Freescale Semiconductor, Inc.

MC68HC05E5

General Release Specification

February 3, 1997

CSIC MCU Design Center
Austin, Texas

Freescale Semiconductor, Inc.

Freescale Semiconductor, Inc.

List of Sections

Freescale Semiconductor, Inc.

Freescale Semiconductor, Inc.

Table of Contents

Section 1. General Description

Section 2. Memory

Section 3. Central Processing Unit (CPU)

Section 4. Interrupts

Section 5. Resets

Section 6. Operating Modes

Section 7. Input/Output (I/O) Ports

Section 8. Timer

Section 9. Phase-Locked Loop (PLL) Synthesis

Section 10. Computer Operating Properly (COP) Watchdog

Freescale Semiconductor, Inc.

Section 11. Motorola Bus (M Bus) Interface

Section 12. Synchronous Serial Interface (SSI)

Section 13. Instruction Set

Section 14. Electrical Specifications

Section 15. Mechanical Data

Section 16. Ordering Information

List of Figures

List of Tables

Freescale Semiconductor, Inc.

Freescale Semiconductor, Inc.

Section 1. General Description

1.1 Contents

1.2 Introduction

The MC68HC05E5 is a low-cost introduction to the M68HC05 Family of microcontrollers (MCUs). The HC05 central processing unit (CPU) core has been enhanced with a 15-stage multifunctional timer and programmable phase-locked loop (PLL). The MCU is available in a 28-pin package and has two 8-bit input/output (I/O) ports and one 4-bit I/O port. The 8-Kbyte memory map includes 384 bytes of random access memory (RAM) and 5120 bytes of user read-only memory (ROM). The MC68HC705E5 serves as an erasable, programmable ROM (EPROM) based emulation device for the MC68HC05E5.

General Description

1.3 Features

Features of the MC68HC05E5 include:

- Low Cost

- HC05 Core

- 28-Pin Package

- On-Chip Oscillator (Crystal or Ceramic Resonator)

- Phase-Locked Loop (PLL) Synthesizer with Programmable Speed

- Synchronous Serial Interface (SSI) with Interrupts and Most Significant Bit (MSB) or Least Significant Bit (LSB) First

- M-Bus (I^2C) Communication Port

- 5120 Bytes of User ROM (Including 16 Bytes of User Vectors)

- 384 Bytes of On-Chip RAM

- 15-Stage Multifunctional Timer with Programmable Input

- Real-Time Interrupt Circuit

- Computer Operating Properly (COP) Watchdog Timer Mask Option

- Custom Periodic Interrupt Circuit

- 20 Bidirectional I/O Lines

- Single-Chip Mode

- Self-Check Mode

- Power-Saving Stop and Wait Modes

- Edge-Only Sensitive or Edge- and Level-Sensitive Interrupt Trigger Mask Option

- STOP Instruction Disable Mask Option

- System Control and Status Register

- Illegal Address Reset
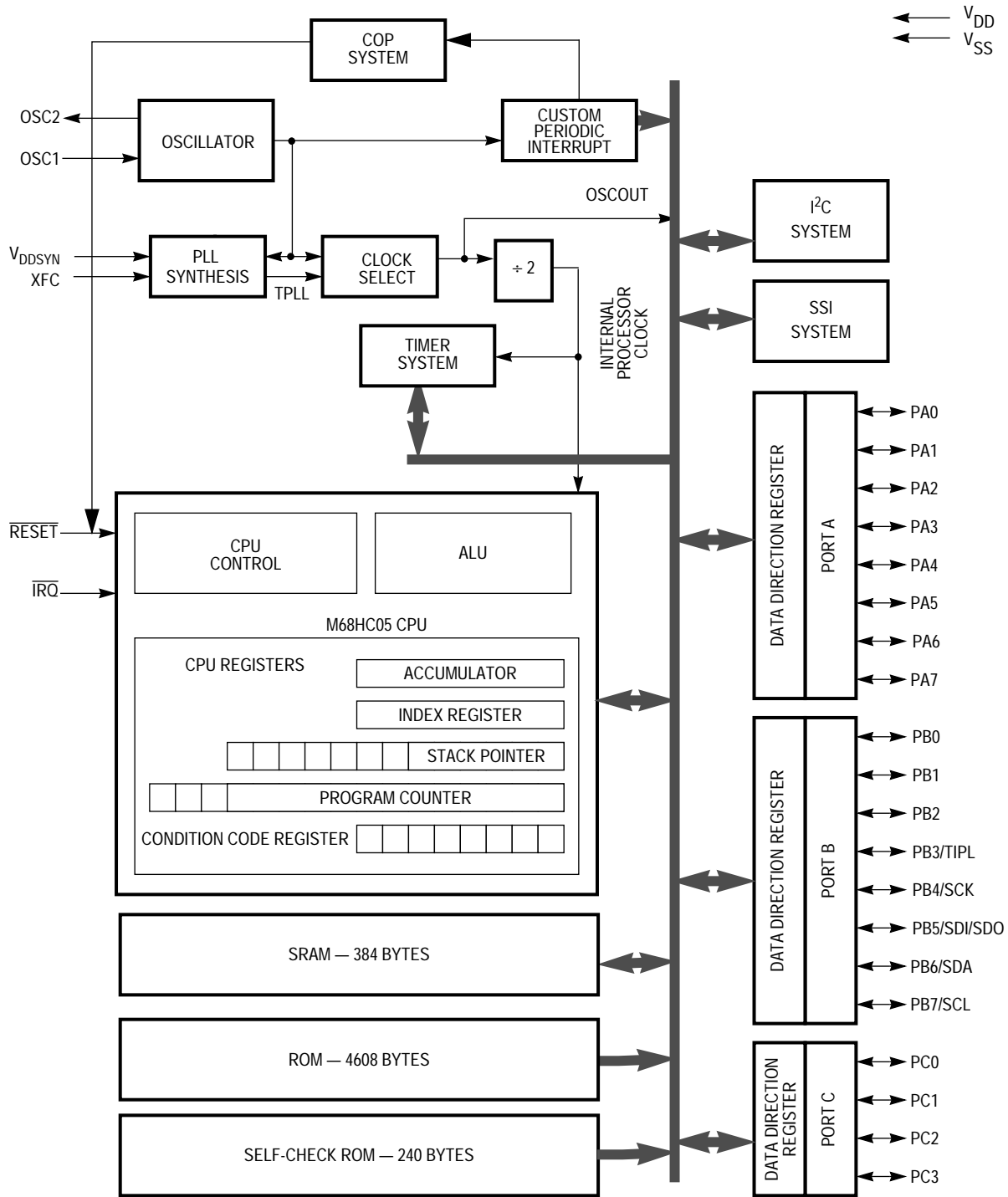
Figure 1-1. Block Diagram

1.4 Mask Options

The M68HC05E5 has four mask options:

1. STOP instruction (enable/disable)

2. $\overline{\text{IRQ}}$ (edge-sensitive only or edge- and level-sensitive)

3. COP watchdog timer (enable/disable)

4. CPI Rate (1 second, 0.5 second, or 0.25 second)

NOTE: *A line over a signal name indicates an active low signal. For example, $\overline{\text{RESET}}$ is active low.*

1.5 Functional Pin Description

Figure 1-2 shows the single-chip mode pinout for the MC68HC05E5. Refer to the following subsections for a description of the pins.

$\overline{\text{IRQ}}$	1 ●	28	XFC
$\overline{\text{RESET}}$	2	27	V_{DDSYN}
OSC1	3	26	PA0
OSC2	4	25	PA1
PB7/SCL	5	24	PA2
PB6/SDA	6	23	PA3
PB5/SDIO	7	22	PA4
PB4/SCK	8	21	PA5
PB3/TIPL	9	20	PA6
PB2	10	19	PA7
PB1	11	18	PC0
PB0	12	17	PC1
V_{DD}	13	16	PC2
V_{SS}	14	15	PC3
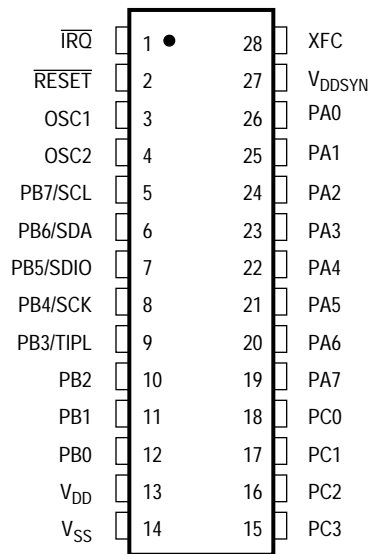
Figure 1-2. Single-Chip Mode Pinout

1.5.1 V_{DD} and V_{SS}

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

1.5.2 $\overline{IRQ}$

The maskable interrupt request ($\overline{IRQ}$) has a programmable option that provides two different choices of interrupt triggering sensitivity. The options are:

1. Negative edge-sensitive triggering only

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

The MCU completes the current instruction before it responds to the interrupt request. When $\overline{IRQ}$ goes low for at least one t_{ILIH}, a logic 1 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 1, and the interrupt mask bit (I bit) in the condition code register is clear, the MCU then begins the interrupt sequence.

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

The $\overline{IRQ}$ pin contains an internal Schmitt trigger as part of its input to improve noise immunity. Refer to **Section 4. Interrupts** for more detail.

NOTE: *The voltage on the $\overline{IRQ}$ pin affects the mode of operation. For additional information, see* **Section 6. Operating Modes***.*

1.5.3 OSC1 and OSC2

These pins provide control input for an on-chip clock oscillator circuit which can optionally drive a PLL clock. A crystal, a ceramic resonator, or an external signal connects to these pins providing a system clock. The oscillator frequency is two times the internal bus rate if the PLL is not used.

Crystal

Figure 1-3 (a) shows the recommended circuit for using a crystal. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time.

Ceramic Resonator

A ceramic resonator may be used in place of the crystal in cost-sensitive applications. **Figure 1-3 (a)** shows the recommended circuit for using a ceramic resonator. The manufacturer of the particular ceramic resonator being considered should be consulted for specific information.

External Clock

An external clock should be applied to the OSC1 input with the OSC2 pin not connected (refer to **Figure 1-3 (b)**). This setup can be used if the user does not wish to run the CPU with a 32.768-kHz crystal or the PLL frequencies are not suitable for the application.

(a) Crystal/Ceramic Resonator
    Oscillator Connections

(b) External Clock Source
    Connections
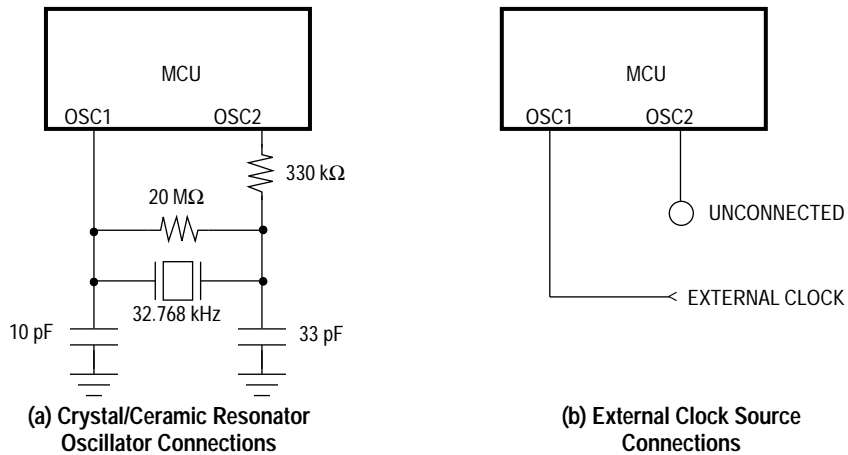
Figure 1-3. Oscillator Connections

General Description

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

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

1.5.5 PA0–PA7

These eight I/O lines comprise port A. The state of any pin is software programmable and all port A lines are configured as input during power-on or reset. See **7.6 Input/Output Programming** for additional information.

1.5.6 PB0–PB7

These eight I/O lines comprise port B. The state of any pin is software programmable and all port B lines are configured as input during power-on or reset. PB7 (SCL) and PB6 (SDA) can be configured as an M-bus interface. (Refer to **Section 11. Motorola Bus (M Bus) Interface** for M-bus pin configurations). PB3–PB5 (TIPL, SCK, and SDIO) can be configured as a synchronous serial interface (SSI). Refer to **Section 12. Synchronous Serial Interface (SSI)** and to **7.6 Input/Output Programming** for additional information.

1.5.7 PC0–PC3

These four I/O lines comprise port C. The state of any pin is software programmable and all port C lines are configured as input during power-on or reset. **7.6 Input/Output Programming** for additional information.

1.5.8 XFC

This pin provides a means for connecting an external filter capacitor to the synthesizer PLL filter. For additional information concerning this capacitor, see **Section 9. Phase-Locked Loop (PLL) Synthesis**.

1.5.9 V_{DDSYN}

This pin provides a separate power connection to the PLL synthesizer which should be at the same potential as V_{DD}.

NOTE: *Any unused inputs and I/O ports should be tied to an appropriate logic level (either V_{DD} or V_{SS}). Although the I/O ports of the MC68HC05E5 do not require termination, it is recommended to reduce the possibility of static damage.*

Section 2. Memory

2.1 Contents

2.2 Introduction

The MC68HC05E5 has an 8-Kbyte memory map, consisting of user read-only memory (ROM), user random access memory (RAM), self-check ROM, control registers, and input/output (I/O). Refer to **Figure 2-1** for the memory map and **Figure 2-2** for the register map.

2.3 ROM

The user ROM consists of 5120 bytes located from $0B00 to $1EFF, with 16 additional bytes of user vectors from $1FF0 to $1FFF. The self-check ROM and vectors are located from $1F00 to $1FEF.

2.4 RAM

The user RAM, including the stack area, consists of 384 bytes located from $0080 to $01FF. The stack begins at address $00FF. The stack pointer can access 64 bytes of RAM from $00FF to $00C0. Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

2.5 Memory Map

$0000	I/O 32 BYTES	0000
$001F		0031
$0020	UNUSED 96 BYTES	0032
$007F		0127
$0080	RAM 128 BYTES	0128
$00BF		0191
$00C0	↑ STACK 64 BYTES	0192
$00FF		0255
$0100	RAM 256 BYTES	0256
$01FF		0511
$0200	UNUSED 2304 BYTES	0512
$0AFF		2815
$0B00	USER ROM 5120 BYTES	2816
$1EFF		7935
$1F00	SELF-CHECK ROM & VECTORS 240 BYTES	7936
$1FEF		8175
$1FF0	USER VECTORS 16 BYTES	8176
$1FFF		8191

PORT A DATA REGISTER	$00
PORT B DATA REGISTER	$01
PORT C DATA REGISTER	$02
UNUSED	$03
PORT A DATA DIRECTION REGISTER	$04
PORT B DATA DIRECTION REGISTER	$05
PORT C DATA DIRECTION REGISTER	$06
PLL CONTROL REGISTER	$07
TIMER CONTROL & STATUS REGISTER	$08
TIMER COUNTER REGISTER	$09
SSI REGISTERS	$0A-$0C
UNUSED	$0D-$11
CPI CONTROL & STATUS REGISTER	$12
SCSR REGISTER	$13
I^2C (M -BUS) REGISTERS	$18
⋮	
	$1C
UNUSED ⋮	$1D
	$1E
RESERVED	$1F

UNUSED	$1FF0
UNUSED	$1FF1
M BUS VECTOR (HIGH BYTE)	$1FF2
M BUS VECTOR (LOW BYTE)	$1FF3
SSI VECTOR (HIGH BYTE)	$1FF4
SSI VECTOR (LOW BYTE)	$1FF5
CPI VECTOR (HIGH BYTE)	$1FF6
CPI VECTOR (LOW BYTE)	$1FF7
TIMER VECTOR (HIGH BYTE)	$1FF8
TIMER VECTOR (LOW BYTE)	$1FF9
IRQ VECTOR (HIGH BYTE)	$1FFA
IRQ VECTOR (LOW BYTE)	$1FFB
SWI VECTOR (HIGH BYTE)	$1FFC
SWI VECTOR (LOW BYTE)	$1FFD
RESET VECTOR (HIGH BYTE)	$1FFE
RESET VECTOR (LOW BYTE)	$1FFF

Figure 2-1. Memory Map

2.6 Register Summary

Addr.	Register	Bit 7	6	5	4	3	2	1	Bit 0
$0000	Port A Data Register								
$0001	Port B Data Register								
$0002	Port C Data Register	0	0	0	0				
$0003	Unimplemented								
$0004	Port A Data Direction Register								
$0005	Port B Data Direction Register								
$0006	Port C Data Direction Register								
$0007	PLL Control Register	0	BCS	0	BWC	PLLON	VCOTST	PS1	PS0
$0008	Timer Control and Status Register.	TOF	RTIF	TOFE	RTIE	TOFA	RTIFA	RT1	RT0
$0009	Timer Counter Register								
$000A	SSI Control Register	SIE	SE	LSBF	MSTR	CPOL	SDIR	SR1	SR0
$000B	SSI Status Register	SF	DCOL	0	0	0	0	0	TIPL
$000C	SSI Data Register	D7	D6	D5	D4	D3	D2	D1	D0
$000D	Unimplemented								
$000E	Unimplemented								
$000F	Unimplemented								
$0010	Unimplemented								
$0011	Unimplemented								
$0012	CPI Control and Status Register.	—	CPIF	—	CPIE	—	—	—	—
$0013	System Control and Status Register	0	0	0	STOPR	ILADR	COPR	CRS1	CRS0
$0014	Unimplemented								
$0015	Unimplemented								
$0016	Unimplemented								
$0017	Unimplemented								

　　= Unimplemented

Figure 2-2. I/O Registers

Addr.	Register	Bit 7	6	5	4	3	2	1	Bit 0
$0018	M-Bus Address Register	MAD7	MAD6	MAD5	MAD4	MAD3	MAD2	MAD1	
$0019	M Bus Frequency Divider Register				FD4	FD3	FD2	FD1	FD0
$001A	M Bus Control Register	MEN	MIEN	MSTA	MTX	TXAK	MMUX		
$001B	M Bus Status Register	MCF	MAAS	MBB	MAL		SRW	MIF	MXAK
$001C	M Bus Data I/O Register	MD7	MD6	MD5	MD4	MD3	MD2	MD1	MD0
$001D	Unimplemented								
$001E	Unimplemented								
$001F	Reserved	R	R	R	R	R	R	R	R
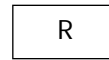
    ▨ = Unimplemented   R = Reserved

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

Section 3. Central Processing Unit (CPU)

3.1 Contents

3.2 Introduction

The MCU contains five registers as shown in **Figure 3-1**. The interrupt stacking order is shown in **Figure 3-2**.
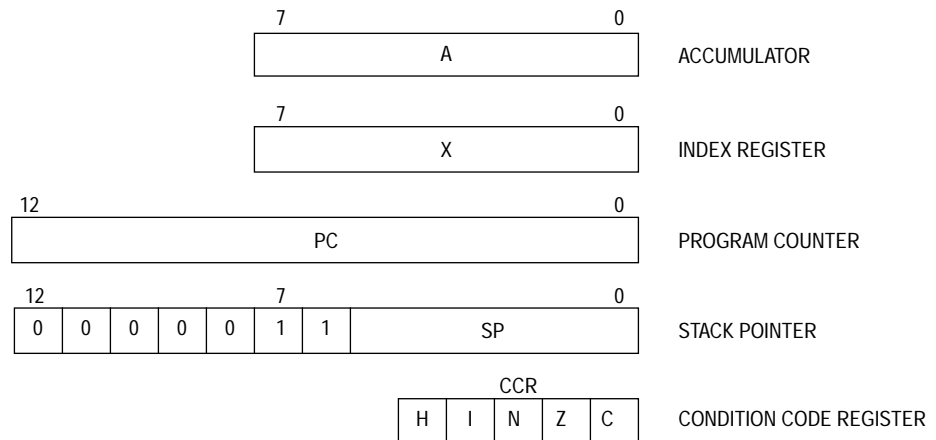
Figure 3-1. Programming Model

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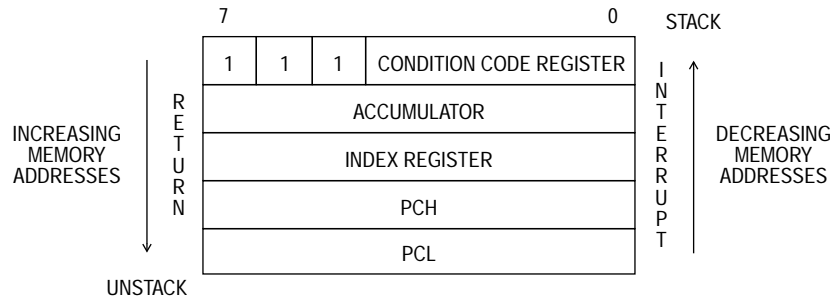
NOTE: Since the stack pointer decrements during pushes, the PCL is stacked first, followed by PCH, etc. Pulling from the stack is in the reverse order.

Figure 3-2. Stacking Order

3.3 Accumulator

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

Bit 7	6	5	4	3	2	1	Bit 0
A							
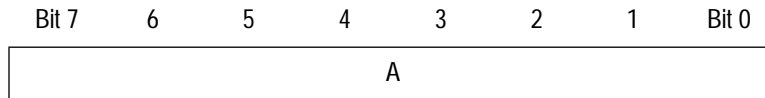
Figure 3-3. Accumulator (A)

3.4 Index Register

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.

Bit 7	6	5	4	3	2	1	Bit 0
X							
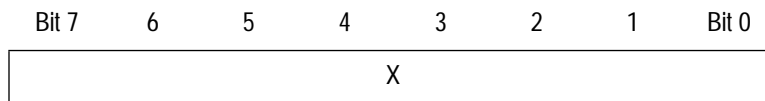
Figure 3-4. Index Register (X)

3.5 Condition Code Register

The CCR is a 5-bit register in which the H, N, Z, and C bits are used to indicate the results of the instruction just executed, and the I bit is used to enable interrupts. These bits can be tested individually by a program, and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.

Bit 4	3	2	1	Bit 0
H	I	N	Z	C

Figure 3-5. Condition Code Register (CCR)

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 timer and external interrupt are masked (disabled). If an interrupt occurs while this bit is set, the interrupt is latched and processed as soon as the I 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 also is affected during bit test and branch instructions and during shifts and rotates.

3.6 Stack Pointer

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 then is decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

When accessing memory, the seven most significant bits (MSB) are permanently set to 0000011. These seven bits are appended to the six least significant register bits to produce an address within the range of $00FF to $00C0. Subroutines and interrupts may use up to 64 (decimal) locations. If 64 locations are exceeded, the stack pointer wraps around and loses the previously stored information. A subroutine call occupies two locations on the stack; an interrupt uses five locations.

Bit 12	11	10	9	8	7	6	5	4	3	2	1	Bit 0
0	0	0	0	0	1	SP						
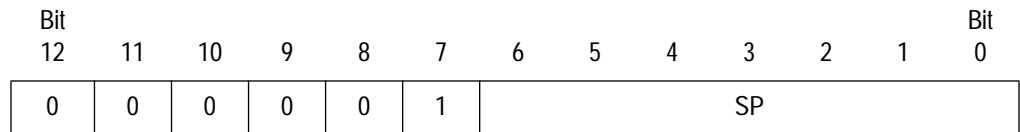
Figure 3-6. Stack Pointer (SP)

3.7 Program Counter

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

Bit 12	11	10	9	8	7	6	5	4	3	2	1	**Bit 0**
PC												
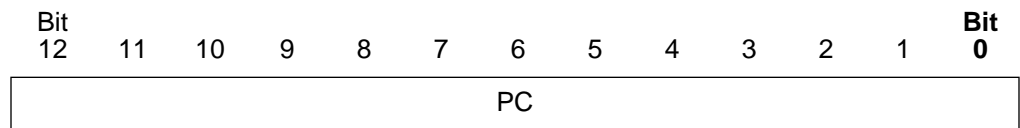
Figure 3-7. Program Counter (PC)

NOTE: *The HC05 CPU core is capable of addressing 16-bit locations. For this implementation, however, the addressing registers are limited to an 8-Kbyte memory map.*

Section 4. Interrupts

4.1 Contents

4.2 Introduction

The MCU can be interrupted six different ways: the five maskable hardware interrupts ($\overline{\text{IRQ}}$, timer, CPI, SSI, and M bus) and the nonmaskable 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.

Interrupts

Unlike $\overline{\text{RESET}}$, hardware interrupts do not cause the current instruction execution to be halted, but are considered pending until the current instruction is complete. 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 the corresponding interrupt enable bit is set, the processor proceeds with interrupt processing; otherwise, the next instruction is fetched and executed.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction, regardless of the I-bit state.

Table 4-1. Vector Address for Interrupts and Reset

Register	Flag Name	Interrupts	CPU Interrupt	Vector Address
N/A	N/A	Reset	$\overline{\text{RESET}}$	$1FFE–$1FFF
N/A	N/A	Software	SWI	$1FFC–$1FFD
N/A	N/A	External Interrupt	$\overline{\text{IRQ}}$	$1FFA–$1FFB
TCSR	TOF	Timer Overflow	TIMER	$1FF8–$1FF9
N/A	RTIF	Real-Time Interrupt	TIMER	$1FF8–$1FF9
CPICSR	CPIF	Custom Periodic Interrupt	CPI	$1FF6–$1FF7
SSR	SF	Synchronous Serial Interrupt	SSI	$1FF4–$1FF5
MSR	MIF	M-Bus Interrupt	M Bus	$1FF2–$1FF3

4.3 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. See **Figure 4-1** and **Figure 4-2**. 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 as described in **Section 5. Resets**.

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 clock running. This "rest" state of the processor can be cleared by reset, an external interrupt ($\overline{\text{IRQ}}$), or timer interrupt. There are no special wait vectors for these individual interrupts.

4.4 Software Interrupt (SWI)

The SWI is an executable instruction and a nonmaskable 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.

4.5 External Interrupt

If the I bit of the condition code reister (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 $\overline{\text{IRQ}}$. It is then synchronized internally and serviced by the interrupt service routine located at the address specified by the contents of $1FFA and $1FFB.

Either a level-sensitive and edge-sensitive trigger or an edge-sensitive-only trigger is available as a mask option.

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.*

4.6 Timer Interrupt

Two different timer interrupt flags cause a timer interrupt whenever they are set and enabled. The interrupt flags and enable bits are located in the timer control and status 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 $1FF8 and $1FF9. For additional information, refer to **8.3 Timer Control and Status Register**.

```
                        ┌──────────────┐
                        │    FROM      │
                        │    RESET     │
                        └──────┬───────┘
                               │
                    Y      ╱───┴───╲
              ◄────────────   IS
                             I BIT
                             SET?
                          ╲───┬───╱
                             N│
                       ╱──────┴──────╲        Y     ┌──────────────┐
                          IRQ          ───────────► │  CLEAR IRQ   │
                        EXTERNAL                     │   REQUEST    │
                       INTERRUPT?                    │    LATCH.    │
                       ╲──────┬──────╱               └──────────────┘
                             N│
                       ╱──────┴──────╲        Y
                         TIMER         ───────────►
                        INTERNAL
                       INTERRUPT?                    ┌──────────────┐
                       ╲──────┬──────╱               │    STACK     │
                             N│                      │  PC, X, A, CC.│
                       ╱──────┴──────╲        Y      └──────┬───────┘
                          CPI          ───────────►         │
                        INTERNAL                     ┌──────┴───────┐
                       INTERRUPT?                    │    SET       │
                       ╲──────┬──────╱               │    I BIT.    │
                             N│                      └──────┬───────┘
                       ┌──────┴───────┐              ┌──────┴───────────────┐
                       │ FETCH NEXT   │              │ LOAD PC FROM:        │
                       │ INSTRUCTION. │              │   SWI:   $1FFC–$1FFD │
                       └──────┬───────┘              │   IRQ:   $1FFA–$1FFB │
                             N│                      │   TIMER: $1FF8–$1FF9 │
                       ╱──────┴──────╲               │   CPI:   $1FF6–$1FF7 │
                          SWI          Y             │   SSI:   $1FF4–$1FF5 │
                       INSTRUCTION  ───────────►     │   M BUS: $1FF2–$1FF3 │
                          ?                          └──────────────────────┘
                       ╲──────┬──────╱
                             N│
                       ╱──────┴──────╲        Y      ┌──────────────────┐
                          RTI          ───────────►  │ RESTORE RESISTERS│
                       INSTRUCTION                   │   FROM STACK     │
                          ?                          │   CC, A, X, PC   │
                       ╲──────┬──────╱               └──────────────────┘
                             N│
                       ┌──────┴───────┐
                       │   EXECUTE    │
                       │ INSTRUCTION. │
                       └──────────────┘
```
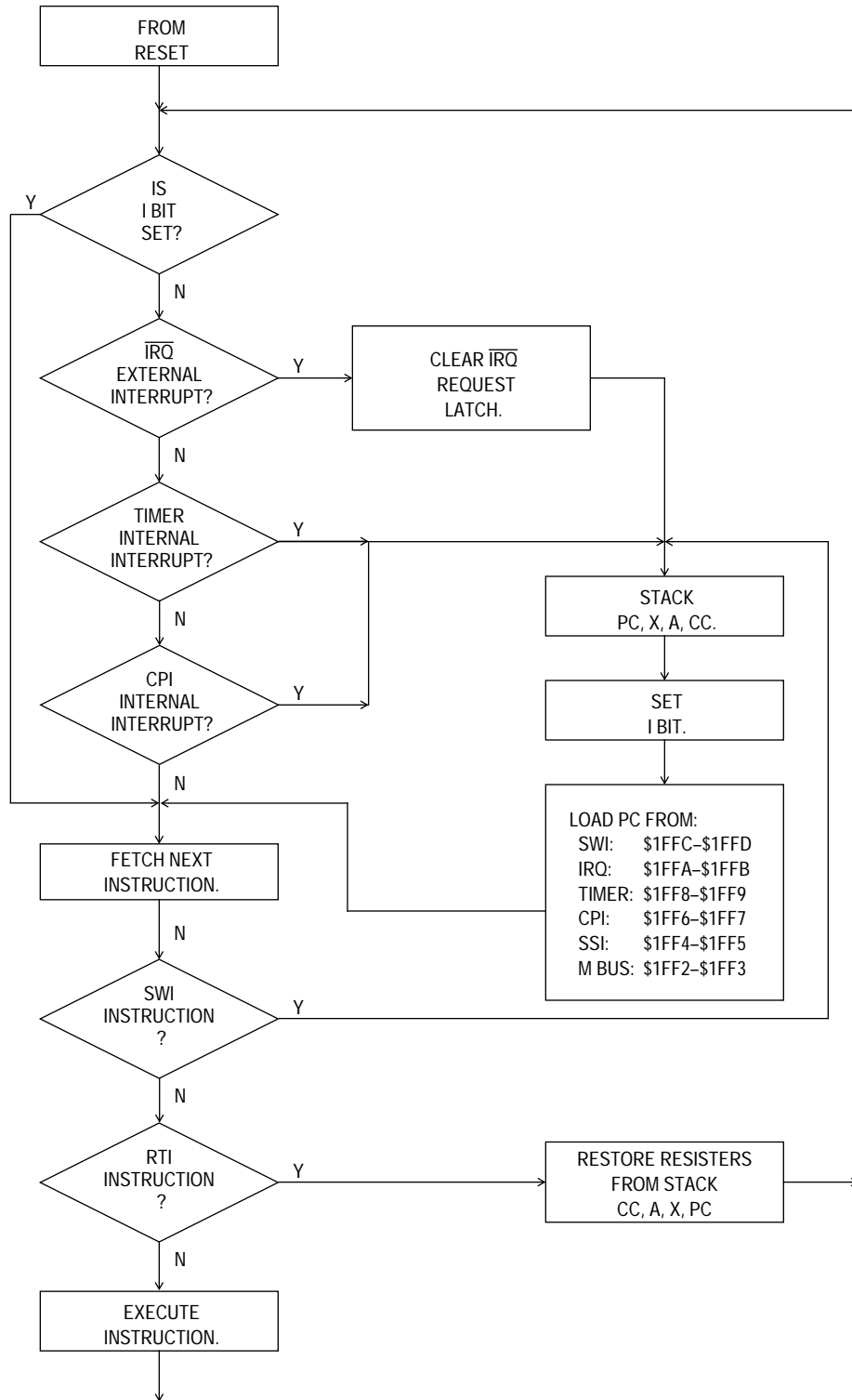
Figure 4-1. Interrupt Processing Flowchart

Freescale Semiconductor, Inc.
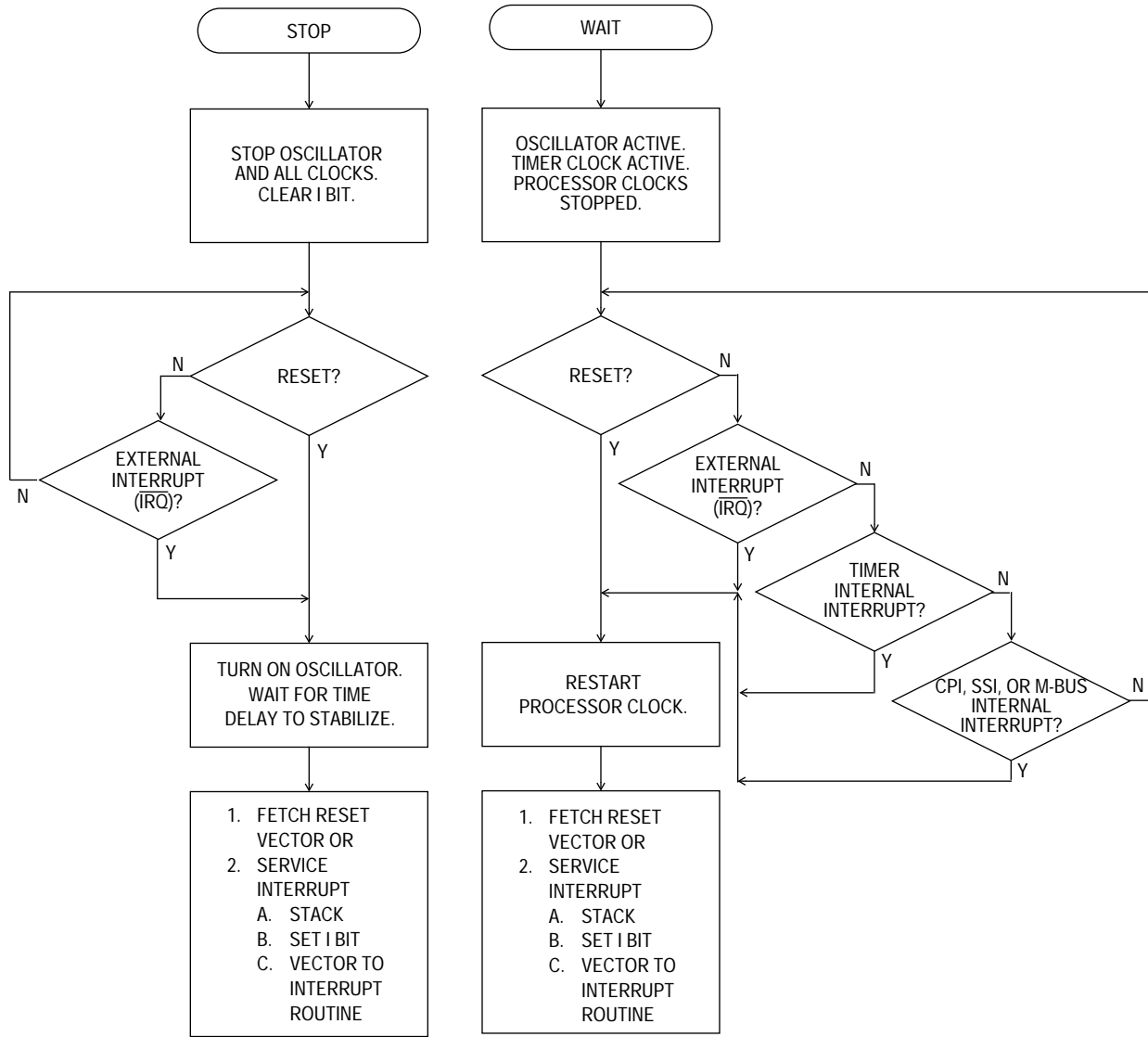
Figure 4-2. STOP/WAIT Flowcharts

4.7 Custom Periodic Interrupt (CPI)

The CPI flag and enable bits are located in the CPI control and status register (CPICSR). A CPI interrupt will vector to the interrupt service routine located at the address specified by the contents of memory location $1FF6 and $1FF7.

The custom periodic interrupt is mask programmable to a 0.25 second, 0.5 second, or 1 second interrupt. The interrupt is generated from the 32-kHz OSC1 input by a 15-bit counter. This interrupt is under the control of the custom periodic interrupt control and status register located at $12.

Address $0012

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	0	CPIF	0	CPIE	0	0	0	0
Reset:	0	0	0	0	0	0	0	0

**Figure 4-3. Custom Periodic Interrupt Control
and Status Register (CPICSR)**

CPIF — Custom Periodic Interrupt Flag

CPIF is a clearable, read-only status bit and is set when the 15-bit counter changes from $7FFF to $0000. A CPU interrupt request will be generated if CPIE is set. Clearing the CPIF is done by writing a zero to it. Writing a one to CPIF has no effect on the bit's value. Reset clears CPIF.

CPIE — Custom Periodic Interrupt Enable

When this bit is cleared, the CPI interrupts are disabled. When this bit is set, the CPU interrupt request is generated when the CPIF bit is set. Reset clears this bit.

4.8 Synchronous Serial Interface Interrupt (SSI)

The SSI flag and enable bits are located in the SSI control (SCR) and status (SSR) registers. An SSI interrupt will vector to the interrupt service routine located at the address specified by the contents of memory locations $1FF4 and $1FF5. For additional information, refer to **12.4 SSI Registers**.

4.9 M-Bus (I^{2}C) Interrupt (M Bus)

The MIF flag and enable bits are located in the M-bus status (MSR) and control (MCR) registers. An M-bus interrupt will vector to the interrupt service routine located at the address specified by the contents of memory locations $1FF2 and $1FF3. For further information, refer to **11.6 M-Bus Registers**.

4.10 Operation During Stop Mode

The timer system is cleared and the CPI counter is halted when going into stop mode. When stop mode is exited by an external interrupt or an external $\overline{\text{RESET}}$, the internal oscillator will resume, followed by a 4064-cycle internal processor oscillator stabilization delay. The timer system counter is then cleared and operation resumes. The CPI will continue counting once the oscillator resumes and does not wait for the oscillator to stabilize.

4.11 Operation During Wait Mode

The CPU clock halts during wait mode, but the timer and CPI remain active. A timer interrupt or custom periodic interrupt, SSI, and M bus will cause the processor to exit wait mode if the interrupts are enabled.

Section 5. Resets

5.1 Contents

5.2 Introduction

The MCU can be reset from five sources: one external input and four internal restart conditions. The $\overline{\text{RESET}}$ pin is an input with a Schmitt trigger as shown in **Figure 5-1**. All the internal peripheral modules will be reset by the internal reset signal (RST). Refer to **Figure 5-2** for reset timing detail.

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

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

NOTE: *Activation of the RST signal is generally referred to as a reset of the device, unless otherwise specified.*
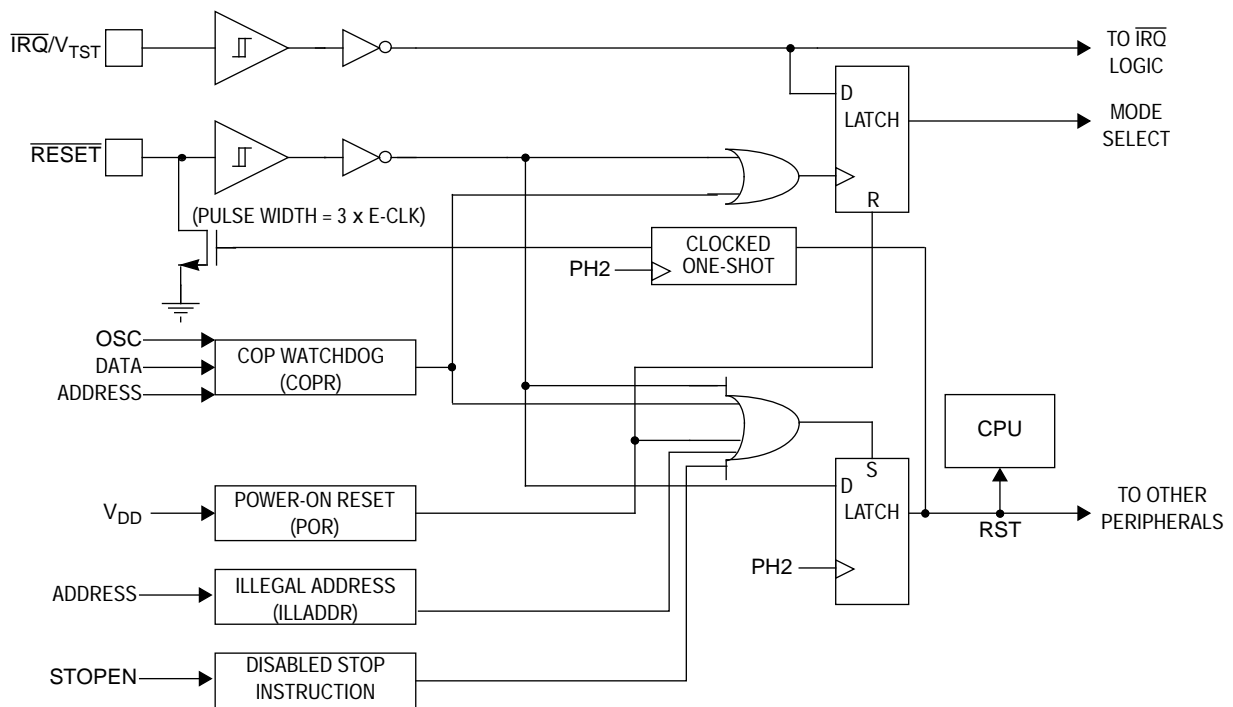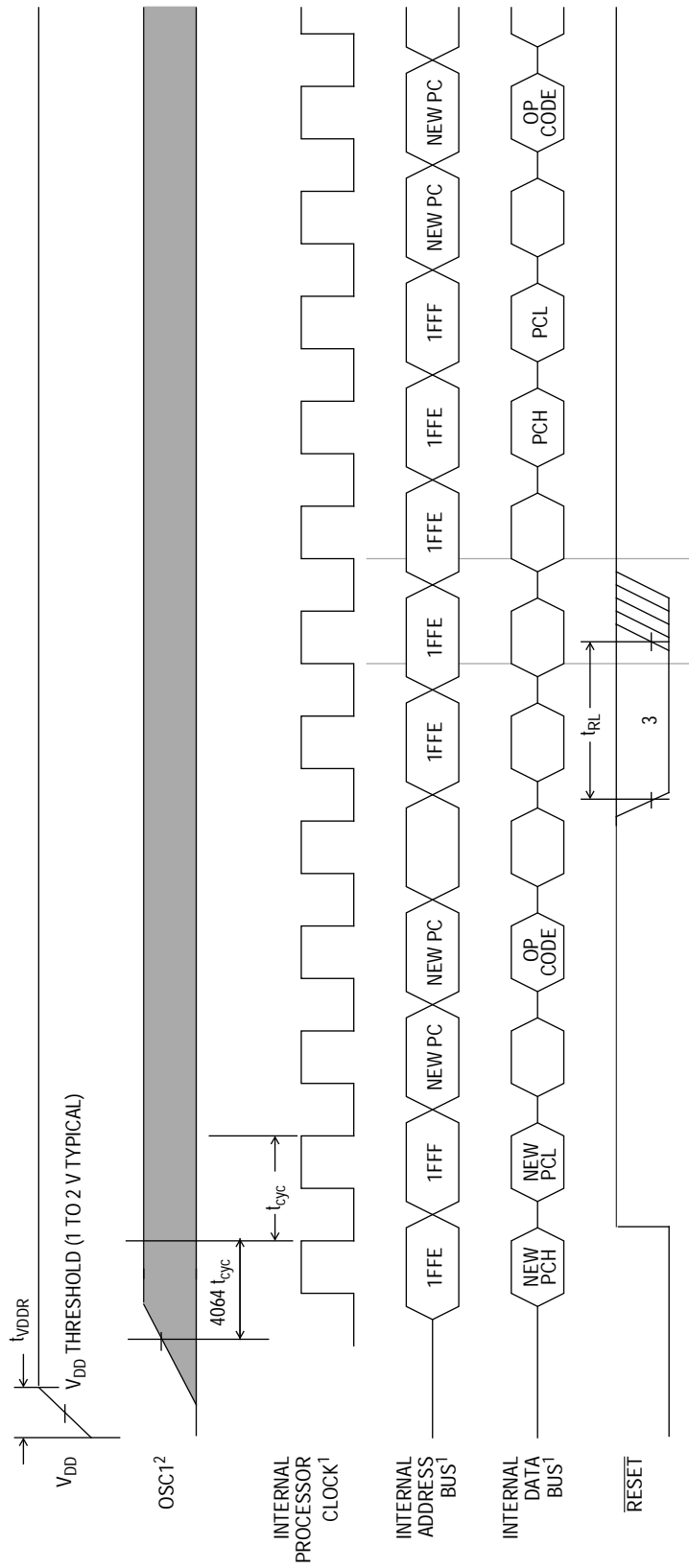
Figure 5-1. Reset Block Diagram

Figure 5-2. RESET and POR Timing Diagram

NOTES:
1. Internal timing signal and bus information are not available externally.
2. OSC1 line is not meant to represent frequency. It is only used to represent time.
3. The next rising edge of the internal processor clock following the rising edge of RESET initiates the reset sequence.

The $\overline{\text{RESET}}$ pin can also act as an open-drain output. It will be pulled to a low state by an internal pulldown that is activated by any reset source. This $\overline{\text{RESET}}$ pulldown device will be asserted only by three to four cycles of the internal clock, PH2 (PH2 period = E clock period), or as long as an internal reset source is asserted. When the external $\overline{\text{RESET}}$ pin is asserted, the pulldown device will be turned on for the three to four internal clock cycles only.

5.4 Internal Resets

The four internally generated resets are the initial power-on reset function, the COP watchdog timer reset, the illegal address detector, and the disabled STOP instruction. The only reset sources that can alter the MCU's operating mode are termination of the external $\overline{\text{RESET}}$ input or the internal COP watchdog timer. The other internal resets will not have any effect on the mode of operation when their reset state ends. All internal resets will also assert (pull to logic 0) the external $\overline{\text{RESET}}$ pin for the duration of the reset or three to four internal clock cycles, whichever is longer.

5.5 Power-On Reset (POR)

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

The POR will generate the RST signal which will reset the CPU. If any other reset function is active at the end of the 4064-cycle delay, the RST signal will remain in the reset condition until the other reset condition(s) end.

POR will activate the $\overline{\text{RESET}}$ pin pulldown device connected to the pin. V_{DD} must drop below V_{POR} for the internal POR circuit to detect the next rise of V_{DD}.

5.6 Computer Operating Properly Reset (COPR)

The MCU contains a watchdog timer that automatically times out if not reset (cleared) within a specific time by a program reset sequence. If the COP watchdog timer is allowed to timeout, an internal reset is generated to reset the MCU. Regardless of an internal or external reset, the MCU comes out of a COP reset according to the standard rules of mode selection.

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

The COP watchdog reset will activate the internal pulldown device connected to the $\overline{\text{RESET}}$ pin.

5.6.1 Resetting the COP

Preventing a COP reset is done by writing a logic 0 to the COPF bit. This action will reset the counter and begin the timeout period again. The COPF bit is bit 0 of address $1FF0. A read of address $1FF0 will return user data programmed at that location.

5.6.2 COP During Wait Mode

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

5.6.3 COP During Stop Mode

When the stop enable mask option is selected, stop mode disables the oscillator circuit and thereby turns the clock off for the entire device. The COP counter will be reset when stop mode is entered. If a reset is used to exit stop mode, the COP counter will be held in reset during the 4064 cycles of startup delay. If any operable interrupt is used to exit stop mode, the COP counter will not be reset during the 4064-cycle startup delay and will have that many cycles already counted when control is returned to the program.

5.6.4 COP Watchdog Timer Considerations

If enabled by a mask option, the COP watchdog timer is active in all modes of operation (disabled in test and self-check modes). If the COP watchdog timer is selected by a mask option, any execution of the STOP instruction (either intentional or inadvertent due to the CPU being disturbed) will cause the oscillator to halt and prevent the COP watchdog timer from timing out. Therefore, it is recommended that the STOP instruction should be disabled if the COP watchdog timer is enabled.

If the COP watchdog timer is selected by a mask option, the COP will reset the MCU when it times out. Therefore, it is recommended that the COP watchdog should be disabled for a system that must have intentional uses of the wait mode for periods longer than the COP timeout period.

The recommended interactions and considerations for the COP watchdog timer, STOP instruction, and WAIT instruction are summarized in **Figure 5-1**.

Table 5-1. COP Watchdog Timer Recommendations

IF the Following Conditions Exist:		THEN the COP Watchdog Timer Should Be:
STOP Instruction	Wait Time	
Converted to Reset	WAIT Time Less Than COP Timeout	Enable or Disable COP by Mask Option
Converted to Reset	WAIT Time More Than COP Timeout	Disable COP by Mask Option
Acts as STOP	Any Length WAIT Time	Disable COP by Mask Option

5.7 Illegal Address Reset

When an opcode fetch occurs from an address which is not implemented in the RAM ($0080–$01FF) or ROM ($0F00–$1FFF), the part is reset automatically.

Section 6. Operating Modes

6.1 Contents

6.2 Introduction

The MCU has two modes of operation: single-chip mode and self-check mode. This section describes these modes as well as the two low-power modes: stop mode and wait mode.

Refer to **Table 6-1** for the conditions required to go into each of the operating modes.

Table 6-1. Operating Mode Conditions

RESET	IRQ	PB1	Mode
⌐	$V_{SS} - V_{DD}$	$V_{SS} - V_{DD}$	Single-Chip
	V_{TST}	V_{DD}	Self-Check

$V_{TST} = 2 \times V_{DD}$

6.3 Single-Chip Mode

In single-chip mode, the address and data buses are not available externally, but there are two 8-bit input/output (I/O) ports and one 4-bit I/O port. This mode allows the MCU to function as a self-contained microcontroller, with maximum use of the pins for on-chip peripheral functions. All address and data activity occurs within the MCU. Single-chip mode is entered on the rising edge of $\overline{\text{RESET}}$ if the $\overline{\text{IRQ}}$ pin is within normal operating range.

Refer to **Figure 1-2** for the single-chip user mode pinout diagram.

6.4 Self-Check Mode

The self-check mode provides an internal check to determine if the device is functional.

6.5 Low-Power Modes

The following subsections provide a description of the low-power 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 (and COP watchdog timer) operation.

During stop mode, the I bit in the CCR is cleared to enable external interrupts. All other registers, including the bits in the TCSR, and memory remain unaltered. All input/output lines remain unchanged. The processor can be brought out of stop mode only by an external interrupt or $\overline{\text{RESET}}$.

The STOP instruction can be disabled by a mask option. When disabled, the STOP instruction causes a chip reset.

Refer to **Figure 4-2** and to **4.10 Operation During Stop Mode** for additional information.

6.5.2 Wait Mode

The WAIT instruction places the MCU in a low power-consumption mode, but the wait mode consumes more power than the stop mode. All CPU action is suspended, but the timer, CPI, COP, SSI, and M bus remain active. An interrupt from the timer, SSI, or M bus can 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 timer, SSI, and/or IIC modules may be enabled to allow a periodic exit from the wait mode.

Refer to **Figure 4-2** and to **4.11 Operation During Wait Mode** for additional information.

6.5.3 Data-Retention Mode

The contents of RAM and CPU registers are retained at supply voltages as low as 2.0 Vdc. This is called the data-retention mode where the data is held, but the device is not guaranteed to operate. $\overline{\text{RESET}}$ must be held low during data-retention mode.

Section 7. Input/Output (I/O) Ports

7.1 Contents

7.2 Introduction

In single-chip mode, 20 lines are arranged as two 8-bit input/output (I/O) ports and one 4-bit I/O port. These ports are programmable as either inputs or outputs under software control of the data direction registers.

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 (DDR).

7.3 Port A

Port A is an 8-bit bidirectional port which does not share any of its pins with other subsystems. The port A data register is at $0000 and the 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.

7.4 Port B

Port B is an 8-bit bidirectional port which does share some of its pins with other subsystems. The address of the port B data register is $0001 and the DDR is at address $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 bit to output mode. Refer to **Section 11. Motorola Bus (M Bus) Interface** and **Section 12. Synchronous Serial Interface (SSI)** for descriptions of port B behavior while either module is enabled.

7.5 Port C

Port C is a 4-bit bidirectional port which does not share any of its pins with other subsystems. The port C data register is at $0002 and the 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.

7.6 Input/Output Programming

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 DDR with each port having an associated DDR. Any port A, port B, or port C pin is configured as an output if its corresponding DDR bit is set to a logic 1. A pin is configured as an input if its corresponding DDR bit is cleared to a logic 0.

At power-on or reset, all DDRs are cleared, which configures all port A, B, and C pins as inputs. The data direction registers are capable of being written to or read from 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. See **Table 7-1** and **Figure 7-1**.

Freescale Semiconductor, Inc.

Table 7-1. I/O Pin Functions

R/$\overline{W}$	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 of 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.
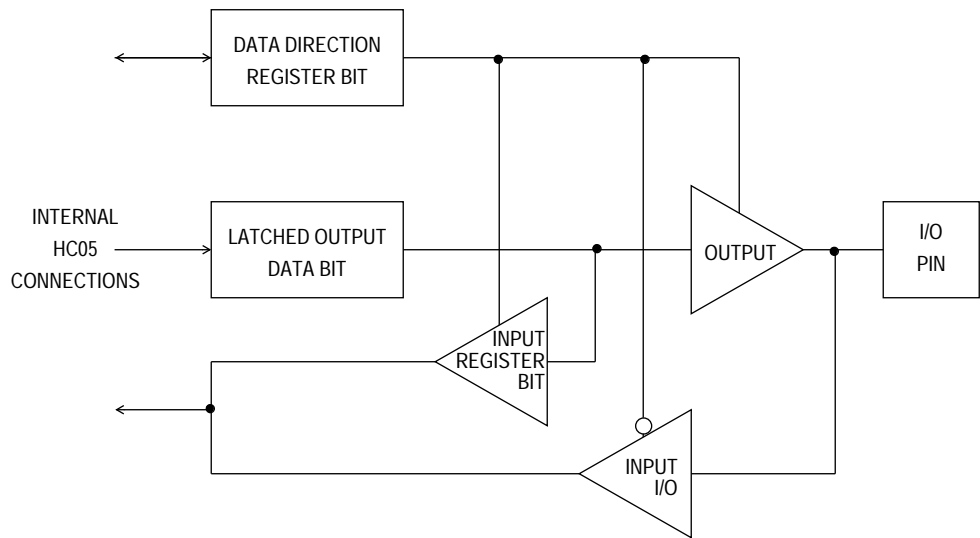
Figure 7-1. Port I/O Circuitry

Freescale Semiconductor, Inc.

Section 8. Timer

8.1 Contents

8.2 Introduction

The timer for this device is a 15-stage multifunctional ripple counter. The features include timer overflow, power-on reset (POR), and real-time interrupt.

As seen in **Figure 8-1**, the timer is driven by the output of the clock select circuit (as determined by the value of BCS in the PLLCR) and then 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 counter register (TCR) at address $09. A timer overflow function is implemented on the last stage of this counter, giving a possible interrupt at the rate of $f_{op}/1024$. Two additional stages produce the POR function at $f_{op}/4064$.

This circuit is followed by two more stages, with the resulting clock ($f_{op}/16,384$) driving the real-time interrupt circuit. The RTI circuit consists of three divider stages with a one-of-four selector. The RTI rate selector bit and the RTI and TOF enable bits and flags are located in the timer control and status register at location $0008.

MC68HC05E4 INTERNAL BUS
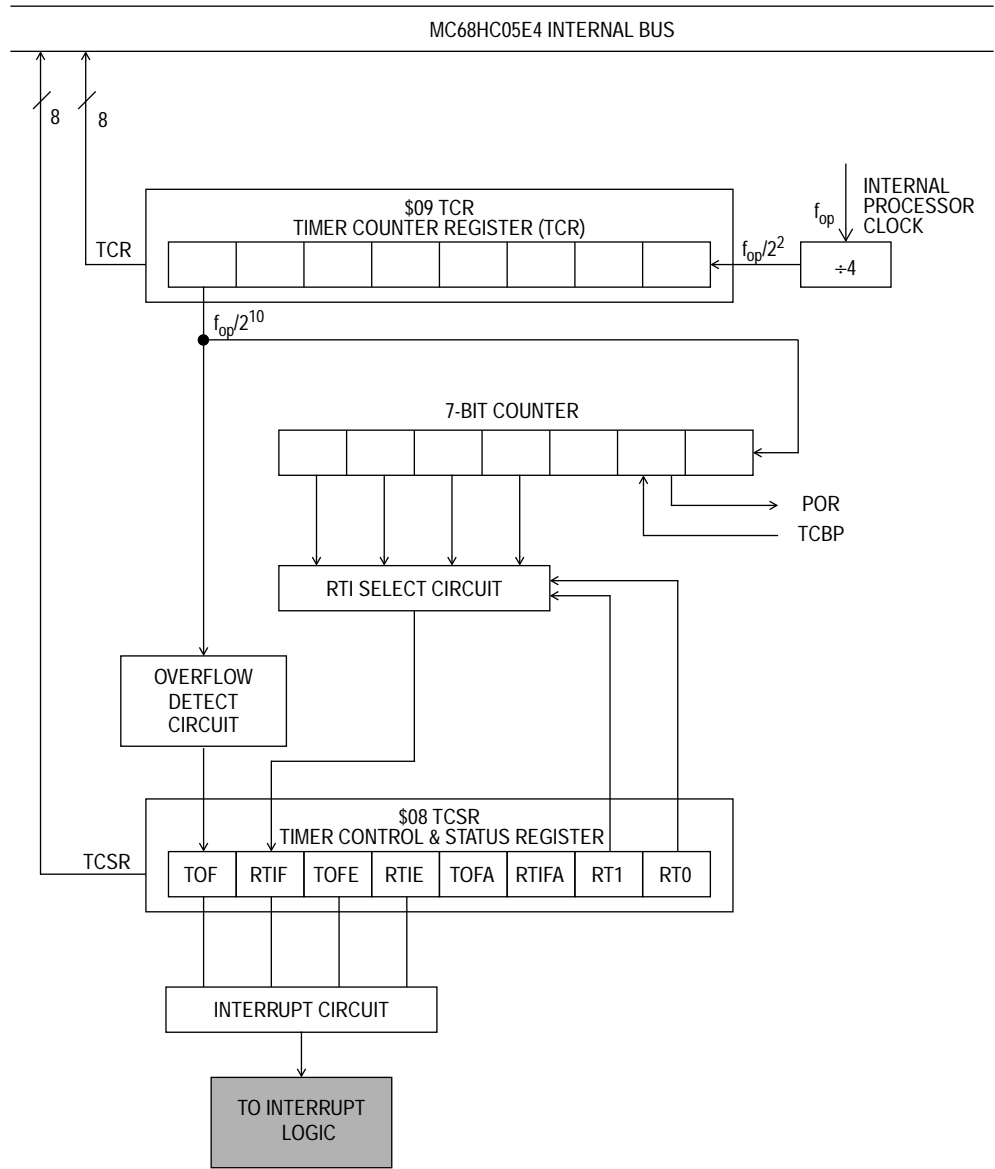
Figure 8-1. Timer Block Diagram

8.3 Timer Control and Status Register

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

Address: $0008

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	TOF	RTIF	TOFE	RTIE	TOFA	RTIFA	RT1	RT0
Reset:	0	0	0	0	0	0	1	1

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

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 logic 1 to TOFA. This is a read-only bit. Reset also clears TOF.

RTIF — Real-Time Interrupt Flag

The real-time interrupt circuit consists of a 3-stage divider and a one-of-four selector. The clock frequency that drives the RTI circuit is $f_{op}/2^{13}$ (or $f_{op}/8192$) with three additional divider stages giving a maximum interrupt period of four seconds at a crystal frequency of 32.768 kHz. RTIF is a clearable, read-only status bit and is set when the output of the chosen (one-of-four selection) stage goes active. A CPU interrupt request will be generated if RTIE is set. Clearing the RTIF is done by writing a logic 1 to RTIFA. Reset also clears RTIF.

TOFE — Timer Overflow 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.

TOFA — Timer Over Flow Flag Acknowledge

When a one is written to this bit location, the TOF flag bit is cleared. This bit always reads as a zero.

RTIFA — Real-Time Interrupt Flag Acknowledge

When a one is written to this bit location, the RTIF flag bit is cleared. This bit always reads as a zero.

RT1–RT0 — Real-Time Interrupt Rate Select

These two bits select one of four taps from the real-time interrupt circuit. **Table 8-1** shows the available interrupt rates with several f_{op} values. Reset sets RT0 and RT1, selecting the lowest periodic rate and therefore 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.

Table 8-1. RTI Rates

RT1flRT0	RT1 Rates at f_{op} Frequency Specified:					
	16.384 kHz	524 kHz	1.049 MHz	2.097 MHz	4.194 MHz	f_{op}
00	1 s	31.3 ms	15.6 ms	7.8 ms	3.9 ms	$2^{14} \div f_{op}$
01	2 s	62.5 ms	31.3 ms	15.6 ms	7.8 ms	$2^{15} \div f_{op}$
10	4 s	125 ms	62.5 ms	31.3 ms	15.6 ms	$2^{16} \div f_{op}$
11	8 s	250 ms	125.1 ms	62.5 ms	31.3 ms	$2^{17} \div f_{op}$

8.4 Timer Counter Register

The timer counter register (TCR) 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 f_{op} divided by four and can be used for various functions including a software input capture. Extended time periods can be attained using the TOF function to increment a temporary RAM storage location, thereby simulating a 16-bit (or more) counter.

Address: $0009

	Bit 7	6	5	4	3	2	1	Bit 0
Read:								
Write:								
Reset:	0	0	0	0	0	0	1	1

= Unimplemented
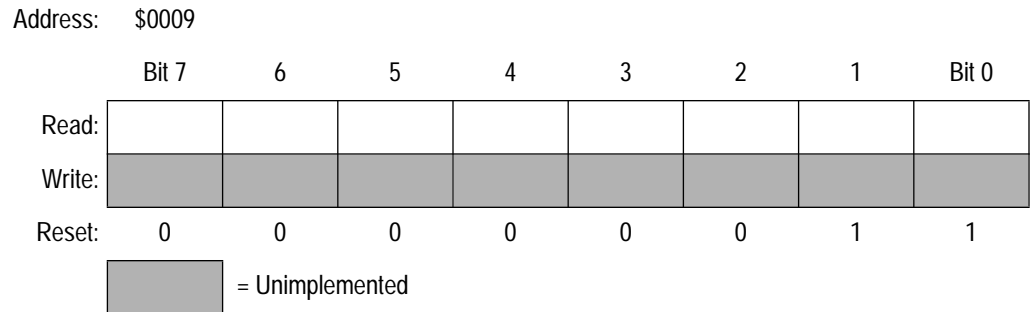
Figure 8-3. Timer Counter Register (TCR)

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

Section 9. Phase-Locked Loop (PLL) Synthesis

9.1 Contents

9.2 Introduction

The PLL consists of a variable bandwidth loop filter, a voltage-controlled oscillator (VCO), a feedback frequency divider, and a digital phase detector. The PLL requires an external loop filter capacitor (typically 0.1 µF) connected between XFC and V_{DDSYN}. This capacitor should be located as close to the chip as possible to minimize noise. V_{DDSYN} is the supply source for the PLL and should be bypassed to minimize noise. The V_{DDSYN} bypass cap should be as close as possible to the chip.

The phase detector compares the frequency and phase of the feedback frequency (t_{FB}) and the crystal oscillator reference frequency (t_{REF}) and generates the output, PCOMP, as shown in **Figure 9-1**. The output waveform is then integrated and amplified. The resultant DC voltage is applied to the voltage controlled oscillator. The output of the VCO is divided by a variable frequency divider of 256, 128, 64, or 32 to provide the feedback frequency for the phase detector.
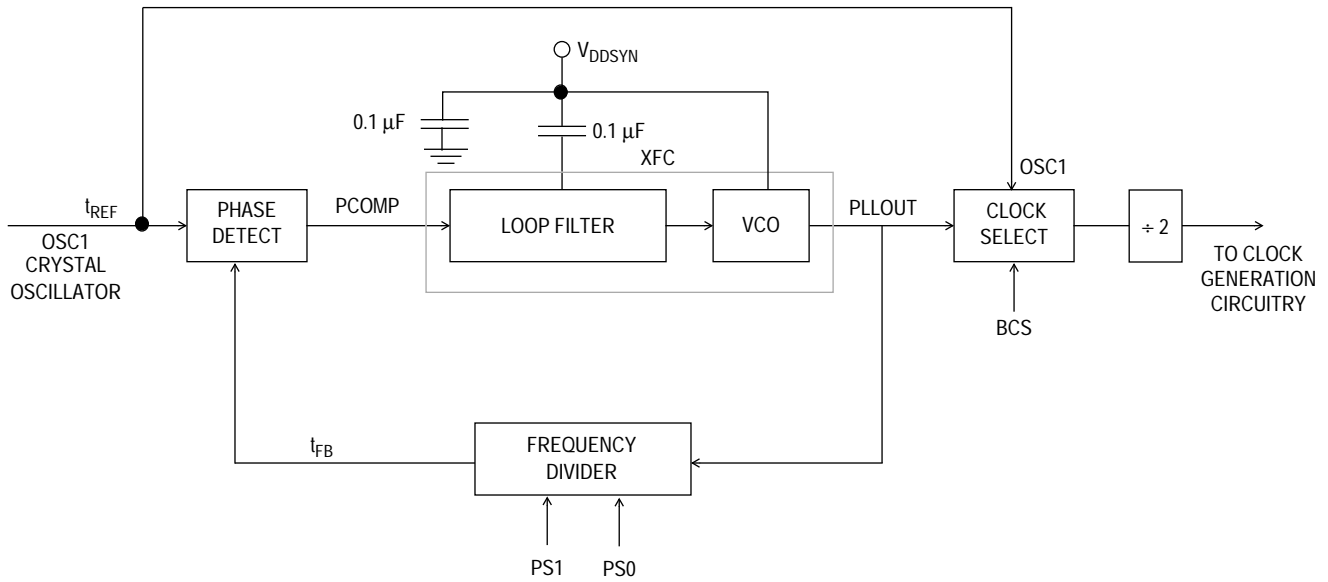
Figure 9-1. PLL Circuit

To change PLL frequencies, follow the procedure outlined here:

1. Clear BCS to enable the low-frequency bus rate.

2. Clear PLLON to disable the PLL and select high bandwidth.

3. Select the speed using PS1 and PS0.

4. Set PLLON to enable the PLL.

5. Wait a time of 90% t_{PLLS} for the PLL frequency to stabilize and select manual low bandwidth, wait another 10% t_{PLLS}.

6. Set BCS to switch to the high-frequency bus rate

The user cannot switch among the high speeds with the BCS bit set. Following the procedure above will prevent possible bursts of high frequency operation during the re-configuration of the PLL.

Whenever the PLL is first enabled, the wide bandwidth mode should be used. This enables the PLL frequency to ramp up quickly. When the output frequency is near the desired frequency, the filter is switched to the narrow bandwidth mode to make the final frequency more stable.

9.3 Phase-Locked Loop Control Register

This read/write register contains the control bits which select the PLL frequency and enable/disable the synthesizer.

Address: $0007

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	0	BCS	0	BWC	PLLON	VCOTST	PS1	PS0
Reset:	0	0	0	0	1	1	0	1

Figure 9-2. Phase-Locked Loop Control Register (PLLCR)

BCS — Bus Clock Select

When this bit is set, the output of the PLL is used to generate the internal processor clock. When clear, the internal bus clock is driven by the crystal (OSC1 ÷ 2). Once BCS has been changed, it may take up to 1.5 OSC1 cycles + 1.5 PLLOUT cycles to make the transition. During the transition, the clock select output will be held low and all CPU and timer activity will cease until the transition is complete. Before setting BCS, allow at least a time of t_{PLLS} after PLLON is set.

This bit cannot be set unless the PLLON bit is already set on a previous instruction. Reset clears this bit.

BWC — Bandwidth Control

This bit selects high bandwidth control when set and low bandwidth control when clear. The low bandwidth driver is always enabled, so this bit determines whether the high bandwidth driver is on or off. When the PLL is turned on, the BWC bit should be set to a logic 1 for a time of 90% t_{PLLS} to allow the PLL time to acquire a frequency close to the desired frequency. The BWC bit should then be cleared and software should delay for a time 10% t_{PLLS} to allow the PLL time to make the final adjustments. The PLL clock cannot be used (BCS bit set). Although it is NOT prohibited in hardware, the BCS bit should not be set unless the BWC bit is cleared and the proper delay times have been followed. The PLL will generate a lower jitter clock when the BWC bit is cleared. Reset clears this bit.

PLLON — PLL On

This bit activates the synthesizer circuit without connecting it to the control circuit. This allows the synthesizer to stabilize before it can drive the CPU clocks. When this bit is cleared, the PLL is shut off and the BCS bit cannot be set. (Setting the BCS bit would engage the disabled PLL onto the bus.) Reset sets this bit.

NOTE: *PLLON cannot be cleared unless the BCS bit has been cleared on a previous write to the register.*

VCOTST — VCO Test

This bit is used to isolate the loop filter from the VCO to facilitate testing. When cleared only in test or self-check modes, the low bandwidth mode of the PLL filter is disabled. When set, the loop filter operates as indicated by the value of the BWC bit. Reset sets this bit.

NOTE: *This bit is intended for use by Motorola to test and characterize the PLL. This bit cannot be cleared in user mode.*

PS1–PS0 — PLL Synthesizer Speed Select

These two bits select one-of-four taps from the PLL to drive the CPU clocks. These bits are used in conjunction with PLLON and BCS bits in the PLL control register. These bits should not be written if BCS in the PLLCR is at a logic high. Reset clears PS1 and sets PS0, choosing a bus clock frequency of 1.049 MHz.

Table 9-1. PS1 and PS0 Speed Selects with 32.768-kHz Crystal

PS1–PS0	CPU Bus Clock Frequency (f_{op})
0 0	524 kHz
0 1	1.049 MHz Reset Condition
1 0	2.097 MHz See Note Below
1 1	4.194 MHz See Note Below

NOTE:
For the standard MC68HC05E5, the 4.194-MHz bus clock frequency should never be selected, and the 2.097-MHz bus clock frequency should not be selected when running the part below V_{DD} = 4.5 V.

9.4 Operation During Stop Mode

The PLL is switched to low-frequency bus rate and is temporarily turned off when STOP is executed. Coming out of stop mode with an external IRQ, the PLL is turned on with the same configuration it had before going into STOP, with the exception of BCS which is reset. Otherwise, the PLL control register is in the reset condition.

9.5 Noise Immunity

The MCU should be insulated as much as possible from noise in the system. We recommend the following steps be taken to help prevent problems due to noise injection.

1. The application environment should be designed so that the MCU is not near signal traces which switch often, such as a clock signal.

2. The oscillator circuit for the MCU should be placed as close as possible to the OSC1 and OSC2 pins on the MCU.

3. All power pins should be filtered (to minimize noise on these signals) by using bypass capacitors placed as close as possible to the MCU.

See the application note *Designing for Electromagnetic Compatibility (EMC) with HCMOS Microcontrollers*, available through the Motorola Literature Distribution Center, Motorola document number AN1050/D.

Freescale Semiconductor, Inc.

Phase-Locked Loop (PLL) Synthesis

Section 10. Computer Operating Properly (COP) Watchdog

10.1 Contents

10.2 Introduction

The COP watchdog system is a mask-programmable feature which will generate a system reset if not serviced within the specified COP timeout period. The COP counter chain is derived from an output of the CPI circuit. This input signal is divided to give the COP reset rate selected by the first write to the system control and status register (SCSR) located at address $13.

A COP reset is done by writing a logic zero to bit 0 of address $1FF0. This will reset the COP counter chain and begin the timeout countdown again. The COP counter chain is also cleared when the MCU is in reset or stop mode.

Computer Operating Properly (COP) Watchdog

10.3 System Control and Status Register

The SCSR is a read/write register containing the control flags for the COP rate, COP inhibit, and IRQ level and status flags indicating the cause of the last reset. **Figure 10-1** shows the value of each bit in the SCSR when coming out of reset.

Address: $0013

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	0	0	0	STOPR	ILADR	COPR	CRS1	CRS0
Reset:	0	0	0	R	R	R	0	0

R = Determined by cause of previous reset

Figure 10-1. System Control and Status Register (SCSR)

NOTE: *The debounce time for the $\overline{IRQ}$ input must be shorter than the COP timeout period.*

STOPR — Illegal STOP Instruction Reset

STOPR is a read-only status bit. This bit is set by the execution of a STOP instruction when the STOP instruction option is disabled. This bit is cleared by POR, external reset, or COP reset.
   1 = Last reset was the execution of a disabled STOP instruction.
   0 = Last reset was not the execution of a disabled STOP instruction.

ILADR — Illegal Address Reset

ILADR is a read-only status bit. This bit is set by an ILADR reset, but is cleared by POR, external reset, or COP reset.
   1 = Last reset was an ILADR reset.
   0 = Last reset was not an ILADR reset.

COPR — COP Reset

COPR is a read-only status bit. This bit is set by a COP reset, but is cleared by POR, external reset, or illegal address reset.
   1 = Last reset was a COP reset.
   0 = Last reset was not a COP reset.

NOTE: *The COP watchdog reset is a mask option. Therefore, a COP reset will only occur when this option is enabled. This option cannot be disabled by software.*

CRS1 and CRS0 — COP Rate Select

The value of these two bits determines the COP timeout rate. These bits can be written only on the first write to this register after reset. If these bits are never written to, the COP reset rate will be set at one second. The COP counter chain is cleared when these bits are written.

NOTE: *Although these bits default to zero, the user should write to these bits to prevent subsequent writes from changing the COP rate.*

A bit set/clear for any bit in this register is executed as a read-modify-write of this register. If used as the first write to this register, further writes to CRS1 and CRS0 would not be valid, and the default value would be set.

Table 10-1. COP Rates at f_{osc} = 32.768 kHz

CRS1	CRS0	Minimum COP Rate
0	0	1 second
0	1	2 seconds
1	0	4 seconds
1	1	8 seconds

10.4 COP During Wait Mode

The CPU clock halts during wait mode, but the oscillator and the COP system are still active. The software should exit wait mode to service the COP system before the COP timeout period.

10.5 COP During Stop Mode

Prior to entry into stop mode, the COP should be cleared. This allows for proper stop recovery and eliminates a possible COP time out during stop mode recovery, if the COP was about to time out prior to the STOP instruction. If enabled, stop mode turns off the oscillator and, therefore, will stop the COP.

Section 11. Motorola Bus (M Bus) Interface

11.1 Contents

11.2 Introduction

Motorola bus (M bus) is a 2-wire, bidirectional serial bus which provides a simple, efficient way for data exchange between devices. It is fully compatible to I^{2}C bus standards and is similar to the MC68HC05T10.

This bus is suitable for applications that require frequent communications over a short distance between a number of devices. It also provides a flexibility that allows additional devices to be connected to the bus. The maximum data rate is limited to 100 Kbits and the maximum communication distance and number of devices that can be connected is limited by the maximum bus capacitance of 400 pF.

The M-bus system is a true multimaster bus including collision detection and arbitration to prevent data corruption if two or more masters intend to control the bus simultaneously. This feature provides the capability for complex applications with multiprocessor control. It may also be used for rapid testing and alignment of end products by way of external connections to an assembly-line computer.

11.3 M-Bus Interface Features

Features of the M-bus interface include:

- Fully Compatible with I²C Bus Standard

- Multimaster Operation

- Software Programmable for 1of 32 Different Serial Clock Frequencies

- Software Selectable Acknowledge Bit

- Interrupt Driven Byte-by-Byte Data Transfer

- Arbitration Lost Driven Interrupt with Automatic Mode Switching from Master to Slave

- Calling Address Identification Interrupt

- Generate/Detect the Start or Stop Signal

- Repeated Start Signal Generation

- Generate/Recognize the Acknowledge Bit

- Bus Busy Detection

11.4 M-Bus System Configuration

The M-bus system uses a serial data line (SDA) and a serial clock line (SCL) for data transfer. All devices connected to it must have open-drain or open-collector outputs and the logical AND function is performed on both lines by two pullup resistors.

11.5 M-Bus Protocol

Normally, a standard communication is composed of four parts: start signal, slave address transmission, data transfer, and stop signal. These are described briefly in the following subsections and illustrated in **Figure 11-1**.

11.5.1 Start Signal

When the bus is free (for example, no master device is engaging the bus and both SCL and SDA lines are at logical high), a master may initiate communication by sending a start signal. As shown in **Figure 11-1**, a start signal is defined as a high-to-low transition of SDA while SCL is high. This signal denotes the beginning of new data transfer (each data transfer may contain several bytes of data) and wakes up all slaves.
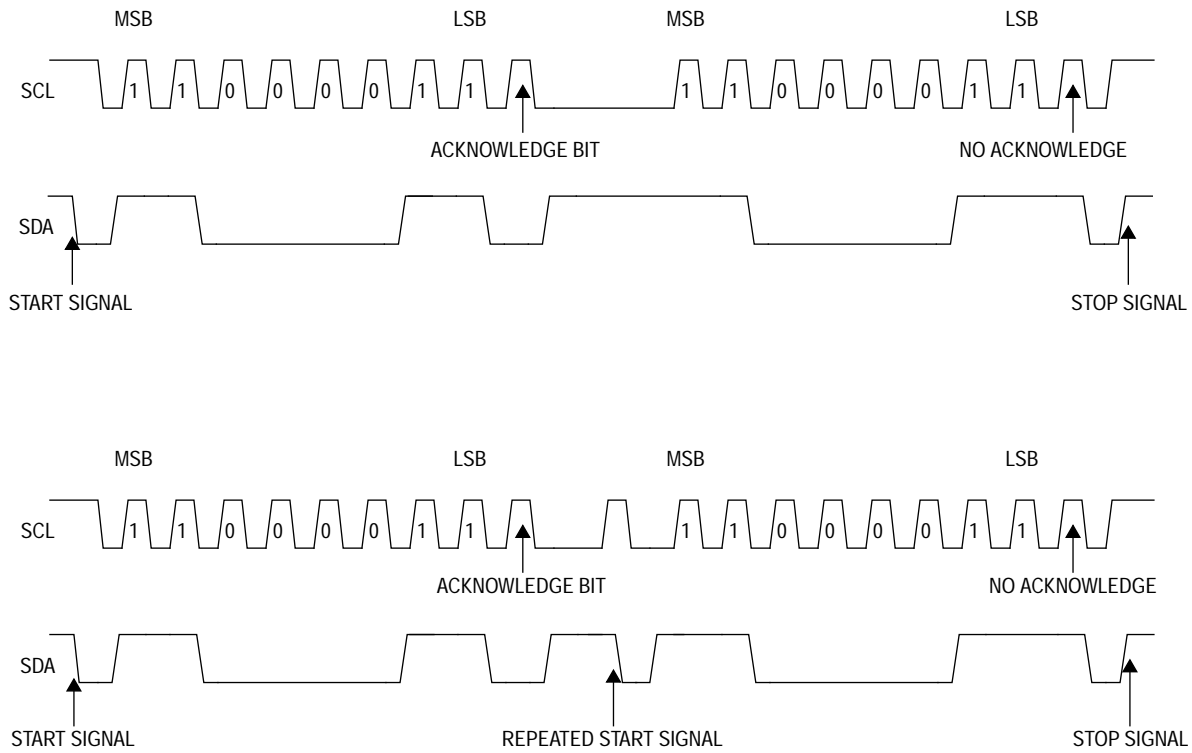
Figure 11-1. M-Bus Transmission Signal Diagram

11.5.2 Slave Address Transmission

Immediately after the start signal, the first byte of data transfer is the slave address transmitted by the master. This data is a 7-bit calling address followed by a R/$\overline{W}$ bit. The R/$\overline{W}$ bit tells the slave the desired direction of data transfer.

Only the slave with a matched address will respond by sending back an acknowledge bit. This acknowledge bit is accomplished by pulling SDA low on the ninth clock cycle. (See **Figure 11-1**.)

11.5.3 Data Transfer

Once a successful slave addressing is achieved, the data transfer can proceed byte by byte in the direction specified by the R/$\overline{W}$ bit sent by the calling master.

Each data byte is eight bits long. Data can be changed only when SCL is low and must be held stable while SCL is high as shown in **Figure 11-1**. The MSB is transmitted first and each byte has to be followed by an acknowledge bit. The acknowledge bit is signalled by the receiving device by pulling the SDA low on the ninth clock cycle. Therefore, one complete data byte transfer needs nine clock cycles.

If the slave receiver does not acknowledge the master, the SDA line should be left high by the slave. The master can then generate a stop signal to abort the data transfer or a start signal (repeated start) to commence a new transfer.

If the master receiver does not acknowledge the slave transmitter after a byte has been transmitted, it means an "end of data" to the slave. The slave should now release the SDA line for the master to generate a stop or start signal.

11.5.4 Repeated Start Signal

As shown in **Figure 11-1**, a repeated start signal is used to generate a start signal without first generating a stop signal to terminate the communication. This is used by the master to communicate with another slave or with the same slave in a different mode (transmit/receive mode) without releasing the bus.

11.5.5 Stop Signal

The master can terminate the communication by generating a stop signal to free the bus. However, the master may generate a start signal followed by a calling command without first generating a stop signal. This is called repeat start. A stop signal is defined as a low-to-high transition of SDA while SCL is at logical high. (See **Figure 11-1**.)

11.5.6 Arbitration Procedure

This interface circuit is a true multimaster system which allows more than one master to be connected to it. If two or more masters try to control the bus at the same time, a clock synchronization procedure determines the bus clock, for which the low period is equal to the longest clock low period and the high is equal to the shortest one among the masters. A data arbitration procedure determines the priority. The masters will lose arbitration if they transmit a logic 1 while another transmits logic 0. The losing masters will immediately switch over to slave receive mode and stop its data and clock outputs. In this case, the transition from master to slave mode will not generate a stop condition; however, a software bit will be set by hardware to indicate loss of arbitration.

11.5.7 Clock Synchronization

Since wired-AND logic is performed on the SCL line, a high-to-low transition will affect the devices connected to the bus. The devices start counting their low period and once a device's clock has gone low, it will hold the SCL line low until the clock high state is reached. However, the change of low to high in this device clock may not change the state of the SCL line if another device clock is still within its low period. Therefore, the synchronized clock SCL will be held low by the device with the longest low period. Devices with shorter low periods enter a high wait state during this time. (See **Figure 11-2**.) When all devices concerned have counted off their low period, the synchronized SCL line will be released and go high. There will then be no difference between the device clocks and the state of the SCL line and all devices will start counting their high periods. The first device to complete its high period will again pull the SCL line low.

11.5.8 Handshaking

The clock synchronization mechanism can be used as a handshake in data transfer. Slave devices may hold the SCL low after completion of one byte. In such cases, the device will halt the bus clock and force the master clock into a wait state until the slave releases the SCL line.
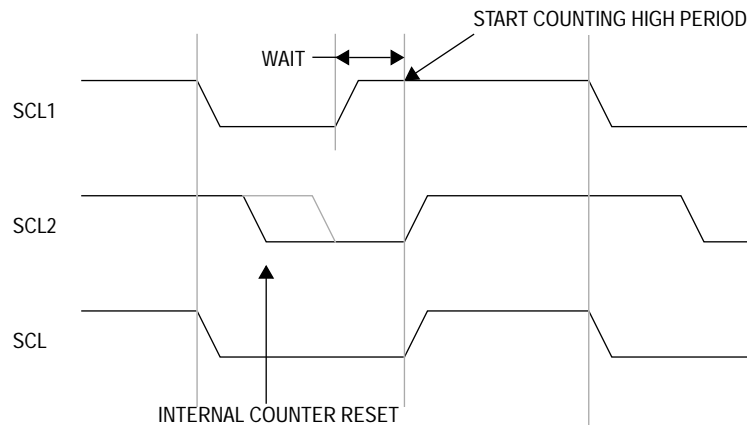
Figure 11-2. Clock Synchronization

11.6 M-Bus Registers

Five different registers are used in the M-bus interface. The internal configuration of these registers is discussed in the following paragraphs.

NOTE: *The register addresses show only the low-order address bits (for example ABL3–ABL0). The registers can be placed anywhere in the device memory map by generating an appropriate module select signal in the map logic.*

A block diagram of the M-bus system is shown in **Figure 11-3**.

11.6.1 M-Bus Address Register

Address: $0018

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	MAD7	MAD6	MAD5	MAD4	MAD3	MAD2	MAD1	
Reset:	0	0	0	0	0	0	0	—

☐ = Unimplemented
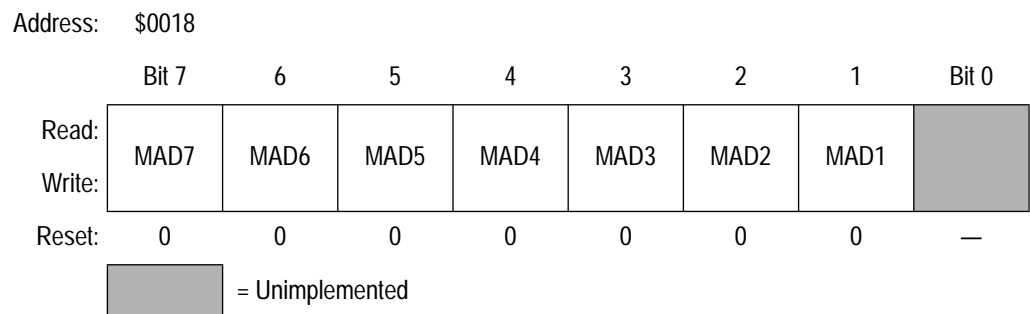
Figure 11-3. M-Bus Address Register (MADR)

Bit 1–Bit 7

Each of these bits contains its own specific slave address. This register is cleared upon reset.
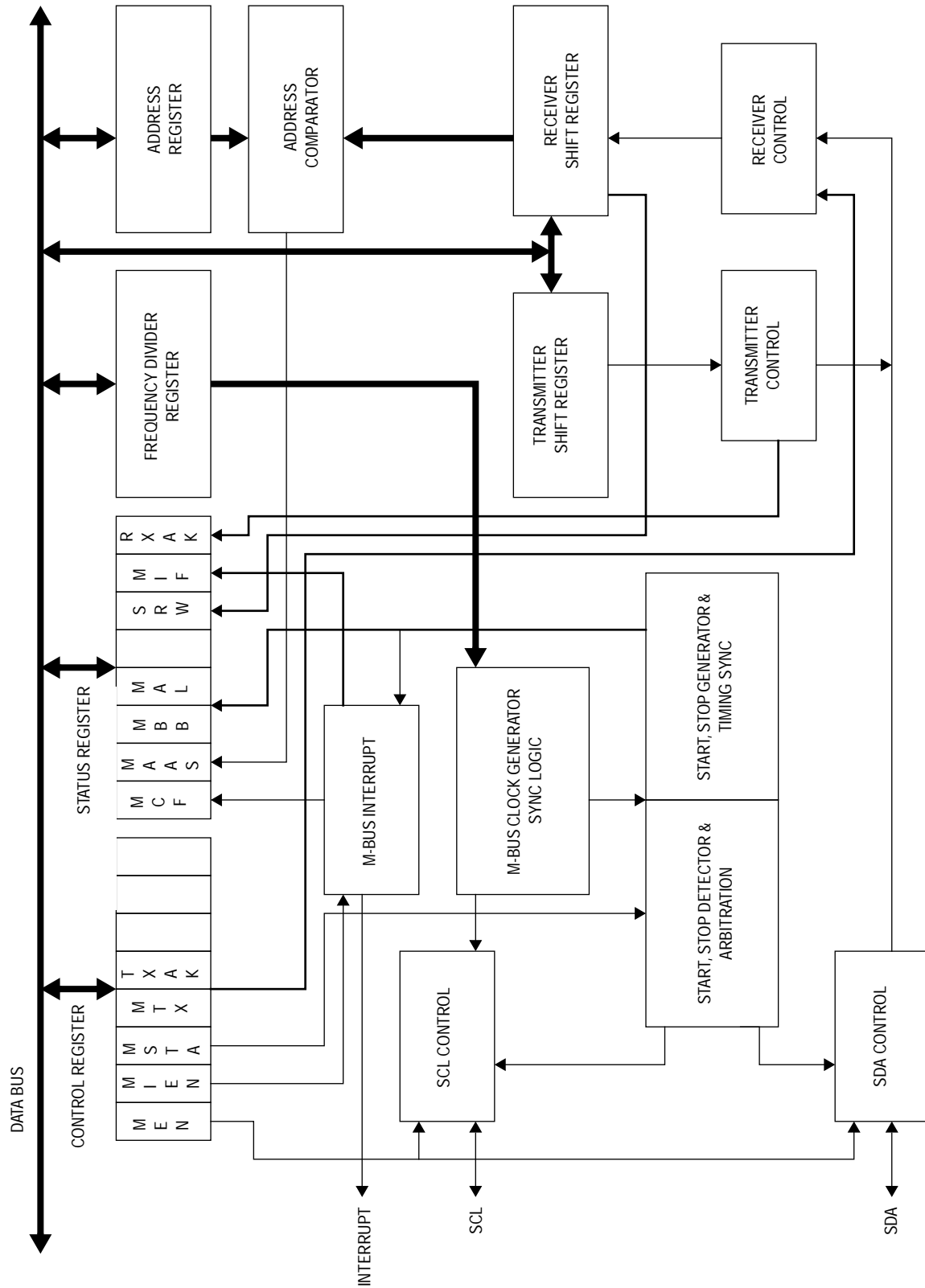
Figure 11-4. M-Bus Interface Block Diagram

11.6.2 M-Bus Frequency Divider Register

Address: $0019

	Bit 7	6	5	4	3	2	1	Bit 0
Read:				FD4	FD3	FD2	FD1	FD0
Write:								
Reset:	—	—	—	0	0	0	0	0

= Unimplemented
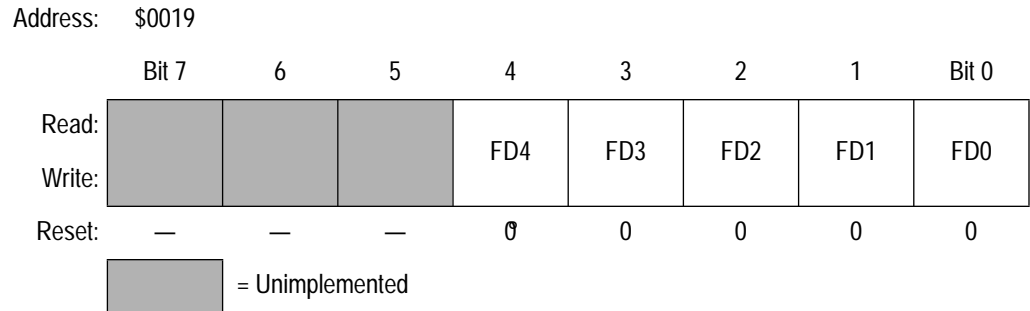
Figure 11-5. M-Bus Frequency Divider Register (MFDR)

Bit 0–Bit 4

These bits are used for clock rate selection. The serial bit clock frequency is equal to the CPU clock divided by the divider shown in **Table 11-1**. This register is cleared upon reset.

For a 4-MHz external crystal operation (2-MHz internal operating frequency), the serial bit clock frequency of the M-bus ranges from 460 Hz to 90,909 Hz.

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

Table 11-1. M-Bus Clock Prescaler

FD4, FD3, FD2, FD1, FD0	Divider	FD4, FD3, FD2, FD1, FD0	Divider
00000	22	10000	352
00001	24	10001	384
00010	28	10010	448
00011	34	10011	544
00100	44	10100	704
00101	48	10101	768
00110	56	10110	896
00111	68	10111	1088
01000	88	11000	1408
01001	96	11001	1536
01010	112	11010	1792
01011	136	11011	2176
01100	176	11100	2816
01101	192	11101	3072
01110	224	11110	3584
01111	272	11111	4352

11.6.3 M-Bus Control Register

The M-bus control register (MCR) provides five control bits and is cleared upon reset.

Address: $001A

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	MEN	MIEN	MSTA	MTX	TXAK	MMUX		
Reset:	0	0	0	0	0	0	—	—

= Unimplemented
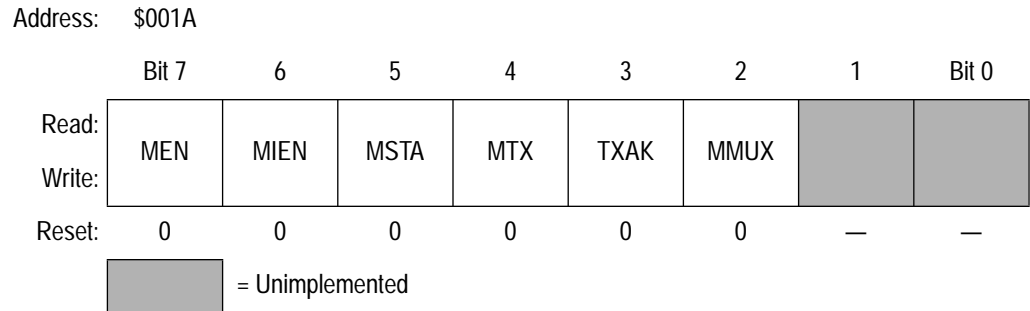
Figure 11-6. M-Bus Control Register (MCR)

MEN — M-Bus Enable Bit

If MEN is set, the M-bus interface system is enabled. If MEN is cleared, the interface is reset and disabled. The MEN bit must be set first before any bits of MCR are set.

MIEN — M-Bus Interrupt Enable Bit

If MIEN is set, an interrupt occurs provided the MIF flag in the status register is set and the I bit in the condition code register is cleared. If MIEN is cleared, the M-bus interrupt is disabled.

MSTA — Master/Slave Mode Select Bit

Upon reset, this bit is cleared. When this bit is changed from a logic 0 to a logic 1, a start signal is generated on the bus, and master mode is selected. When this bit is changed from a logic 1 to a logic 0, a stop signal is generated and the operating mode changes from master to slave.

In master mode, a bit clear immediately followed by a bit set generates a repeated start signal (see **Figure 11-1**) without generating a stop signal.

    1 = Master
    0 = Slave

MTX — Transmit/Receiver Mode Select Bit

This bit selects the direction of master and slave transfers. When addressed as a slave, this bit should be set by software according to the SRW bit in the status register. In master mode, this bit should be set according to the type of transfer required. Hence, for address cycles this bit will always be high.

1 = Transmit
0 = Receive

TXAK — Transmit Acknowledge Enable Bit

If TXAK is cleared, an acknowledge signal will be sent out to the bus at the ninth clock bit after receiving one byte of data. When TXAK is set, there will be no acknowledge signal response (for example, acknowledge bit = 1).

MMUX — M-Bus Multiplexer

This bit is used to enable PB7 and PB6 to be under the control of the M-bus circuit. When set, both PB7 and PB6 become open-collector outputs or inputs when enabled by the M-bus control. When cleared PB7 and PB6 are under control of the port DDR logic. This bit can be set or cleared independent of the MEN bit. Caution should be used if PB7 and PB6 are used as general-purpose I/O.

1 = M-bus control
0 = POR condition, port B DDR control

11.6.4 M-Bus Status Register

This status register is software readable only with exception of bit 1 (MIF) and bit 4 (MAL) which are software clearable. All bits are cleared upon reset except bit 7 (MCF) and bit 0 (RXAK).

Address: $001B

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	MCF	MAAS	MBB	MAL		SRW	MIF	RXAK
Write:				MAL CLR			MIF CLR	
Reset:	1	0	0	0	—	0	0	1
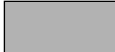
☐ = Unimplemented

Figure 11-7. M-Bus Status Register (MSR)

MCF — Data Transferring Bit

While one byte of data is being transferred, this bit is cleared. It is set by the falling edge of the ninth clock of a byte transfer.
    1 = Transfer complete
    0 = Transfer in progress

MAAS — Addressed as a Slave Bit

When its own specific address (MADR) is matched with the calling address, this bit is set. The CPU is interrupted provided MIEN is set. Then CPU needs to check the SRW bit and set its TX/RX mode accordingly.
    1 = Addressed as a slave
    0 = Not addressed

Writing to the M-bus control register clears this bit.

MBB — Bus Busy Bit

This bit indicates the status of the bus. When a start signal is detected, the MBB is set. If a stop signal is detected, it is cleared.
    1 = Bus busy
    0 = Bus idle

MAL — Arbitration Lost Bit

MAL is set by hardware when the arbitration procedure is lost during a master transmission. This bit must be cleared by software.

SRW — R/$\overline{W}$ Command Bit

When MAAS is set, the R/$\overline{W}$ command bit of the calling address (sent from master) is latched into the R/$\overline{W}$ command bit (SRW). Checking this bit, the CPU can select the slave transmit/receive mode according to the command of master.

   1 = Slave transmit, master reading from slave
   0 = Slave receive, master writing to slave

MIF — M-Bus Interrupt Bit

MIF is set when an interrupt is pending. This will cause an M-bus interrupt request provided MIEN is set. This bit is set when one of the following events occurs:

   – Transmission of one byte is completed. The bit is set at the falling edge of the ninth clock.
   – Reception of a calling address which matches its own specific address in slave receive mode.
   – Arbitration is lost.

This bit must be cleared by writing a logic 0 to it.

RXAK — Receive Acknowledge Bit

If RXAK is low, it indicates an acknowledge signal has been received after the completion of an 8-bit data transmission on the bus. If RXAK is high, it means no acknowledge signal is detected at the ninth clock.

   1 = No acknowledge received
   0 = Acknowledge received

RXAK is set upon reset.

11.6.5 M-Bus Data I/O Register

Address: $001C

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	MD7	MD6	MD5	MD4	MD3	MD2	MD1	MD0

Reset: Unaffected by Reset
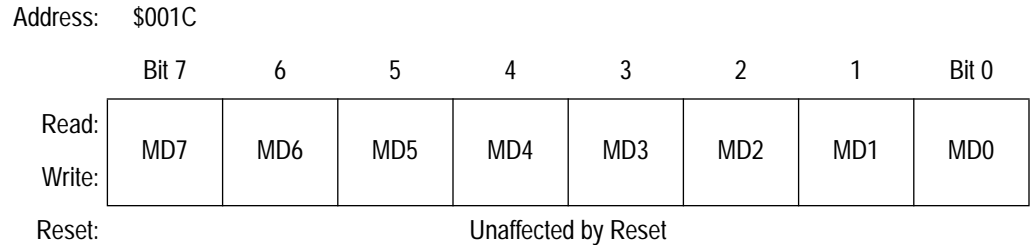
Figure 11-8. M-Bus Data I/O Register (MDR)

In master transmit mode, data written to this register is sent (MSB first) to the bus automatically. In master receive mode, reading from this register initiates reception of the next byte of data. This is accomplished by holding the SCL clock line low until a read of this register occurs. Once the data is read, the device releases the SCL line to allow the transmitting device to transmit the next byte. In slave mode, the same function is available after it is addressed.

Freescale Semiconductor, Inc.

```
                        ┌──────────────┐
                        │  CLEAR MIF   │
                        └──────────────┘
                               │
              Y          ◇ MASTER ◇        N
         ┌───────────────   MODE   ───────────────┐
         │                                        │
    TX ◇ TX/RX ◇ RX                          ◇ ARBITRATION ◇ Y
   ┌─────        ─────┐                  ┌───    LOST    
   │                  │                  │       N
◇ LAST BYTE ◇ Y    ◇ LAST BYTE ◇ Y  ┌─────────┐   ◇ MAAS = 1 ◇ Y
  TRANSMITTED        TO BE READ     │CLEAR MAL│
   N                   N            └─────────┘      N
◇ RXAK=0 ◇ N       ◇ LAST 2nd BYTE ◇   N  ◇ MAAS=1 ◇  ◇ TX/RX ◇ RX
   Y                TO BE READ         Y (READ)  TX
```
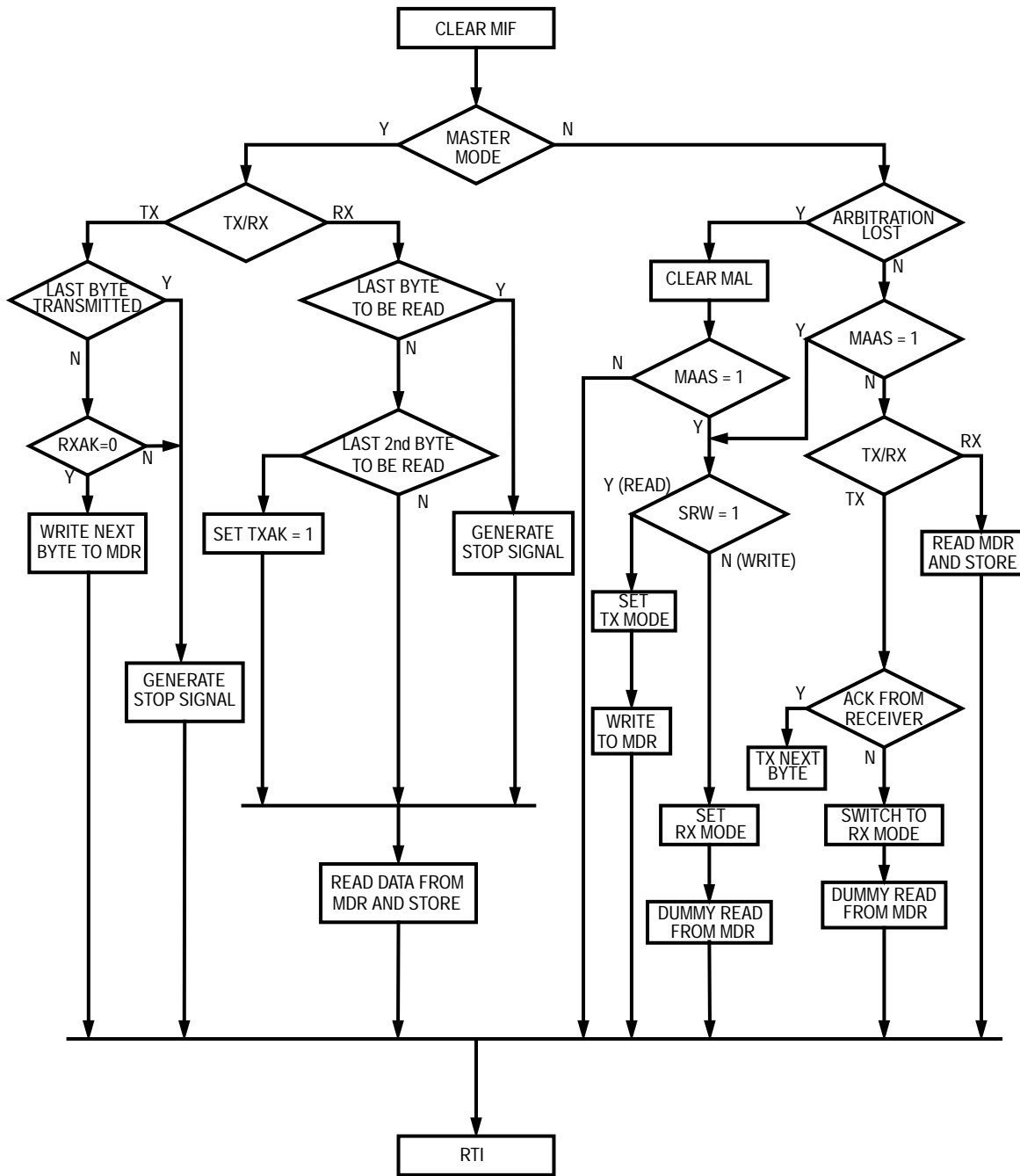
Figure 11-9. Flowchart of M-Bus Interrupt Routine

11.7 M-Bus Pin Configuration

When the M-bus interface is enabled with the MEN bit and the MMUX bit in the M-bus control register (MCR), the port B data direction register bits 6 and 7 relinquish control to the M-bus control register bits. Enabling the M-bus does not alter the state of the port B DDR bits.

11.8 Programming Considerations

Programming considerations are discussed in the following subsections.

11.8.1 Initialization

Initialization is accomplished using the following steps:

1. Update frequency divider register (MFDR) to select an SCL frequency.

2. Update M-bus address register (MADR) to define its own slave address.

3. Set MEN bit of the M-bus control register (MCR) to enable the M-bus interface system and set the MMUX bit to allow M-bus control of the PB7 and PB6 pins.

4. Modify the M-bus control register (MCR) bits to select master/slave mode, transmit/receive mode, interrupt enable, or not.

11.8.2 Generation of a Start Signal and the First Byte of Data Transfer

After completion of the initialization procedure, serial data can be transmitted by selecting the master transmitter mode. If the device is connected to a multimaster bus system, the state of the M-bus busy bit (MBB) must be tested to check whether the serial bus is free. If the bus is free (MBB = 0), the start condition and the first byte (the slave address) can be sent.

An example of a program which generates the start signal and transmits the first byte of data (slave address) is shown here. (The MMUX bit must be set to allow control of PB7 and PB6 pins.)

```
        SEI                     ; DISABLE INTERRUPT
CHFALG  BRSET   5,MSR,CHFLAG; CHECK THE MBB BIT OF THE
                            ; STATUS REGISTER. IF IT IS
                            ; SET, WAIT UNTIL IT IS CLEAR
TXSTART BSET    4,MCR       ; SET TRANSMIT MODE
        BSET    5,MCR       ; SET MASTER MODE
                            ; i.e., GENERATE START CONDITION
        LDA     #CALLING    ; GET THE CALLING ADDRESS
        STA     MDR         ; TRANSMIT THE CALLING
                            ; ADDRESS
        CLI                 ; ENABLE INTERRUPT
```

11.8.3 Software Responses after Transmission or Reception of a Byte

Transmission or reception of a byte will set the data transferring bit (MCF) to a logic 1, which indicates one byte of communication is finished. Also, the M-bus interrupt bit (MIF) is set to generate an M-bus interrupt if the interrupt function is enabled during initialization. Software must clear the MIF bit in the interrupt routine first. The MCF bit will be cleared by reading from the M-bus data I/O register (MDR) in receive mode or writing to MDR in transmit mode. Software may serve the M-bus I/O in the main program by monitoring the MIF bit if the interrupt function is disabled.

The following is an example of a software response by a master transmitter in the interrupt routine. See **Figure 11-9**.

```
ISR        BCLR    1,MSR           ; CLEAR THE MIF FLAG
           BRCLR   5,MCR,SLAVE     ; CHECK THE MSTA FLAG,
                                   ; BRANCH IF SLAVE MODE
           BRCLR   4,MCR,RECEIVE   ; CHECK THE MODE FLAG,
                                   ; BRANCH IF IN RECEIVE MODE
           BRSET   0,MSR,END       ; CHECK ACK FROM RECEIVER
                                   ; IF NO ACK, END OF
                                   ; TRANSMISSION
TRANSMIT LDA       DATABUF         ; GET THE NEXT BYTE OF DATA
           STA     MDR             ; TRANSMIT THE DATA
```

11.8.4 Generation of the Stop Signal

A data transfer ends with a stop signal generated by the master device. A master transmitter can simply generate a stop signal after all the data has been transmitted.

The following is an example showing how a stop condition is generated by a master transmiter.

```
MASTX     BRSET     0,MSR,END    ; IF NO ACK, BRANCH TO END
          LDA       TXCNT        ; GET VALUE FROM THE
                                 ; TRANSMITTING COUNTER
          BEQ       END          ; IF NO MORE DATA, BRANCH TO
                                 ; END
          LDA       DATABUF      ; GET NEXT BYTE OF DATA
          STA       MDR          ; TRANSMIT THE DATA
          DEC       TXCNT        ; DECREASE THE TXCNT
          BRA       EMASTX       ; EXIT
END       BCLR      5,MCR        ; GENERATE A STOP CONDITION
EMASTX    RTI                    ; RETURN FROM INTERRUPT
```

If a master receiver wants to terminate a data transfer, it must inform the slave transmitter by not acknowledging the last byte of data. This can be done by setting the transmit acknowledge bit (TXAK) before reading the second to the last byte of data. Before reading the last byte of data, a stop signal must be generated first.

The following is an example showing how a stop signal is generated by a master receiver.

```
MASR      DEC       RXCNT
          BEQ       ENMASR       ; LAST BYTE TO BE READ
          LDA       RXCNT
          DECA                   ; CHECK LAST 2ND BYTE TO BE READ
          BNE       NXMAR        ; NOT LAST ONE OR LAST SECOND
LAMAR     BSET      3,MCR        ; LAST SECOND, DISABLE ACK
                                 ; TRANSMITTING
          BRA       NXMAR
ENMASR    BCLR      5,MCR        ; LAST ONE, GENERATE 'STOP'
                                 ; SIGNAL
NXMAR     LDA       MDR          ; READ DATA AND STORE
          STA       RXBUF
          RTI
```

11.8.5 Generation of a Repeated Start Signal

If at the end of data transfer the master still wants to communicate on the bus, it can generate another start signal followed by another slave address without first generating a stop signal. A program example is shown here.

```
RESTART  BCLR     5,MCR        ; ANOTHER START (RESTART) IS
         BSET     5,MCR        ; GENERATED BY THESE TWO
                               ; CONSEQUENCE INSTRUCTION
         LDA      #CALLING     ; GET THE CALLING ADDRESS
         STA      MDR          ; TRANSMIT THE CALLING
                               ; ADDRESS
```

11.8.6 Slave Mode

In the slave service routine, the master addressed as slave bit (MAAS) should be tested to see if a calling of its own address has just been received. If MAAS is set, software should set the transmit/receive mode select bit (MTX bit of MCR) according to the R/$\overline{W}$ command bit (SRW). Writing to the MCR clears the MAAS automatically. A data transfer may then be initiated by writing information to MDR or dummy reading from MDR.

In the slave transmitter routine, the received acknowledge bit (RXAK) must be tested before transmitting the next byte of data. If RXAK is set, indicating an end of the data signal from the master receiver, then RXAK must switch from transmitter mode to receiver mode by software. A dummy read must follow to release the SCL line so that the master can generate a stop signal.

11.8.7 Arbitration Lost

If more than one master wants to engage the bus simultaneously, only one master wins and the others lose arbitration. The arbitration loss devices immediately switch to slave receive mode by hardware. Their data output to the SDA line is stopped, but the internal transmitting clock still runs until the end of the current byte transmission. An interrupt occurs when this dummy byte transmission is accomplished with

MAL = 1 and MSTA = 0. If one master attempts to start transmission while the bus is being engaged by another master:

1. The hardware will inhibit the transmission.

2. The MSTA bit will switch from one to zero without generating a stop condition.

3. Interrupt to CPU will be generated.

4. MAL will be set to indicate that the attempt to engage the bus has failed.

In consideration of these cases, the slave service routine should test the MAL first, and software should clear the MAL bit if it is set.

11.9 Operation During Wait Mode

During wait mode the M-bus block is idle. If in slave mode, the M-bus block will wake up on receiving a valid start condition. If the interrupt is enabled, the CPU will come out of wait mode after the end of a byte transmission.

11.10 Operation During Stop Mode

In stop mode, the whole block is disabled.

Freescale Semiconductor, Inc.

Section 12. Synchronous Serial Interface (SSI)

12.1 Contents

Synchronous Serial Interface (SSI)

12.2 Introduction

This synchronous serial I/O module is also used on the MC68HC05X1. The module is similar to the SIOP used on the MC68HC05P7 and the MC68HC705P9 and the SPI used on the MC68HC05L5.

The SSI is a 2-wire master/slave system including serial clock (SCK) and serial data input output (SDIO). Data is transferred eight bits at a time. An interrupt may be generated at the completion of each transfer, and a software programmable option determines whether the SSI transfers data most significant bit (MSB) or least significant bit (LSB) first. When operating as a master device, the serial clock speed is selectable between four rates; as a slave device, the clock speed may be chosen over a wide range. Refer to **Figure 12-1**.

In master mode, transmission is initiated by a write to the SSI data register (SDR). A transfer cannot be initiated in slave mode; however, the external master will initiate the transfer. The programmer must choose between master or slave mode before the SSI is enabled. It is up to the programmer to ensure that only one master exists in the system at any one time. All devices in the system must operate with the same clock polarity and data rates. Slaves should always be disabled before the master is disabled. Likewise, the master should always be enabled before the slaves are enabled.
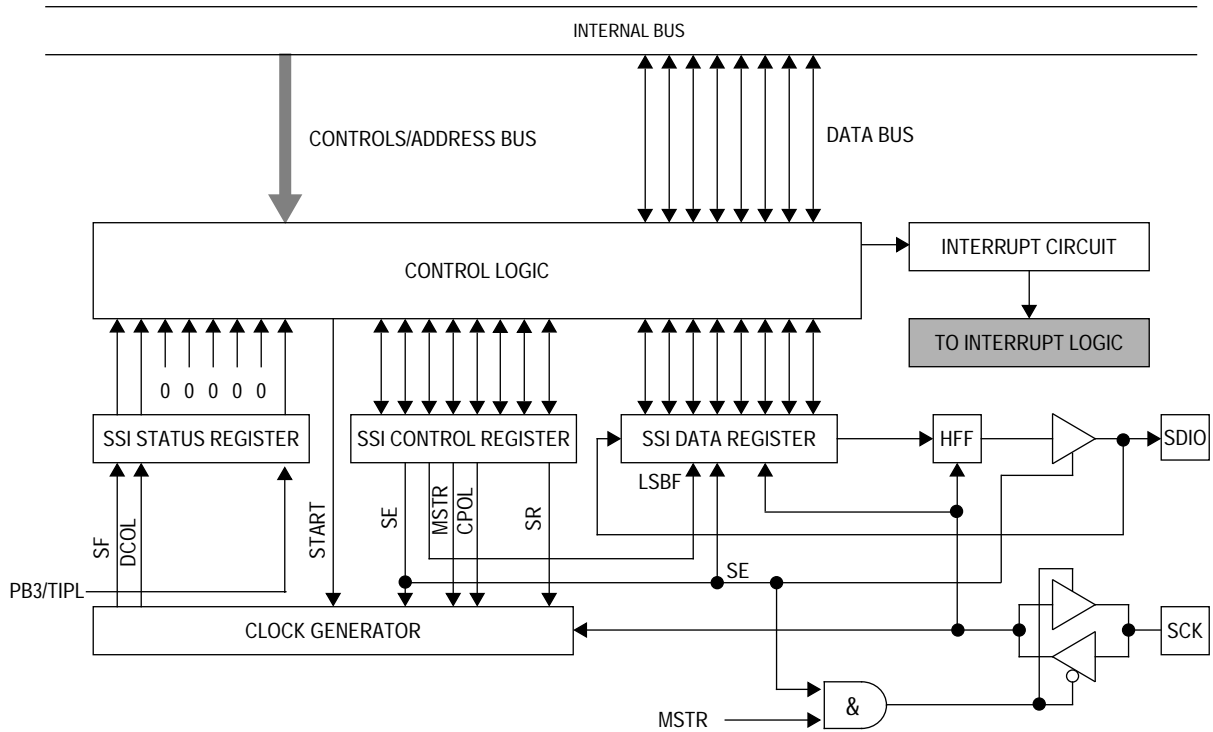
Figure 12-1. SSI Block Diagram

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

12.3 SSI Signals

The following sections describe the SSI signals.

12.3.1 Serial Clock (SCK)

In master mode (MSTR = 1), the SCK pin is an output with a selectable frequency of:

f_{op} divided by 16 (SR1–SR0 = 00),

f_{op} divided by 8 (SR1–SR0 = 01),

f_{op} divided by 4 (SR1–SR0 = 10), or

f_{op} divided by 2 (SR1–SR0 = 11).

This pin will be high (CPOL = 1) or low (CPOL = 0) between transmissions.

In slave mode (MSTR = 0), the SCK pin is an input and the clock must be supplied by an external master with a maximum frequency of f_{op} divided by 2. There is no minimum SCK frequency. This pin should be driven high (CPOL = 1) or low (CPOL = 0) between transmissions by the external master and must be stable before the SSI is first enabled (SE = 1).

NOTE: *Data is always captured with the SDIO pin on the rising edge of SCK. Data is always shifted out and presented at the SDIO pin on the falling edge of SCK.*

12.3.2 Serial Data Input/Output (SDIO)

This pin receives and transmits data to or from the SSI module as described in the following paragraphs.

SDIO as an Output Pin

Prior to enabling the SSI (SE = 0), the SDIO pin will be three-stated. The SDIO pin will be active when the SSI is enabled (SE = 1), the serial direction (SDIR = 1) bit is set, and MSTR = 1. The state of the

pin will depend on the value of the CPOL bit. Data can be sent or received in either MSB first format (LSBF = 0) or LSB first format (LSBF = 1).

If (CPOL = 1), the first falling edge of SCK will shift the first data bit out to the SDIO pin. Subsequent falling edges of SCK will shift the remaining data bits out.

If (CPOL = 0), the first data bit will be driven out to the SDIO pin before the first rising edge of SCK. Subsequent falling edges of SCK will shift the remaining data bits out.

SDIO as an Input Pin

The SDIO pin will accept data once the SSI is enabled and the SDIR bit = 0. Valid data must be present at least 100 ns before the rising edge of the clock and remain valid for 100 ns after the edge. See **Figure 12-2** and **Figure 12-3**.
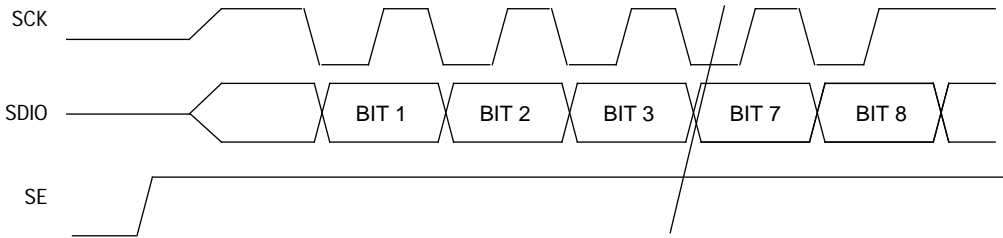
Figure 12-2. Synchronous Serial Interface Timing (CPOL = 1)
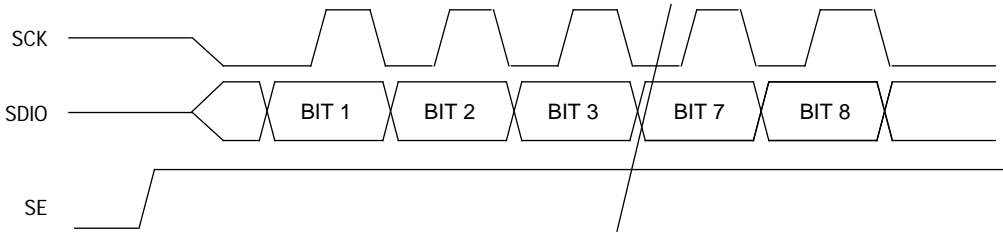
Figure 12-3. Synchronous Serial Interface Timing (CPOL = 0)

12.4 SSI Registers

The SSI registers are described in the following subsections.

12.4.1 SSI Control Register

This register is located at address $000A. A reset clears all of these bits, except bit 3 which is set. Writes to this register during a transfer should be avoided, with the exception of clearing the SE bit to disable the SSI.

In addition, the clock polarity, rate, data format, and master/slave selection should not be changed while the SSI is enabled (SE = 1) or being enabled. Always disable the SSI, by clearing the SE bit, before altering control bits within the SCR.

Address: $000A

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	SIE	SE	LSBF	MSTR	CPOL	SDIR	SR1	SR0
Reset:	0	0	0	0	1	0	0	0

Figure 12-4. SSI Control Register (SCR)

SIE — SSI Interrupt Enable

This bit determines whether an interrupt request should be generated when a transfer is complete. Reset clears this bit.
   1 = An interrupt request will be made if the CPU is in the run or wait mode of operation and the status flag bit SF is set.
   0 = No interrupt requests will be made by the SSI.

SE — SSI Enable

When this bit is set, it enables the SSI and SCK pins. When this bit is cleared, any transmission in progress is aborted and the SCK and SDIO are three-stated. The SE bit is readable and writable any time. Clearing SE while a data transfer is occurring will abort the transmission and reset the bit counter. Reset clears this bit.
   1 = Enable the SSI module.
   0 = Disable the SSI module.

LSBF — Least Significant Bit First

The LSBF bit determines the format of the data transfer. The two formats are least significant bit (LSB) or most significant bit (MSB) transferred or received first. Reset clears this bit, initializing the SSI to MSB first order.

1 = Data will be sent and received in an LSB first format.

0 = Data will be sent and received in an MSB first format.

MSTR — Master Mode

Reset clears this bit and configures the SSI for slave operation. MSTR may be set at any time regardless of the state of SE.

1 = SSI is configured for master mode. The transmission is initiated by a write to the data register and the SCK pin becomes an output providing a synchronous data clock at a rate determined by the SR bit.

0 = SSI is configured to slave mode. Any transmission in progress is aborted. Transfers are initiated by an external master which should supply the clock signal to the SCK pin.

CPOL — Clock Polarity

The clock polarity bit controls the state of the SCK pin between transmissions.

1 = SCK will be high between transmissions.

0 = SCK will be low between transmissions.

In both cases, the data is latched on the rising edge of SCK for serial input and is valid on the rising edge of SCK for serial output. Reset sets this bit.

SDIR — Serial Data Direction

When the SE bit = 1, SDIR functions as the output driver enable bit for the SDIO pin with SSI in master or in slave mode. This bit has no effect on the SDIO pin when the SSI is disabled (SE = 0). This bit is cleared by reset.

1 = Enable the output driver of the SDIO pin.

0 = Disable the output driver of the SDIO pin.

SR1 and SR0 — SSI Clock Rate Select

These bits determine the frequency of SCK when in master mode (MSTR = 1). They have no effect in slave mode (MSTR = 0).

Table 12-1. Master Mode SCK Frequency Select

SR1	SR0	SCK Frequency
0	0	$f_{op} \div 16$
0	1	$f_{op} \div 8$
1	0	$f_{op} \div 4$
1	1	$f_{op} \div 2$

12.4.2 SSI Status Register

The SSI status register (SSR) is located at address $000B and contains three bits.

Address: $000B

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	SF	DCOL	0	0	0	0	0	TIPL
Write:								
Reset:	0	0	0	0	1	0	0	0

☐ = Unimplemented
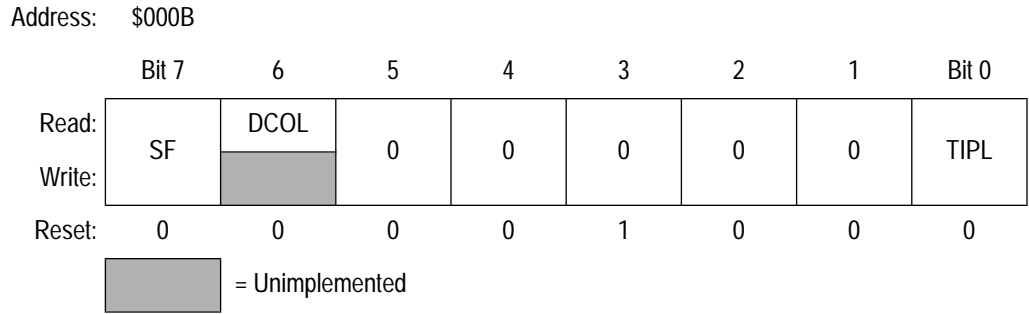
Figure 12-5. SSI Status Register (SSR)

SF — SSI Flag

This bit is set upon occurrence of the last rising clock edge and indicates that a data transfer has taken place. It has no effect on any further transmissions and can be ignored without problem. However, SF must be cleared before a master can initiate a transfer. SF is cleared by reading the SSR with SF set followed by a read or write of the serial data register. If it is cleared before the last edge of the next byte, it will be set again. Reset clears this bit.

DCOL — Data Collision

This is a read-only status bit which indicates that an invalid access to the data register has been made. This can occur any time after the first falling edge of SCK and before SF is set. DCOL is cleared by reading the status register with SF set followed by a read or write of the data register. If the last part of the clearing sequence is done after another transmission has been started, DCOL will be set again. Reset also clears this bit.

TIPL

The state of the PB3 pin is latched and placed into this bit on the eighth rising SCK clock during a shift operation. This is the case regardless of the state of MSTR and CPOL in the SSI control register. Reset clears this bit.

12.4.3 SSI Data Register

This register is located at address $000C and is both the transmit and receive data register. This system is not double buffered but writes to this register during transfers are masked and will not destroy the previous contents. The SDR can be read at any time, but, if a transfer is in progress the results may be ambiguous. This register should only be written to when the SSI is enabled (SE = 1).

Address: $000C

	Bit 7	6	5	4	3	2	1	Bit 0
Read:								
Write:								
Reset:				Reset Results Unknown				
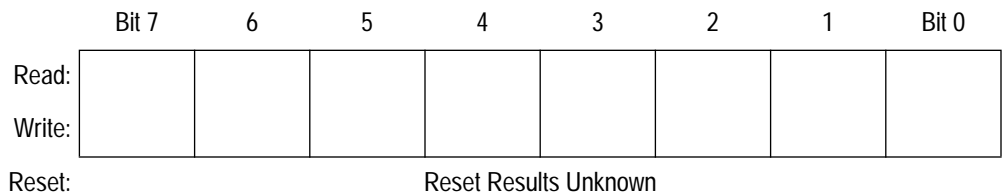
Figure 12-6. SSI Data Register (SDR)

12.5 SSI During Stop Mode

In stop mode, the SSI halts operation. The SDIO and SCK pins will maintain their states.

If the SSI was nearing completion of a transfer when the stop mode is entered, it might be possible for the SSI to generate an interrupt request and cause the processor to immediately exit stop mode. To prevent this occurrence, the programmer should ensure that all transfers are complete before entering stop mode.

If the SSI is configured to slave mode, then further care should be taken in entering stop mode. In slave mode, the SCK pin will still accept a clock from an external master, allowing potentially unwanted transfers to take place and power consumption to be increased. Note that the SSI will not generate interrupt requests in this situation. However, on exiting stop mode through some other means, the SF flag may be found to be set. If, at this point, SIE is also set, an interrupt request will be generated.

NOTE: *To avoid these potential problems, it is safer to disable the SSI completely (SE = 0) before entering stop mode.*

12.6 SSI During Wait Mode

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

12.7 SSI Pin Configuration

When the SSI is enabled via the SE bit of the SCR ($0A), the port B data direction register bits 3–5 relinquish control to the SSI as directed by the combination of the SE, MSTR, and SDIR bits. The states of the port B DDR bits are not altered by the SSI.

Freescale Semiconductor, Inc.

Section 13. Instruction Set

13.1 Contents

13.2 Introduction

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

13.3 Addressing Modes

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

- Inherent

- Immediate

- Direct

- Extended

- Indexed, no offset

- Indexed, 8-bit offset

- Indexed, 16-bit offset

- Relative

13.3.1 Inherent

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

13.3.2 Immediate

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

13.3.3 Direct

Direct instructions can access any of the first 256 memory locations with two bytes. The first byte is the opcode, and the second is the low byte of the operand address. In direct addressing, the CPU automatically uses $00 as the high byte of the operand address.

13.3.4 Extended

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

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

13.3.5 Indexed, No Offset

Indexed instructions with no offset are 1-byte instructions that can access data with variable addresses within the first 256 memory locations. The index register contains the low byte of the effective address of the operand. The CPU automatically uses $00 as the high byte, so these instructions can address locations $0000–$00FF.

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

13.3.6 Indexed, 8-Bit Offset

Indexed, 8-bit offset instructions are 2-byte instructions that can access data with variable addresses within the first 511 memory locations. The CPU adds the unsigned byte in the index register to the unsigned byte following the opcode. The sum is the effective address of the operand. These instructions can access locations $0000–$01FE.

Indexed 8-bit offset instructions are useful for selecting the 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.

13.3.7 Indexed,16-Bit Offset

Indexed, 16-bit offset instructions are 3-byte instructions that can access data with variable addresses at any location in memory. The CPU adds the unsigned byte in the index register to the two unsigned bytes following the opcode. The sum is the effective address of the operand. The first byte after the opcode is the high byte of the 16-bit offset; the second byte is the low byte of the offset.

Indexed, 16-bit offset instructions are useful for selecting the 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.

13.3.8 Relative

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

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

13.4 Instruction Types

The MCU instructions fall into the following five categories:

- Register/Memory Instructions

- Read-Modify-Write Instructions

- Jump/Branch Instructions

- Bit Manipulation Instructions

- Control Instructions

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

13.4.1 Register/Memory Instructions

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

Table 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

13.4.2 Read-Modify-Write Instructions

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

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

Table 13-2. Read-Modify-Write Instructions

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

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

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

13.4.3 Jump/Branch Instructions

Jump instructions allow the CPU to interrupt the normal sequence of the program counter. The unconditional jump instruction (JMP) and the jump-to-subroutine instruction (JSR) have no register operand. Branch instructions allow the CPU to interrupt the normal sequence of the program counter when a test condition is met. If the test condition is not met, the branch is not performed.

The BRCLR and BRSET instructions cause a branch based on the state of any readable bit in the first 256 memory locations. These 3-byte instructions use a combination of direct addressing and relative addressing. The direct address of the byte to be tested is in the byte following the opcode. The third byte is the signed offset byte. The CPU finds the effective branch destination by adding the third byte to the program counter if the specified bit tests true. The bit to be tested and its condition (set or clear) is part of the opcode. The span of branching is from –128 to +127 from the address of the next location after the branch instruction. The CPU also transfers the tested bit to the carry/borrow bit of the condition code register.

Table 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

13.4.4 Bit Manipulation Instructions

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

Table 13-4. Bit Manipulation Instructions

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

13.4.5 Control Instructions

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

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

13.5 Instruction Set Summary

Table 13-6. Instruction Set Summary

Source Form	Operation	Description	Effect on CCR					Address Mode	Opcode	Operand	Cycles
			H	I	N	Z	C				
ADC #opr ADC opr ADC opr ADC opr,X ADC opr,X ADC ,X	Add with Carry	$A \leftarrow (A) + (M) + (C)$	↕×	—	↕×	↕×	↕×	IMM DIR EXT IX2 IX1 IX	A9 B9 C9 D9 E9 F9	ii dd hh ll ee ff ff	2 3 4 5 4 3
ADD #opr ADD opr ADD opr ADD opr,X ADD opr,X ADD ,X	Add without Carry	$A \leftarrow (A) + (M)$	↕×	—	↕×	↕	↕	IMM DIR EXT IX2 IX1 IX	AB BB CB DB EB FB	ii dd hh ll ee ff ff	2 3 4 5 4 3
AND #opr AND opr AND opr AND opr,X AND opr,X AND ,X	Logical AND	$A \leftarrow (A) \wedge (M)$	—	—	↕×	↕	—	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	$PC \leftarrow (PC) + 2 + rel \,?\, C = 0$	—	—	—	—	—	REL	24	rr	3
BCLR n opr	Clear Bit n	$Mn \leftarrow 0$	—	—	—	—	—	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	11 13 15 17 19 1B 1D 1F	dd dd dd dd dd dd dd dd	5 5 5 5 5 5 5 5
BCS rel	Branch if Carry Bit Set (Same as BLO)	$PC \leftarrow (PC) + 2 + rel \,?\, C = 1$	—	—	—	—	—	REL	25	rr	3
BEQ rel	Branch if Equal	$PC \leftarrow (PC) + 2 + rel \,?\, Z = 1$	—	—	—	—	—	REL	27	rr	3
BHCC rel	Branch if Half-Carry Bit Clear	$PC \leftarrow (PC) + 2 + rel \,?\, H = 0$	—	—	—	—	—	REL	28	rr	3
BHCS rel	Branch if Half-Carry Bit Set	$PC \leftarrow (PC) + 2 + rel \,?\, H = 1$	—	—	—	—	—	REL	29	rr	3
BHI rel	Branch if Higher	$PC \leftarrow (PC) + 2 + rel \,?\, C \vee Z = 0$	—	—	—	—	—	REL	22	rr	3
BHS rel	Branch if Higher or Same	$PC \leftarrow (PC) + 2 + rel \,?\, C = 0$	—	—	—	—	—	REL	24	rr	3

Table 13-6. Instruction Set Summary (Continued)

Source Form	Operation	Description	H	I	N	Z	C	Address Mode	Opcode	Operand	Cycles
BIH *rel*	Branch if IRQ Pin High	PC ← (PC) + 2 + *rel* ? IRQ = 1	—	—	—	—	—	REL	2F	rr	3
BIL *rel*	Branch if IRQ Pin Low	PC ← (PC) + 2 + *rel* ? 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	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
BRN *rel*	Branch Never	PC ← (PC) + 2 + *rel* ? 1 = 0	—	—	—	—	—	REL	21	rr	3
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
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

Table 13-6. Instruction Set Summary (Continued)

Source Form	Operation	Description	H	I	N	Z	C	Address Mode	Opcode	Operand	Cycles
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}$) = $FF – (A) X ← ($\overline{X}$) = $FF – (X) M ← ($\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)	—	—	↕×	↕×	↕×	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
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 EC FC	dd hh ll ee ff ff	2 3 4 3 2

Table 13-6. Instruction Set Summary (Continued)
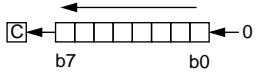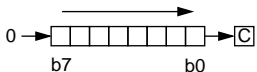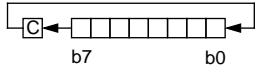
Source Form	Operation	Description	H	I	N	Z	C	Address Mode	Opcode	Operand	Cycles
JSR opr JSR opr JSR opr,X JSR opr,X JSR ,X	Jump to Subroutine	PC ← (PC) + n (n = 1, 2, or 3) Push (PCL); SP ← (SP) − 1 Push (PCH); SP ← (SP) − 1 PC ← Effective Address	—	—	—	—	—	DIR EXT IX2 IX1 IX	BD CD DD ED FD	dd hh ll ee 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
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	dd 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

Table 13-6. Instruction Set Summary (Continued)
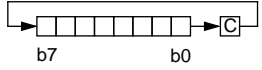
Source Form	Operation	Description	Effect on CCR					Address Mode	Opcode	Operand	Cycles
			H	I	N	Z	C				
ROR *opr* RORA RORX ROR *opr*,X ROR ,X	Rotate Byte Right through Carry Bit	(diagram: b7 → ... → b0 → C)	—	—	↕×	↕	↕	DIR INH INH IX1 IX	36 46 56 66 76	dd ff	5 3 3 6 5
RSP	Reset Stack Pointer	SP ← $00FF	—	—	—	—	—	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		9
RTS	Return from Subroutine	SP ← (SP) + 1; Pull (PCH) SP ← (SP) + 1; Pull (PCL)	—	—	—	—	—	INH	81		6
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
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

Table 13-6. Instruction Set Summary (Continued)

Source Form	Operation	Description	H	I	N	Z	C	Address Mode	Opcode	Operand	Cycles
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		—	0	—	—	—	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

Group legend for columns (MSB / addressing mode):
- 0: Bit Manipulation — DIR
- 1: Bit Manipulation — DIR
- 2: Branch — REL
- 3: Read-Modify-Write — DIR
- 4: Read-Modify-Write — INH
- 5: Read-Modify-Write — INH
- 6: Read-Modify-Write — IX1
- 7: Read-Modify-Write — IX
- 8: Control — INH
- 9: Control — INH
- A: Register/Memory — IMM
- B: Register/Memory — DIR
- C: Register/Memory — EXT
- D: Register/Memory — IX2
- E: Register/Memory — IX1
- F: Register/Memory — IX

Cell format: MNEMONIC MODE cycles/bytes. Blank = unused (shaded).

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

Legend (cell key):

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

Example cell:

MSB / LSB	0
0	BRSET0 5 DIR 3

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

LSB of Opcode in Hexadecimal

Section 14. Electrical Specifications

14.1 Contents

14.2 Introduction

This section contains the electrical and timing specifications.

Electrical Specifications

14.3 Maximum Ratings

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

The MCU contains circuitry to protect the inputs against damage from high static voltages; however, do not apply voltages higher than those shown in the table below. Keep V_{IN} and V_{OUT} within the range $V_{SS} \leq (V_{IN}$ or $V_{OUT}) \leq V_{DD}$. Connect unused inputs to the appropriate voltage level, either V_{SS} or V_{DD}.

Rating	Symbol	Value	Unit
Supply Voltage	V_{DD}	−0.3 to +7.0	V
Input Voltage	V_{IN}	V_{SS} −0.3 to V_{DD} +0.3	V
Self-Check Mode ($\overline{IRQ}$ Pin Only)	V_{IN}	V_{SS} −0.3 to 2 x V_{DD} +0.3	V
Current Drain Per Pin Excluding V_{DD} and V_{SS}	I	25	mA
Storage Temperature Range	T_{STG}	−65 to +150	°C

NOTE: *This device is not guaranteed to operate properly at the maximum ratings. Refer to **14.6 DC Electrical Characteristics** for guaranteed operating conditions.*

14.4 Operating Temperature Range

Characteristic	Symbol	Value	Unit

Operating Temperature Range
  MC68HC(7)05E5DW (Standard)
  MC68HC(7)05E5P
	T_A	T_L to T_H
		0 to +70
		0 to +70
			°C

14.5 Thermal Characteristics

Characteristic	Symbol	Value	Unit

Thermal Resistance
  Plastic DIP
  SOIC
	θ_{JA}	60
		60
			°C/W

I/O Pin Power Dissipation	$P_{I/O}$	User Determined	W
Power Dissipation[1]	P_D	$P_D = (I_{DD} \times V_{DD}) + P_{I/O} =$ $K/(T_J + 273 \,°C)$	W
Constant[2]	K	$P_D \times (T_A + 273 \,°C)$ $+ P_D{}^2 \times \theta_{JA}$	W/°C
Average Junction Temperature	T_J	$T_A + (P_D \times \theta_{JA})$	°C
Maximum Junction Temperature	T_{JM}	125	°C

NOTES:
1. Power dissipation is a function of temperature.
2. K is a constant unique to the device. K can be determined for a known T_A and measured P_D. With this value of K, P_D and T_J can be determined for any value of T_A.

14.6 DC Electrical Characteristics

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_{LOAD} −0.8 mA) PA0–PA7, PB0–PB7, PC0–PC3	V_{OH}	V_{DD} −0.8	—	—	V
Output Low Voltage (I_{LOAD} = 1.6 mA) PA0–PA7, PB0–PB7, PC0–PC3	V_{OL}	—	—	0.4	V
Input High Voltage PA0–PA7, PB0–PB7, PC0–PC3, IRQ, RESET, OSC1	V_{IH}	0.7 x V_{DD}	—	V_{DD}	V
Input Low Voltage PA0–PA7, PB0–PB7, PC0–PC3, IRQ, RESET, OSC1	V_{IL}	V_{SS}	—	0.3 x V_{DD}	V
XFC Wide Bandwidth Source Sink	I_{OH} I_{OL}	−50 50	−100 100	— —	µA
XFC Narrow Bandwidth Source Sink	I_{OH} I_{OL}	−1 1	−2 2	— —	µA
Supply Current (see Notes) Run (f_{osc} = 32.768 kHz, f_{op} = 16.384 kHz) (f_{osc} = 4.2 MHz, f_{op} = 2.1 MHz) Wait (f_{osc} = 32.768 kHz, f_{op} = 16.384 kHz) (f_{osc} = 4.2 MHz, f_{op} = 2.1 MHz) Stop (PLL Off) 25 °C	I_{DD}	 — — — — —	 120 2.5 50 0.7 10	 400 3.5 150 1 50	 µA mA µA mA µA
I/O Ports Hi-Z Leakage Current PA0–PA7, PB0–PB7, PC0–PC3	I_{OZ}	—	—	10	µA
Input Current RESET, IRQ, OSC1	I_{IN}	—	—	1	µA
Capacitance Ports as Input or Output RESET, IRQ	C_{OUT} C_{INT}	— —	— —	12 8	pF

NOTES:
1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = 0 °C to +70 °C, unless otherwise noted
2. Typical values at midpoint of voltage range, 25 °C only
3. Wait I_{DD}: Only timer and CPI systems active
4. Run (Operating) I_{DD}, wait I_{DD}: Measured using external square wave clock source; all inputs 0.2 V from rail; no dc loads; less than 50 pF on all outputs; C_L = 20 pF on OSC2
5. Wait, stop I_{DD}: All ports configured as inputs; V_{IL} = 0.2; V_{IH} = V_{DD} −0.2 V
6. Stop I_{DD} measured with OSC1 = V_{SS}
7. Wait I_{DD} affected linearly by the OSC2 capacitance

Freescale Semiconductor, Inc.
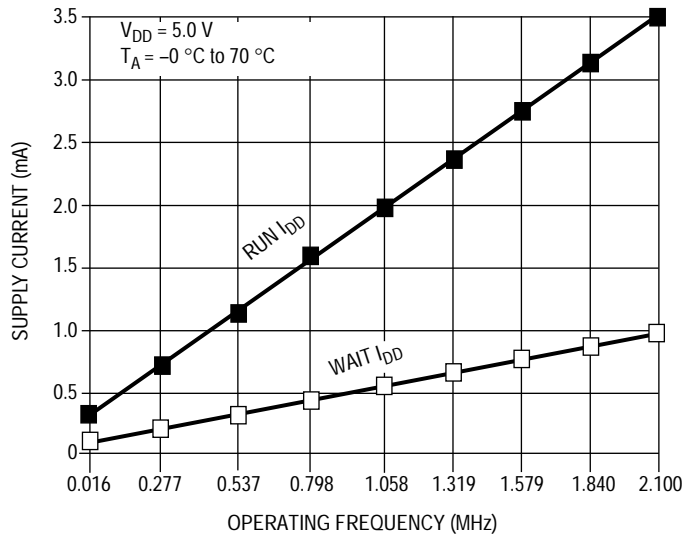
**Figure 14-1. Maximum Supply Current
versus Operating Frequency**
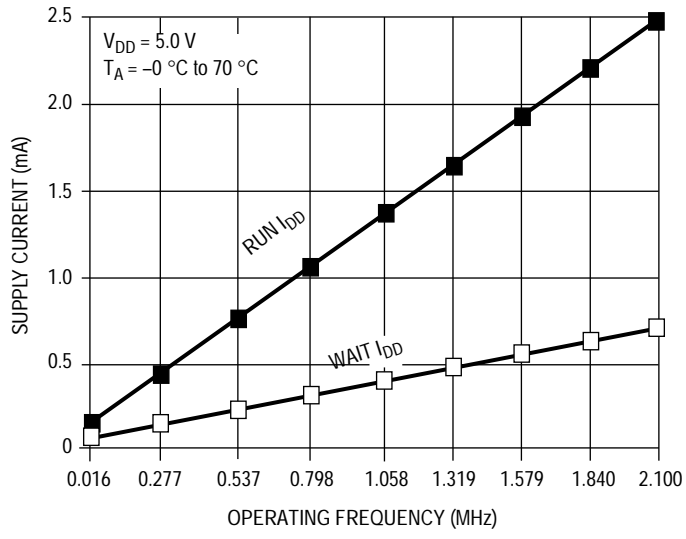
**Figure 14-2. Typical Supply Current
versus Operating Frequency**

14.7 Control Timing

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation Crystal Option External Clock Option	f_{OSC}	— dc	32.768 4.2	kHz MHz
Internal Operating Frequency Crystal ($f_{OSC} \div 2$) External Clock ($f_{OSC} \div 2$)	f_{OP}	— dc	16.384 2.1	kHz MHz
Cycle Time	t_{CYC}	480	—	ns
RESET Pulse Width	t_{RL}	1.5	—	t_{CYC}
Interrupt Pulse Width Low (Edge-Triggered)	t_{ILIH}	125	—	ns
Interrupt Pulse Period	t_{ILIL}	see Note 2	—	t_{CYC}
OSC1 Pulse Width	t_{OH}, t_{OL}	90	—	ns
PLL Startup Stabilization Time	t_{PLLS}	50	—	ms

NOTES:
1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = 0 ºC to +70 ºC, unless otherwise noted
2. The minimum period, t_{ILIL}, should not be less than the number of cycle times it takes to execute the interrupt service routine plus 19 t_{cyc}.
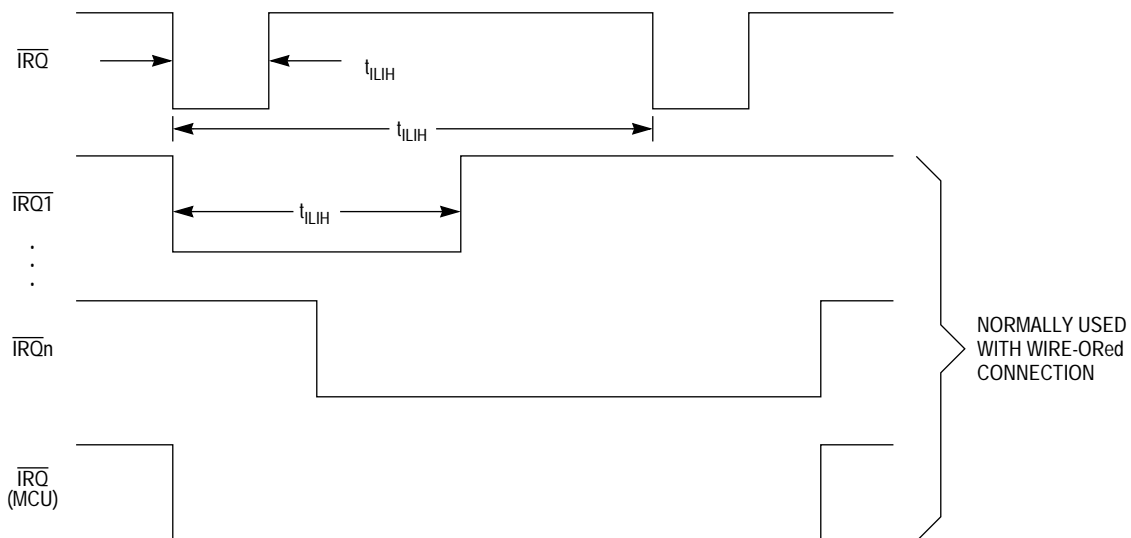
Figure 14-3. External Interrupt Mode Diagram
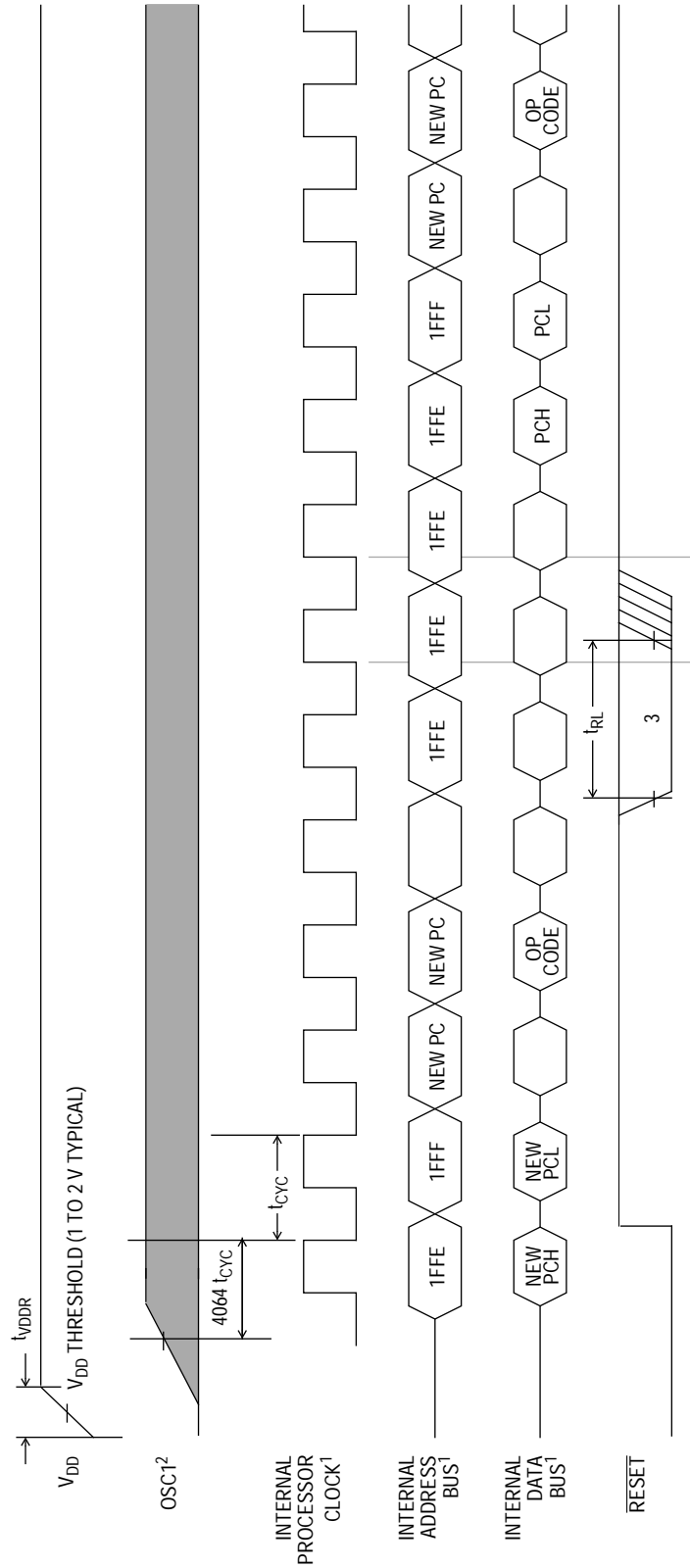
Freescale Semiconductor, Inc.

Figure 14-4. Power-On Reset and RESET

NOTES:
1. Internal timing signal and bus information are not available externally.
2. OSC1 line is not meant to represent frequency. It is only used to represent time.
3. The next rising edge of the internal processor clock following the rising edge of RESET initiates the reset sequence.

14.8 M-Bus Interface Input Signal Timing

Characteristic	Symbol	Min	Max	Unit
Start Condition Hold Time	$t_{HD.STA}$	2	—	t_{cyc}
Clock Low Period	t_{LOW}	4.7	—	t_{cyc}
Clock High Period	t_{HIGH}	4	—	t_{cyc}
SDA/SCL Rise Time	t_R	—	1.0	ms
SDA/SCL Fall Time	t_F	—	300	ns
Data Setup Time	$t_{SU.DAT}$	250	—	ns
Data Hold Time	$t_{HD.DAT}$	0	—	t_{cyc}
Start Condition Setup Time (For Repeated Start Condition Only)	$t_{SU.STA}$	2	—	t_{cyc}
Stop Condition Setup Time	$t_{SU.STO}$	2	—	t_{cyc}

NOTE: V_{DD} = 5.0 Vdc $\pm$ 10%, V_{SS} = 0 Vdc, T_A = −40 ^{o}C to +85 ^{o}C, unless otherwise noted

14.9 M-Bus Interface Output Signal Timing

Characteristic	Symbol	Min	Max	Unit
Start Condition Hold Time	$t_{HD.STA}$	12	—	t_{cyc}
Clock Low Period	t_{LOW}	11	—	t_{cyc}
Clock High Period	t_{HIGH}	11	—	t_{cyc}
SDA/SCL Rise Time	t_R	—	1.0	ms
SDA/SCL Fall Time	t_F	—	300	ns
Data Setup Time	$t_{SU.DAT}$	$t_{Low}-t_{cyc}$	—	ns
Data Hold Time	$t_{HD.DAT}$	0	—	t_{cyc}
Start Condition Setup Time (For Repeated Start Condition Only)	$t_{SU.STA}$	10	—	t_{cyc}
Stop Condition Setup Time	$t_{SU.STO}$	12	—	t_{cyc}

NOTE: V_{DD} = 5.0 Vdc $\pm$ 10%, V_{SS} = 0 Vdc, T_A = −40 ^{o}C to +85 ^{o}C, unless otherwise noted
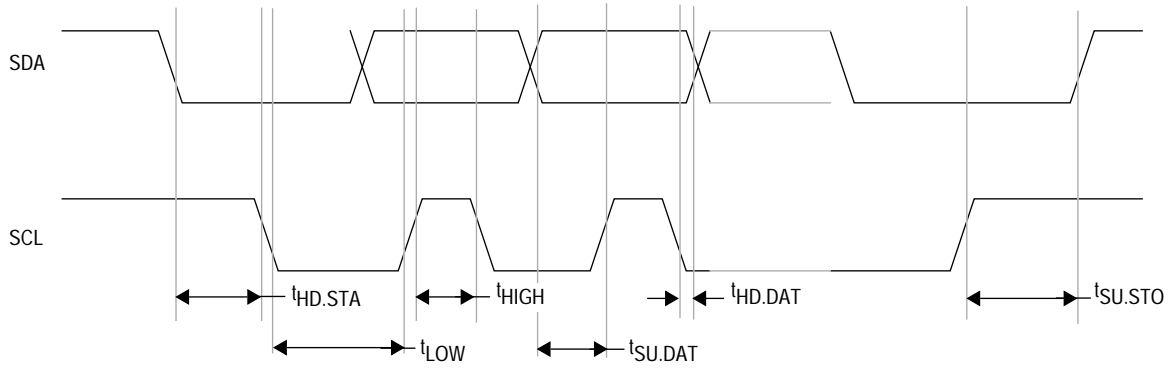
Figure 14-5. M-Bus Interface Timing

Freescale Semiconductor, Inc.

General Release Specification — MC68HC05E5

Section 15. Mechanical Data

15.1 Contents

15.2 Introduction

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

15.3 28-Pin Plastic Dual-in-Line Package (Case 710-02)
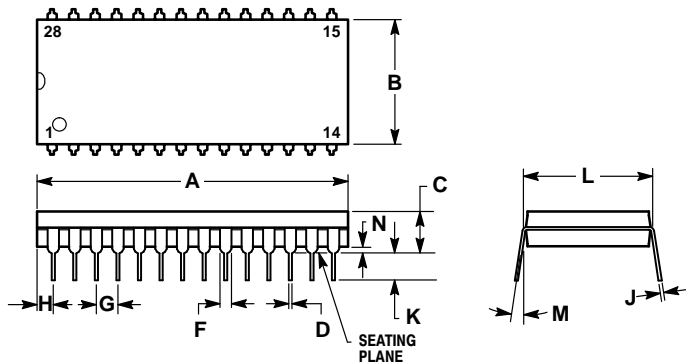
NOTES:
1. POSITIONAL TOLERANCE OF LEADS (D), SHALL BE WITHIN 0.25mm (0.010) AT MAXIMUM MATERIAL CONDITION, IN RELATION TO SEATING PLANE AND EACH OTHER.
2. DIMENSION L TO CENTER OF LEADS WHEN FORMED PARALLEL.
3. DIMENSION B DOES NOT INCLUDE MOLD FLASH.

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

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

15.4 28-Pin Small Outline Integrated Circuit Package (Case 751F-04)
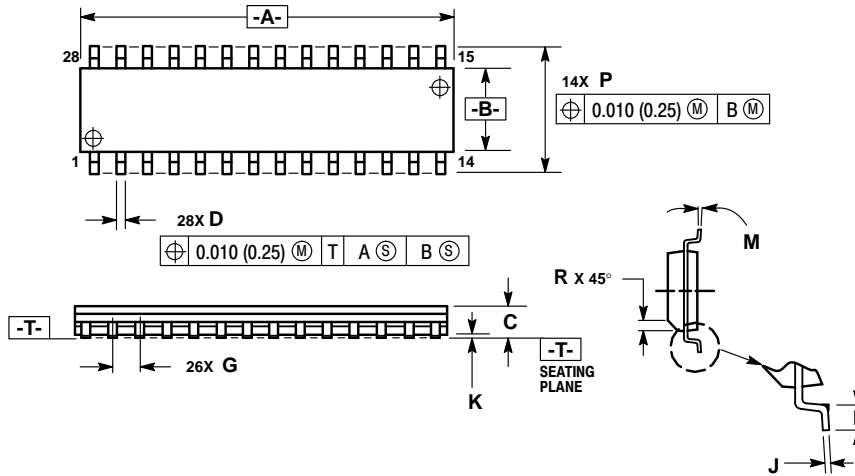
NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: MILLIMETER.
3. DIMENSION A AND B DO NOT INCLUDE MOLD PROTRUSION.
4. MAXIMUM MOLD PROTRUSION 0.15 (0.006) PER SIDE.
5. DIMENSION D DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL BE 0.13 (0.005) TOTAL IN EXCESS OF D DIMENSION AT MAXIMUM MATERIAL CONDITION.

DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	17.80	18.05	0.701	0.711
B	7.40	7.60	0.292	0.299
C	2.35	2.65	0.093	0.104
D	0.35	0.49	0.014	0.019
F	0.41	0.90	0.016	0.035
G	1.27 BSC		0.050 BSC	
J	0.23	0.32	0.009	0.013
K	0.13	0.29	0.005	0.011
M	0°	8°	0°	8°
P	10.05	10.55	0.395	0.415
R	0.25	0.75	0.010	0.029

General Release Specification

MC68HC05E5 — Rev. 1.0

Section 16. Ordering Information

16.1 Contents

16.2 Introduction

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

16.3 MCU Ordering Forms

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

- A current MCU ordering form that is **completely filled out** (Contact your Motorola sales office for assistance.)

- A copy of the customer specification if the customer specification deviates from the Motorola specification for the MCU

- Customer's application program on one of the media listed in **16.4 Application Program Media**

Ordering Information

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

16.4 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 800 K or double-sided high-density 1.4 M)

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

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

Use positive logic for data and addresses.

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

- Customer name

- Customer part number

- Project or product name

- File name of object code

- Date

- Name of operating system that formatted diskette

- Formatted capacity of diskette

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

---

1. Macintosh is a registered trademark of Apple Computer, Inc.
2. MS-DOS is a registered trademark of Microsoft Corporation.
3. PC-DOS is a trademark of International Business Machines Corporation.

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

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

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

16.5 ROM Program Verification

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

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

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

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

Freescale Semiconductor, Inc.

16.6 ROM Verification Units (RVUs)

After receiving the signed listing verify form, Motorola manufactures a custom photographic mask. The mask contains the customer's application program and is used to process silicon wafers. The application program cannot be changed after the manufacture of the mask begins. Motorola then produces 10 MCUs, called RVUs, and sends the RVUs to the customer. RVUs are usually packaged in unmarked ceramic and tested to 5 Vdc at room temperature. RVUs are not tested to environmental extremes because their sole purpose is to demonstrate that the customer's user ROM pattern was properly implemented. The 10 RVUs are free of charge with the minimum order quantity. These units are not to be used for qualification or production. RVUs are not guaranteed by Motorola Quality Assurance.

16.7 MC Order Numbers

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

Table 16-1. MC Order Numbers

Package Type	Operating Temperature Range	MC Order Number
28-Pin Plastic Dual In-Line Package (PDIP)	0 °C to 70°C	MC68HC05E5P
28-Pin Small Outline Integrated Circuit Package (SOIC)	0 °C to 70°C	MC68HC05E5DW

Freescale Semiconductor, Inc.

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