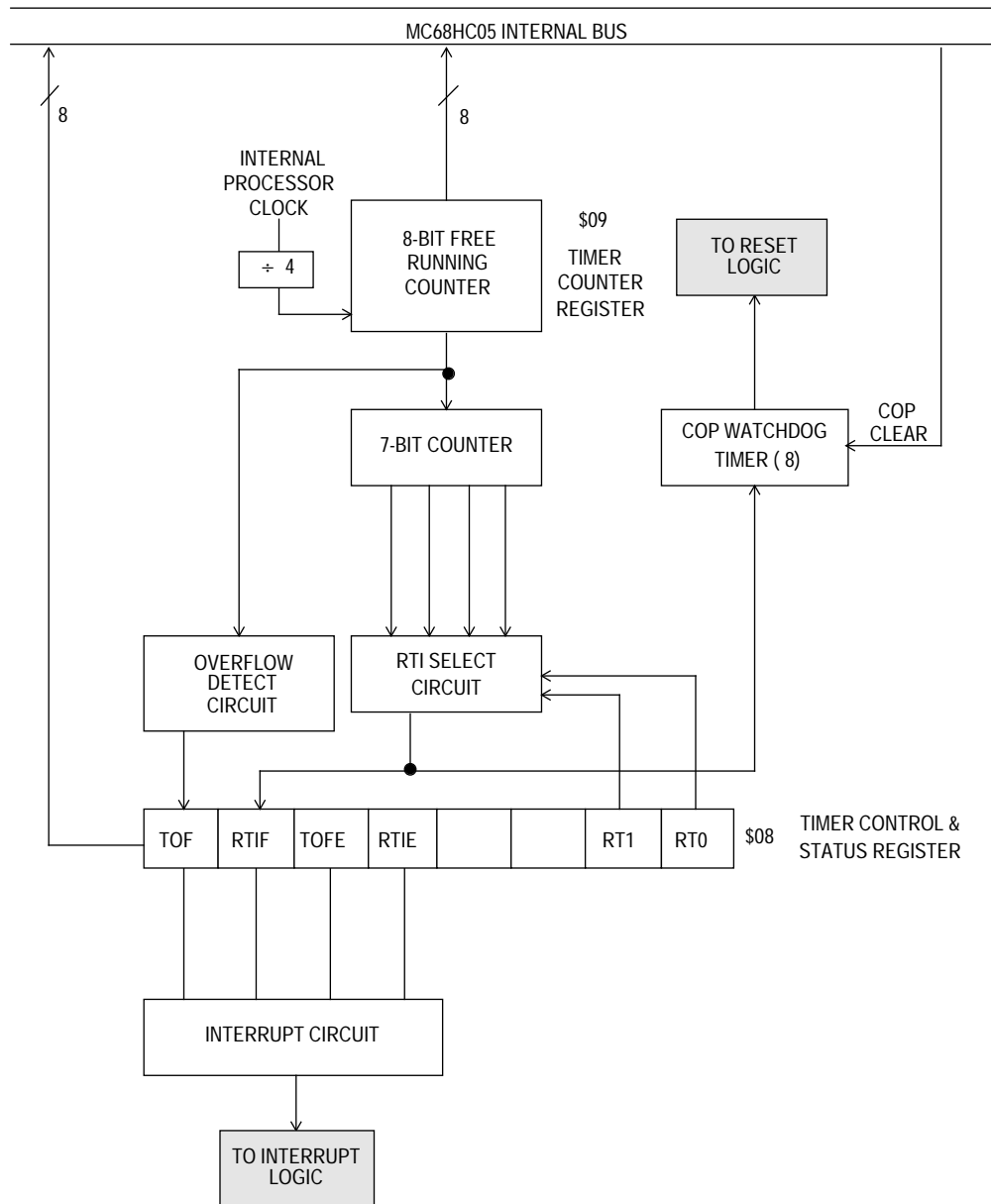
Figure 12-1. Timer 2 Block Diagram

12.2 Flag Clearing Considerations

In rare instances, clearing any of the timer 2 control and status register (T2CSR) flag or enable bits could result in vectoring to the reset vector rather than the timer interrupt vector if the correct precautions are not followed. Do not clear any of the timer flags or enable bits (for example, TOF, TOFE, RTI, and RTIF) with bit manipulation instructions.

12.2.1 Clearing Timer Overflow Flag (TOF)

Use the following program to clear timer overflow flag (TOF) bit.

```
SEI             SEI NOT REQUIRED IF USED WITHIN TIMER INTERRUPT ROUTINE
LDA     #$73
AND     $T2CSR
OR      #$40    MASK RTIF BIT
STA     $T2CSR
CLI             DO NOT USE CLI IF THIS CODE SEGMENT IS USED WITHIN TIMER
                INTERRUPT ROUTINE
```

12.2.2 Clearing Timer Overflow Flag Enable (TOFE)

Use the following program to clear timer overflow flag enable (TOFE) bit.

```
SEI             SEI NOT REQUIRED IF USED WITHIN TIMER INTERRUPT ROUTINE
LDA     #$D3
AND     $T2CSR
OR      #$C0    MASK RTIF & TOF
STA     $T2CSR
CLI             DO NOT USE CLI IF THIS CODE SEGMENT IS USED WITHIN TIMER
                INTERRUPT ROUTINE.
```

Masking the real-time interrupt flag (RTIF) and timer overflow flag (TOF) bits with the OR instruction prevents these bits from being cleared if a TOF or real time interrupt (RTI) is generated during the clearing routine. Similar sequences should be used for clearing of the RTIF and real-time interrupt enable (RTIE) bits.

12.3 Timer 2 Control and Status Register (T2CSR)

The T2CSR contains the timer interrupt flag, the timer interrupt enable bits, and the real time interrupt rate select bits.

		Bit 7	6	5	4	3	2	1	Bit 0
$0013 T1SR	Read:	TOF	RTIF	TOFE	RTIE	COPE	IRQS	RT1	RT0
	Write:	TOFR	RTIFR						
	Reset:	0	0	0	0	0	0	1	1

Figure 12-2. Timer 2 Control/Status Register

TOF — Timer Over Flow

TOF is a clearable, read-only status bit and is set when the 8-bit ripple counter rolls over from $FF to $00. A CPU interrupt request will be generated if TOFE is set. Clearing the TOF is done by writing a zero to it (TOFR). Writing a one to TOFR has no effect on the bit's value. Reset clears TOF.

RTIF — Real Time Interrupt Flag

The real time interrupt circuit consists of a three stage divider and a 1 of 4 selector. The clock frequency that drives the RTI circuit is E/2**14 (or E/16384) with three additional divider stages giving a maximum interrupt period of 65.5 milliseconds at a bus rate of 2 MHz. RTIF is a clearable, read-only status bit and is set when the output of the chosen (1 of 4 selection) stage goes active. A CPU interrupt request will be generated if RTIE is set. Clearing the RTIF is done by writing a zero to it (RTIFR). Writing a one to RTIFR has no effect on this bit. Reset clears RTIF.

NOTE

Care should be taken when clearing TOF and RTIF. Writing a zero to a bit that is already clear can cause a pending interrupt to be missed. For this reason, the use of a read-modify-write instruction is not recommended. To insure proper operation, only write a logic zero to a flag that is already set and a logic one to any flag that is clear.

TOFE — Timer Over Flow Enable

When this bit is set, a CPU interrupt request is generated when the TOF 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.

COPE — COP Enable

When set, COPE turns on the COP. It can only be written to once after reset. Reset clears COPE and either a 0 or a 1 can be written to it in order to lock in the desired function. Please refer to **12.4 COP Watchdog Reset** for more information on the COP.
   1 = COP Enabled
   0 = COP disabled

IRQS — $\overline{\text{IRQ}}$ Select

This bit selects the type of input recognized by the $\overline{\text{IRQ}}$ interrupt input. It can only be written to once after reset. Reset clears IRQS and either a 0 or a 1 can be written to it in order to lock in the desired function.
   1 = Negative Edge sensitive only.
   0 = Negative Edge and Level Sensitive.Always read zero

RT1:RT0 — Real Time Interrupt Rate Select

These two bits select one of four taps from the Real Time Interrupt circuit. Table 12-1 shows the available interrupt rates with a 2 MHz bus clock. 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 time-out 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 12-1. RTI and COP Rates at 2 MHz Bus Frequency

RT1:RT0	Ratio	RTI Rate	Minimum COP Reset
00	$E/2^{14}$	8.2 ms	57.4 ms
01	$E/2^{15}$	16.4 ms	114.8 ms
10	$E/2^{16}$	32.8 ms	229.6 ms
11	$E/2^{17}$	65.5 ms	458.5 ms

12.4 COP Watchdog 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 12-1. If the COP circuit times out, an internal reset is generated and the normal reset vector is fetched. Preventing a COP time-out is done by writing a zero to bit 0 of address $1FF0. When the COP is cleared, only the final divide by eight stage (output of the RTI) is cleared.

12.5 Timer 2 Counter Register (T2CR)

The timer 2 counter register is a read-only register which contains the current value of the 8-bit ripple counter at the beginning of the timer chain. This counter is clocked at E divided by 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.

	Bit 7	6	5	4	3	2	1	Bit 0
$0009 T2CR — Read: / Write:	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0

Unaffected by Reset
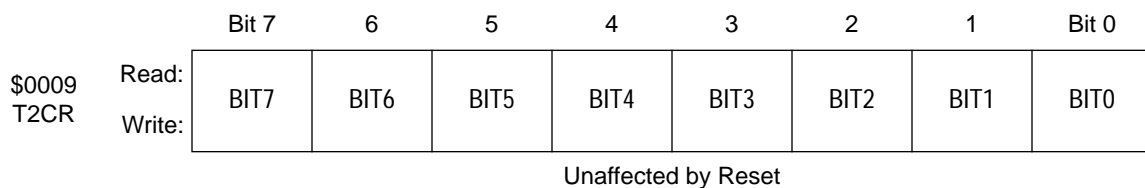
Figure 12-3. Timer 2 Counter Register

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 $\overline{RESET}$ is not asserted, the timer will start counting up from zero and normal device operation will begin. When $\overline{RESET}$ is asserted anytime during operation (other than POR), the counter chain will be cleared.

12.6 Timer 2 During Wait Mode

The CPU clock halts during the wait mode, but the timer remains active. If interrupts are enabled, a timer interrupt will cause the processor to exit the WAIT mode.

12.7 Timer 2 During Stop Mode

The timer is cleared when going into stop mode. When stop is exited by an external interrupt or an external $\overline{RESET}$, the internal oscillator will resume, followed by a 4064 internal processor oscillator stabilization delay. The timer is then cleared and operation resumes.

SECTION 13
INSTRUCTION SET

13.1 Introduction

This section describes the addressing modes and instruction types.

13.2 Addressing Modes

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

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

13.2.1 Inherent

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

13.2.2 Immediate

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

13.2.3 Direct

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

13.2.4 Extended

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

When using the Motorola assembler, the programmer does not need to specify whether an instruction is direct or extended. The assembler automatically selects the shortest form of the instruction.

13.2.5 Indexed, No Offset

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

Indexed, no offset instructions are often used to move a pointer through a table or to hold the address of a frequently used RAM or I/O location.

13.2.6 Indexed, 8-Bit Offset

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

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

13.2.7 Indexed, 16-Bit Offset

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

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

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

13.2.8 Relative

Relative addressing is only for branch instructions. If the branch condition is true, the CPU finds the conditional branch destination by adding the signed byte following the opcode to the contents of the program counter. If the branch condition is not true, the CPU goes to the next instruction. The offset is a signed, two's complement byte that gives a branching range of –128 to +127 bytes from the address of the next location after the branch instruction.

When using the Motorola assembler, the programmer does not need to calculate the offset, because the assembler determines the proper offset and verifies that it is within the span of the branch.

13.3 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

13.3.1 Register/Memory Instructions

Most of these instructions use two operands. One operand is in either the accumulator or the index register. The CPU finds the other operand in memory. Table 13-1 lists the register/memory instructions.

Table 13-1. 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

13.3.2 Read-Modify-Write Instructions

These instructions read a memory location or a register, modify its contents, and write the modified value back to the memory location or to the register. The test for negative or zero instruction (TST) is an exception to the read-modify-write sequence because it does not write a replacement value. Table 13-2 lists the read-modify-write instructions.

Table 13-2. Read-Modify-Write Instructions

Instruction	Mnemonic
Arithmetic Shift Left	ASL
Arithmetic Shift Right	ASR
Clear Bit in Memory	BCLR
Set Bit in Memory	BSET
Clear	CLR
Complement (One's Complement)	COM
Decrement	DEC
Increment	INC
Logical Shift Left	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

13.3.3 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. All branch instructions use relative addressing.

Bit test and branch instructions cause a branch based on the state of any readable bit in the first 256 memory locations. These three-byte instructions use a combination of direct addressing and relative addressing. The direct address of the byte to be tested is in the byte following the opcode. The third byte is the signed offset byte. The CPU finds the conditional branch destination by adding the third byte to the program counter if the specified bit tests true. The bit to be tested and

its condition (set or clear) is part of the opcode. The span of branching is from –128 to +127 from the address of the next location after the branch instruction. The CPU also transfers the tested bit to the carry/borrow bit of the condition code register. Table 13-3 lists the jump and branch instructions.

Table 13-3. Jump and Branch Instructions

Instruction	Mnemonic
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

13.3.4 Bit Manipulation Instructions

The CPU can set or clear any writable bit in the first 256 bytes of memory. Port registers, port data direction registers, timer registers, and on-chip RAM locations are in the first 256 bytes of memory. The CPU can also test and branch based on the state of any bit in any of the first 256 memory locations. Bit manipulation instructions use direct addressing. Table 13-4 lists these instructions.

Table 13-4. Bit Manipulation Instructions

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

13.3.5 Control Instructions

These register reference instructions control CPU operation during program execution. Control instructions, listed in Table 13-5, use inherent addressing.

Table 13-5. 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

13.4 Instruction Set Summary

Table 13-6 is an alphabetical list of all M68HC05 instructions and shows the effect of each instruction on the condition code register.

Table 13-6. Instruction Set Summary
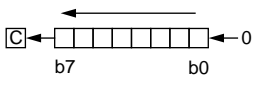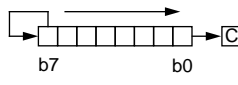
Source Form	Operation	Description	H	I	N	Z	C	Address Mode	Opcode	Operand	Cycles
ADC #*opr* ADC *opr* ADC *opr* ADC *opr*,X ADC *opr*,X ADC ,X	Add with Carry	A ← (A) + (M) + (C)	↕	—	↕	↕	↕	IMM DIR EXT IX2 IX1 IX	A9 B9 C9 D9 E9 F9	ii dd hh ll ee ff ff	2 3 4 5 4 3
ADD #*opr* ADD *opr* ADD *opr* ADD *opr*,X ADD *opr*,X ADD ,X	Add without Carry	A ← (A) + (M)	↕	—	↕	↕	↕	IMM DIR EXT IX2 IX1 IX	AB BB CB DB EB FB	ii dd hh ll ee ff ff	2 3 4 5 4 3
AND #*opr* AND *opr* AN*D* *opr* AND *opr*,X AND *opr*,X AND ,X	Logical AND	A ← (A) ∧ (M)	—	—	↕	↕	—	IMM DIR EXT IX2 IX1 IX	A4 B4 C4 D4 E4 F4	ii dd hh ll ee ff ff	2 3 4 5 4 3
ASL *opr* ASLA ASLX ASL *opr*,X ASL ,X	Arithmetic Shift Left (Same as LSL)		—	—	↕	↕	↕	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		—	—	↕	↕	↕	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 ← (PC) + 2 + *rel* ? C = 0	—	—	—	—	—	REL	24	rr	3
BCLR *n opr*	Clear Bit n	Mn ← 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 ← (PC) + 2 + *rel* ? C = 1	—	—	—	—	—	REL	25	rr	3
BEQ *rel*	Branch if Equal	PC ← (PC) + 2 + *rel* ? Z = 1	—	—	—	—	—	REL	27	rr	3

Table 13-6. Instruction Set Summary (Continued)

Source Form	Operation	Description	H	I	N	Z	C	Address Mode	Opcode	Operand	Cycles
BHCC *rel*	Branch if Half-Carry Bit Clear	PC ← (PC) + 2 + *rel* ? H = 0	—	—	—	—	—	REL	28	rr	3
BHCS *rel*	Branch if Half-Carry Bit Set	PC ← (PC) + 2 + *rel* ? H = 1	—	—	—	—	—	REL	29	rr	3
BHI *rel*	Branch if Higher	PC ← (PC) + 2 + *rel* ? C ∨ Z = 0	—	—	—	—	—	REL	22	rr	3
BHS *rel*	Branch if Higher or Same	PC ← (PC) + 2 + *rel* ? C = 0	—	—	—	—	—	REL	24	rr	3
BIH *rel*	Branch if $\overline{IRQ}$ Pin High	PC ← (PC) + 2 + *rel* ? $\overline{IRQ}$ = 1	—	—	—	—	—	REL	2F	rr	3
BIL *rel*	Branch if $\overline{IRQ}$ Pin Low	PC ← (PC) + 2 + *rel* ? $\overline{IRQ}$ = 0	—	—	—	—	—	REL	2E	rr	3
BIT #*opr* BIT *opr* BIT *opr* BIT *opr*,X BIT *opr*,X BIT ,X	Bit Test Accumulator with Memory Byte	(A) ∧ (M)	—	—	↕	↕	—	IMM DIR EXT IX2 IX1 IX	A5 B5 C5 D5 E5 F5	ii dd hh ll ee ff ff p	2 3 4 5 4 3
BLO *rel*	Branch if Lower (Same as BCS)	PC ← (PC) + 2 + *rel* ? C = 1	—	—	—	—	—	REL	25	rr	3
BLS *rel*	Branch if Lower or Same	PC ← (PC) + 2 + *rel* ? C ∨ Z = 1	—	—	—	—	—	REL	23	rr	3
BMC *rel*	Branch if Interrupt Mask Clear	PC ← (PC) + 2 + *rel* ? I = 0	—	—	—	—	—	REL	2C	rr	3
BMI *rel*	Branch if Minus	PC ← (PC) + 2 + *rel* ? N = 1	—	—	—	—	—	REL	2B	rr	3
BMS *rel*	Branch if Interrupt Mask Set	PC ← (PC) + 2 + *rel* ? I = 1	—	—	—	—	—	REL	2D	rr	3
BNE *rel*	Branch if Not Equal	PC ← (PC) + 2 + *rel* ? Z = 0	—	—	—	—	—	REL	26	rr	3
BPL *rel*	Branch if Plus	PC ← (PC) + 2 + *rel* ? N = 0	—	—	—	—	—	REL	2A	rr	3
BRA *rel*	Branch Always	PC ← (PC) + 2 + *rel* ? 1 = 1	—	—	—	—	—	REL	20	rr	3
BRCLR *n opr rel*	Branch if bit n clear	PC ← (PC) + 2 + *rel* ? Mn = 0	—	—	—	—	↕	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	01 03 05 07 09 0B 0D 0F	dd rr dd rr dd rr dd rr dd rr dd rr dd rr dd rr	5 5 5 5 5 5 5 5
BRSET *n opr rel*	Branch if Bit n Set	PC ← (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
BRN *rel*	Branch Never	PC ← (PC) + 2 + *rel* ? 1 = 0	—	—	—	—	—	REL	21	rr	3

Table 13-6. Instruction Set Summary (Continued)

Source Form	Operation	Description	Effect on CCR					Address Mode	Opcode	Operand	Cycles
			H	I	N	Z	C				
BSET *n opr*	Set Bit n	Mn ← 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 *rel*	Branch to Subroutine	PC ← (PC) + 2; push (PCL) SP ← (SP) − 1; push (PCH) SP ← (SP) − 1 PC ← (PC) + *rel*	—	—	—	—	—	REL	AD	rr	6
CLC	Clear Carry Bit	C ← 0	—	—	—	—	0	INH	98		2
CLI	Clear Interrupt Mask	I ← 0	—	0	—	—	—	INH	9A		2
CLR *opr* CLRA CLRX CLR *opr*,X CLR ,X	Clear Byte	M ← $00 A ← $00 X ← $00 M ← $00 M ← $00	—	—	0	1	—	DIR INH INH IX1 IX	3F 4F 5F 6F 7F	dd ff	5 3 3 6 5
CMP #*opr* CMP *opr* CMP *opr* CMP *opr*,X CMP *opr*,X CMP ,X	Compare Accumulator with Memory Byte	(A) − (M)	—	—	↕	↕	↕	IMM DIR EXT IX2 IX1 IX	A1 B1 C1 D1 E1 F1	ii dd hh ll ee ff ff	2 3 4 5 4 3
COM *opr* COMA COMX COM *opr*,X COM ,X	Complement Byte (One's Complement)	M ← ($\overline{M}$) = $FF − (M) A ← ($\overline{A}$) = $FF − (M) X ← ($\overline{X}$) = $FF − (M) M ← ($\overline{M}$) = $FF − (M) M ← ($\overline{M}$) = $FF − (M)	—	—	↕	↕	1	DIR INH INH IX1 IX	33 43 53 63 73	dd ff	5 3 3 6 5
CPX #*opr* CPX *opr* CPX *opr* CPX *opr*,X CPX *opr*,X CPX ,X	Compare Index Register with Memory Byte	(X) − (M)	—	—	↕	↕	1	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 *opr* DECA DECX DEC *opr*,X DEC ,X	Decrement Byte	M ← (M) − 1 A ← (A) − 1 X ← (X) − 1 M ← (M) − 1 M ← (M) − 1	—	—	↕	↕	—	DIR INH INH IX1 IX	3A 4A 5A 6A 7A	dd ff	5 3 3 6 5
EOR #*opr* EOR *opr* EOR *opr* EOR *opr*,X EOR *opr*,X EOR ,X	EXCLUSIVE OR Accumulator with Memory Byte	A ← (A) ⊕ (M)	—	—	↕	↕	—	IMM DIR EXT IX2 IX1 IX	A8 B8 C8 D8 E8 F8	ii dd hh ll ee ff ff	2 3 4 5 4 3

INSTRUCTION SET
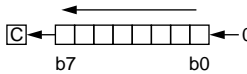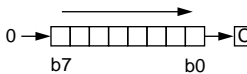
Table 13-6. Instruction Set Summary (Continued)

Source Form	Operation	Description	H	I	N	Z	C	Address Mode	Opcode	Operand	Cycles
INC *opr* INCA INCX INC *opr*,X INC ,X	Increment Byte	M ← (M) + 1 A ← (A) + 1 X ← (X) + 1 M ← (M) + 1 M ← (M) + 1	—	—	↕	↕	—	DIR INH INH IX1 IX	3C 4C 5C 6C 7C	dd ff	5 3 3 6 5
JMP *opr* JMP *opr* JMP *opr*,X JMP *opr*,X JMP ,X	Unconditional Jump	PC ← Jump Address	—	—	—	—	—	DIR EXT IX2 IX1 IX	BC CC DC CC EC FC	dd hh ll ee ff ff	2 3 4 3 2
JSR *opr* JSR *opr* JSR *opr*,X JSR *opr*,X JSR ,X	Jump to Subroutine	PC ← (PC) + n (n = 1, 2, or 3) Push (PCL); SP ← (SP) − 1 Push (PCH); SP ← (SP) − 1 PC ← Conditional Address	—	—	—	—	—	DIR EXT IX2 IX1 IX	BD CD DD DD ED FD	dd hh ll ee ff ff	5 6 7 6 5
LDA #*opr* LDA *opr* LDA *opr* LDA *opr*,X LDA *opr*,X LDA ,X	Load Accumulator with Memory Byte	A ← (M)	—	—	↕	↕	—	IMM DIR EXT IX2 IX1 IX	A6 B6 C6 D6 E6 F6	ii dd hh ll ee ff ff	2 3 4 5 4 3
LDX #*opr* LDX *opr* LDX *opr* LDX *opr*,X LDX *opr*,X LDX ,X	Load Index Register with Memory Byte	X ← (M)	—	—	↕	↕	—	IMM DIR EXT IX2 IX1 IX	AE BE CE DE EE FE	ii dd hh ll ee ff ff	2 3 4 5 4 3
LSL *opr* LSLA LSLX LSL *opr*,X LSL ,X	Logical Shift Left (Same as ASL)		—	—	↕	↕	↕	DIR INH INH IX1 IX	38 48 58 68 78	dd ff	5 3 3 6 5
LSR *opr* LSRA LSRX LSR *opr*,X 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

Table 13-6. Instruction Set Summary (Continued)
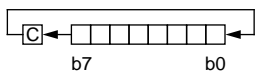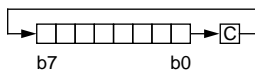
Source Form	Operation	Description	H	I	N	Z	C	Address Mode	Opcode	Operand	Cycles
NEG opr NEGA NEGX NEG opr,X NEG ,X	Negate Byte (Two's Complement)	M ← −(M) = $00 − (M) A ← −(A) = $00 − (A) X ← −(X) = $00 − (X) M ← −(M) = $00 − (M) M ← −(M) = $00 − (M)	—	—	↕	↕	↕	DIR INH INH IX1 IX	30 40 50 60 70	ii ff	5 3 3 6 5
NOP	No Operation		—	—	—	—	—	INH	9D		2
ORA #opr ORA opr ORA opr ORA opr,X ORA opr,X ORA ,X	Logical OR Accumulator with Memory	A ← (A) ∨ (M)	—	—	↕	↕	—	IMM DIR EXT IX2 IX1 IX	AA BA CA DA EA FA	ii dd hh ll ee ff ff	2 3 4 5 4 3
ROL opr ROLA ROLX ROL opr,X ROL ,X	Rotate Byte Left through Carry Bit		—	—	↕	↕	↕	DIR INH INH IX1 IX	39 49 59 69 79	dd ff	5 3 3 6 5
ROR opr RORA RORX ROR opr,X ROR ,X	Rotate Byte Right through Carry Bit		—	—	↕	↕	↕	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)	↕	↕	↕	↕	↕	INH	80		6
RTS	Return from Subroutine	SP ← (SP) + 1; Pull (PCH) SP ← (SP) + 1; Pull (PCL)						INH			
SBC #opr SBC opr SBC opr SBC opr,X SBC opr,X 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 opr STA opr STA opr,X STA opr,X STA ,X	Store Accumulator in Memory	M ← (A)	—	—	↕	↕	—	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

Table 13-6. Instruction Set Summary (Continued)

Source Form	Operation	Description	Effect on CCR					Address Mode	Opcode	Operand	Cycles
			H	I	N	Z	C				
STX *opr* STX *opr* STX *opr*,X STX *opr*,X STX ,X	Store Index Register In Memory	M ← (X)	—	—	↕	↕	—	DIR EXT IX2 IX1 IX	BF CF DF EF FF	dd hh ll ee ff ff	4 5 6 5 4
SUB #*opr* SUB *opr* SUB *opr* SUB *opr*,X SUB *opr*,X 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
TST *opr* TSTA TSTX TST *opr*,X TST ,X	Test Memory Byte for Negative or Zero	(M) – $00	—	—	—	—	—	DIR INH INH IX1 IX	3D 4D 5D 6D 7D	dd ff	4 3 3 5 4
TXA	Transfer Index Register to Accumulator	A ← (X)	—	—	—	—	—	INH	9F		2
WAIT	Stop CPU Clock and Enable Interrupts		—	↕	—	—	—	INH	8F		2

A	Accumulator	*opr*	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	*rel*	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
n	Any bit	—	Not affected

Table 13-7. Opcode Map

Legend for each cell: Mnemonic — Cycles / Addressing Mode / Bytes

LSB \ MSB	0 (Bit Manip DIR)	1 (Bit Manip DIR)	2 (Branch REL)	3 (RMW DIR)	4 (RMW INH)	5 (RMW INH)	6 (RMW IX1)	7 (RMW IX)	8 (Control INH)	9 (Control INH)	A (IMM)	B (DIR)	C (EXT)	D (IX2)	E (IX1)	F (IX)
0	BRSET0 5/DIR/3	BSET0 5/DIR/2	BRA 3/REL/2	NEG 5/DIR/2	NEGA 3/INH/1	NEGX 3/INH/1	NEG 6/IX1/2	NEG 5/IX/1	RTI 9/INH/1		SUB 2/IMM/2	SUB 3/DIR/2	SUB 4/EXT/3	SUB 5/IX2/3	SUB 4/IX1/2	SUB 3/IX/1
1	BRCLR0 5/DIR/3	BCLR0 5/DIR/2	BRN 3/REL/2						RTS 6/INH/1		CMP 2/IMM/2	CMP 3/DIR/2	CMP 4/EXT/3	CMP 5/IX2/3	CMP 4/IX1/2	CMP 3/IX/1
2	BRSET1 5/DIR/3	BSET1 5/DIR/2	BHI 3/REL/2		MUL 11/INH/1				SWI 10/INH/1		SBC 2/IMM/2	SBC 3/DIR/2	SBC 4/EXT/3	SBC 5/IX2/3	SBC 4/IX1/2	SBC 3/IX/1
3	BRCLR1 5/DIR/3	BCLR1 5/DIR/2	BLS 3/REL/2	COM 5/DIR/2	COMA 3/INH/1	COMX 3/INH/1	COM 6/IX1/2	COM 5/IX/1			CPX 2/IMM/2	CPX 3/DIR/2	CPX 4/EXT/3	CPX 5/IX2/3	CPX 4/IX1/2	CPX 3/IX/1
4	BRSET2 5/DIR/3	BSET2 5/DIR/2	BCC 3/REL/2	LSR 5/DIR/2	LSRA 3/INH/1	LSRX 3/INH/1	LSR 6/IX1/2	LSR 5/IX/1			AND 2/IMM/2	AND 3/DIR/2	AND 4/EXT/3	AND 5/IX2/3	AND 4/IX1/2	AND 3/IX/1
5	BRCLR2 5/DIR/3	BCLR2 5/DIR/2	BCS/BLO 3/REL/2								BIT 2/IMM/2	BIT 3/DIR/2	BIT 4/EXT/3	BIT 5/IX2/3	BIT 4/IX1/2	BIT 3/IX/1
6	BRSET3 5/DIR/3	BSET3 5/DIR/2	BNE 3/REL/2	ROR 5/DIR/2	RORA 3/INH/1	RORX 3/INH/1	ROR 6/IX1/2	ROR 5/IX/1			LDA 2/IMM/2	LDA 3/DIR/2	LDA 4/EXT/3	LDA 5/IX2/3	LDA 4/IX1/2	LDA 3/IX/1
7	BRCLR3 5/DIR/3	BCLR3 5/DIR/2	BEQ 3/REL/2	ASR 5/DIR/2	ASRA 3/INH/1	ASRX 3/INH/1	ASR 6/IX1/2	ASR 5/IX/1		TAX 2/INH/1		STA 4/DIR/2	STA 5/EXT/3	STA 6/IX2/3	STA 5/IX1/2	STA 4/IX/1
8	BRSET4 5/DIR/3	BSET4 5/DIR/2	BHCC 3/REL/2	ASL/LSL 5/DIR/2	ASLA/LSLA 3/INH/1	ASLX/LSLX 3/INH/1	ASL/LSL 6/IX1/2	ASL/LSL 5/IX/1		CLC 2/INH/1	EOR 2/IMM/2	EOR 3/DIR/2	EOR 4/EXT/3	EOR 5/IX2/3	EOR 4/IX1/2	EOR 3/IX/1
9	BRCLR4 5/DIR/3	BCLR4 5/DIR/2	BHCS 3/REL/2	ROL 5/DIR/2	ROLA 3/INH/1	ROLX 3/INH/1	ROL 6/IX1/2	ROL 5/IX/1		SEC 2/INH/1	ADC 2/IMM/2	ADC 3/DIR/2	ADC 4/EXT/3	ADC 5/IX2/3	ADC 4/IX1/2	ADC 3/IX/1
A	BRSET5 5/DIR/3	BSET5 5/DIR/2	BPL 3/REL/2	DEC 5/DIR/2	DECA 3/INH/1	DECX 3/INH/1	DEC 6/IX1/2	DEC 5/IX/1		CLI 2/INH/1	ORA 2/IMM/2	ORA 3/DIR/2	ORA 4/EXT/3	ORA 5/IX2/3	ORA 4/IX1/2	ORA 3/IX/1
B	BRCLR5 5/DIR/3	BCLR5 5/DIR/2	BMI 3/REL/2							SEI 2/INH/1	ADD 2/IMM/2	ADD 3/DIR/2	ADD 4/EXT/3	ADD 5/IX2/3	ADD 4/IX1/2	ADD 3/IX/1
C	BRSET6 5/DIR/3	BSET6 5/DIR/2	BMC 3/REL/2	INC 5/DIR/2	INCA 3/INH/1	INCX 3/INH/1	INC 6/IX1/2	INC 5/IX/1		RSP 2/INH/1		JMP 2/DIR/2	JMP 3/EXT/3	JMP 4/IX2/3	JMP 3/IX1/2	JMP 2/IX/1
D	BRCLR6 5/DIR/3	BCLR6 5/DIR/2	BMS 3/REL/2	TST 4/DIR/2	TSTA 3/INH/1	TSTX 3/INH/1	TST 5/IX1/2	TST 4/IX/1		NOP 2/INH/1	BSR 6/REL/2	JSR 5/DIR/2	JSR 6/EXT/3	JSR 7/IX2/3	JSR 6/IX1/2	JSR 5/IX/1
E	BRSET7 5/DIR/3	BSET7 5/DIR/2	BIL 3/REL/2						STOP 2/INH/1		LDX 2/IMM/2	LDX 3/DIR/2	LDX 4/EXT/3	LDX 5/IX2/3	LDX 4/IX1/2	LDX 3/IX/1
F	BRCLR7 5/DIR/3	BCLR7 5/DIR/2	BIH 3/REL/2	CLR 5/DIR/2	CLRA 3/INH/1	CLRX 3/INH/1	CLR 6/IX1/2	CLR 5/IX/1	WAIT 2/INH/1	TXA 2/INH/1		STX 4/DIR/2	STX 5/EXT/3	STX 6/IX2/3	STX 5/IX1/2	STX 4/IX/1

Abbreviations:

INH = Inherent
IMM = Immediate
DIR = Direct
EXT = Extended

REL = Relative
IX = Indexed, No Offset
IX1 = Indexed, 8-Bit Offset
IX2 = Indexed, 16-Bit Offset

Key:

	MSB	
LSB	0	MSB of Opcode in Hexadecimal

| | 0 | BRSET0 5 |
| | | DIR 3 |

LSB of Opcode in Hexadecimal

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

SECTION 14
ELECTRICAL SPECIFICATIONS

14.1 Introduction

This section contains parametric and timing information.

14.2 Maximum Ratings

Rating	Symbol	Value	Unit
Supply Voltage	V_{DD}	−0.3 to +7.0	V
Input Voltage	V_{IN}	V_{SS} −0.3 to V_{DD} +0.3	V
Bootloader Mode ($\overline{IRQ}$ Pin Only)	V_{TST}	V_{SS} −0.3 to 2 x V_{DD} +0.3	V
Current Drain per Pin Excluding V_{DD} and V_{SS}	I	12.5	mA
Storage Temperature Range	T_{STG}	-65 to + 150	°C

NOTE: Voltages referenced to V_{SS}

NOTE

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

14.3 Operating Temperature Range

Rating	Symbol	Value	Unit
Operating Temperature Range MC68HC705CJ4 (Standard)	T_A	T_L to + T_H 0 to +70	°C

14.4 Thermal Characteristics

Characteristic	Symbol	Value	Unit
Thermal Resistance 44-Lead Plastic Quad Flat Pack	θ_{JA}	170	°C/W

14.5 Power Considerations

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

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

where:
T_A = ambient temperature in °C
θ_{JA} = package thermal resistance, junction to ambient in °C/W
$P_D = P_{INT} + P_{I/O}$
$P_{INT} = I_{DD} \times V_{DD}$ = chip internal power dissipation
$P_{I/O}$ = power dissipation on input and output pins (user-determined)

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

Ignoring $P_{I/O}$, the relationship between P_D and T_J is approximately:

$$P_D = \frac{K}{T_J + 273 \,°C} \tag{2}$$

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

$$K = P_D \times T_A + 273\,°C + \theta_{JA} \times P_D \tag{3}$$

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

14.6 Recommended DC Operating Characteristics

Rating	Symbol	Min	Typ	Max	Unit
Operating Voltage f_{OP} = 2.1 MHz f_{OP} = 1.0 MHz	V_{DD}	4.5 3.0	5.0 —	5.5 5.5	V
Programming Voltage	V_{PP}	12.0	12.5	13.0	V

NOTE: Voltages referenced to V_{SS} = 0 Vdc, T_A = 25 °C

14.7 DC Electrical Characteristics[1] (4.5 to 5.5 Vdc)

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage (I_{LOAD} = 10 μA) (I_{LOAD} = −10.0 μA)	V_{OL} V_{OH}	— V_{DD} − 0.1	— —	0.1 —	V
Output High Voltage (I_{OH} = −0.8 mA) PA0–PA7, PB0–PB7, PC0–PC7, TCMP	V_{OH}	V_{DD} − 0.8	—	—	V
Output Low Voltage (I_{OH} = −1.6 mA) PA0–PA7, PB0–PB7, TCMP (I_{OH} = 10 mA) PC0–PC7	V_{OL}	— —	— —	0.4 0.8	V
Input High Voltage PA0–PA7, PB0–PB7, PC0–PC7, PD0–PD7, OSC1, TCAP	V_{IH}	V_{DD} x 0.7	—	V_{DD}	V
Input Low Voltage PA0–PA7, PB0–PB7, PC0–PC7, PD0–PD7, OSC1, TCAP	V_{IL}	V_{SS}	—	0.2 x V_{DD}	V

Supply Current (see Note 2)
Run (f_{op} = 2.1 MHz)
Wait (f_{op} = 2.1 MHz)
Stop
  25 °C
  0 °C to +70 °C
	I_{DD}	—
		—
		
		—
		—
			7.0
			3.0
			
			5.0
			—
				10
				4
				
				10
				20
					mA
					mA
					
					μA
					μA

I/O Ports Hi-Z Leakage Current PA0–PA7, PB0–PB7, PC0–PC7, PD0–PD7	I_{OZ}	—	—	1.0	μA
Input Current $\overline{RESET}$, OSC1	I_{IN}	—	—	1.0	μA
Capacitance Ports (as Input or Output) $\overline{RESET}$	C_{OUT} C_{IN}	— —	— —	12 8	pF

NOTES:
1. V_{DD} = 4.5 Vdc ≤ V_{DD} ≤ 5.5 Vdc, V_{SS} = 0 Vdc, T_A = 0 °C to +70 °C, unless otherwise noted
2. $\overline{RESET}$, $\overline{IRQ}$, SCK, TCAP, SDIO, and SCK2 are equipped with Schmitt trigger inputs.

14.8 DC Electrical Characteristics[1] (3.0 to 4.5 Vdc)

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage (I_{LOAD} = 10.0 μA) (I_{LOAD} = −10.0 μA)	V_{OL} V_{OH}	— $V_{DD} - 0.1$	— —	0.1 —	V
Output High Voltage (I_{OH} = 0.2 mA) PA0–PA7, PB0–PB7, PC0–PC7, TCMP	V_{OH}	$V_{DD} - 0.3$	—	—	V
Output Low Voltage (I_{OH} = 0.4 mA) PA0–PA7, PB0–PB7, TCMP (I_{OH} = 2.5 mA) PC0–PC7	V_{OL}	— —	— —	0.3 0.6	V
Input High Voltage PA0–PA7, PB0–PB7, PC0–PC7, PD0–PD7, OSC1, TCAP	V_{IH}	V_{DD} x 0.7	—	V_{DD}	V
Input Low Voltage PA0–PA7, PB0–PB7, PC0–PC7, PD0–PD7, OSC1, TCAP	V_{IL}	V_{SS}	—	0.2 x V_{DD}	V

Supply Current (see Note 2)
Run (f_{op} = 1.0 MHz)
Wait (f_{op} = 1.0 MHz)
Stop
    25 °C
    0 °C to +70 °C
	I_{DD}	—
		—
		
		—
		—
			3.0
			1.5
			
			3.0
			—
				4
				2
				
				6
				12
					mA
					mA
					
					μA
					μA

I/O Ports Hi-Z Leakage Current PA0–PA7, PB0–PB7, PC0–PC7, PD0–PD7	I_{OZ}	—	—	1.0	μA
Input Current $\overline{RESET}$, OSC1	I_{IN}	—	—	1.0	μA
Capacitance Ports (as Input or Output) $\overline{RESET}$	C_{OUT} C_{IN}	— —	— —	12 8	pF

NOTES:
1. V_{DD} = 3.0 Vdc ≤ V_{DD} ≤ 4.5 Vdc, V_{SS} = 0 Vdc, T_A = 0 °C to +70°C, unless otherwise noted
2. $\overline{RESET}$, $\overline{IRQ}$, SCK, TCAP, SDIO, and SCK2 are equipped with Schmitt trigger inputs.

14.9 Control Timing[(1)] (4.5 to 5.5 Vdc)

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation (± 5%) Crystal Option Square Wave Signal (External Clock Source)	f_{OSC}	— —	4.2 4.2	MHz
Internal Operating Frequency (f_{osc}/2) Crystal Square Wave Signal	f_{OP}	— —	2.1 2.1	MHz
Cycle Time (1/f_{op})	t_{CYC}	476	—	ns
$\overline{RESET}$ Pulse Width	t_{RL}	1.5	—	t_{CYC}
Interrupt Pulse Width Low (Edge-Triggered)	t_{ILIH}	100	—	ns
Interrupt Pulse Period	t_{ILIL}	Note 2	—	ns
OSC1 Pulse Width	t_{OH}, t_{OL}	90	—	ns

NOTES:
1. V_{DD} = 4.5 Vdc ≤ V_{DD} ≤ 5.5 Vdc, T_A = 0 °°C to +70°C, unless otherwise noted
2. The minimum period t_{ILIL} should not be less than the number of cycles to execute the interrupt service routine plus 21 t_{CYC}.

14.10 Control Timing[(1)] (3.0 to 4.5 Vdc)

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation (± 5%) Crystal Option Square Wave Signal (External Clock Source)	f_{OSC}	— —	2.0 2.0	MHz
Internal Operating Frequency (f_{osc}/2) Crystal Square Wave Signal	f_{OP}	— —	1.0 1.0	MHz
Cycle Time (1/f_{op})	t_{CYC}	1000	—	ns
$\overline{RESET}$ Pulse Width	t_{RL}	1.5	—	t_{CYC}
Interrupt Pulse Width Low (Edge-Triggered)	t_{ILIH}	200	—	ns
Interrupt Pulse Period	t_{ILIL}	Note 2	—	ns
OSC1 Pulse Width	t_{OH}, t_{OL}	200	—	ns

NOTES:
1. V_{DD} = 3.0 Vdc ≤ V_{DD} ≤ 4.5 Vdc, V_{SS} = 0 Vdc, T_A = 0 °°C to +70°C, unless otherwise noted
2. The minimum period t_{ILIL} should not be less than the number of cycles to execute the interrupt service routine plus 21 t_{CYC}.

SECTION 15
MECHANICAL SPECIFICATIONS

15.1 Introduction

The MC68HC(7)05CJ4 is available in a 44-lead quad flat pack (QFP) package. Package dimensions for this package were not available at time of this publication.

To make sure that you have the latest case outline specifications, contact one of the following:

- Local Motorola Sales Office

- Motorola Mfax
    - Phone 602-244-6609
    - EMAIL 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.

15.2 44-Lead QFP Package

At time of publication, the package dimension drawing for the 44-lead QFP package was not available.

SECTION 16
ORDERING INFORMATION

16.1 Introduction

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

16.2 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 **16.3 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.

16.3 Application Program Media

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

- Macintosh®[1] 3 1/2-inch diskette (double-sided 800K or double-sided high-density 1.4M)

- MS-DOS®[2] or PC-DOS[TM][3] 3 1/2-inch diskette (double-sided 720K or double-sided high-density 1.44M)

- MS-DOS® or PC-DOS[TM] 5 1/4-inch diskette (double-sided double- density 360K or double-sided high-density 1.2M)

---

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.

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.

NOTE

Begin the application program at the first user ROM location. Program addresses must correspond exactly to the available on-chip user ROM addresses as shown in the memory map. **Write $00 in all non-user ROM locations or leave all non-user ROM locations blank**. Refer to the current MCU ordering form for additional requirements. Motorola may request pattern re-submission if non-user areas contain any non-zero 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.

16.4 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 non-user ROM code, such as self-check code. Motorola sends the customer a computer printout of the listing verify file along with a listing verify form.

To aid the customer in checking the listing verify file, Motorola will program the listing verify file into customer-supplied blank 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.

16.5 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.

16.6 MC Order Numbers

Table 16-1 shows the MC order numbers for the available package types.

Table 16-1. MC Order Numbers

MC Order Number	Operating Temperature Range
MC68HC(7)05CJ4FB	0° to 70°C

NOTE: FB = 44-lead quad flat pack (QFP) package

How to reach us:
  MFAX: RMFAX0@email.sps.mot.com - TOUCHTONE (602) 244-6609
  INTERNET: http://Design-net.com
  USA/EUROPE: Motorola Literature Distribution; P.O. Box 20912; Phoenix, Arizona 85036. 1-800-441-2447
  JAPAN: Nippon Motorola Ltd.; Tatsumi-SPD-JLDC, Toshikatsu Otsuki, 6F Seibu-Butsuryu-Center, 3-14-3 Tatsumi Koto-Ku,
    Tokyo 135, Japan. 03-3521-8315
  HONG KONG: Motorola Semiconductors H.K. Ltd.; 8B Tai Ping Industrial Park, 51 Ting Tok Road,
    Tai Po, N.T., Hong Kong. 852-26629298

MOTOROLA

HC05CJ4GRS/D