Order this document by
MC68HC05K3/D
Rev. 4.0

HC05

MC68HC05K3

HCMOS Microcontroller Unit

TECHNICAL DATA

MOTOROLA

List of Sections

List of Sections

Table of Contents

Section 1. General Description

Section 2. Memory Map

Section 3. Central Processor Unit (CPU) Core

Section 4. Interrupts

Section 5. Resets

Section 6. Operational Modes

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

Section 8. 8-Bit Timer

Section 9. Personality EEPROM (PEEPROM)

Section 10. Instruction Set

Section 11. Electrical Specifications

Section 12. Mechanical Specifications

Section 13. Ordering Information

List of Figures

List of Figures

List of Tables

List of Tables

Section 1. General Description

1.1 Contents

1.2 Introduction

The low-cost MC68HC05K3 microcontroller (MCU) is a member of the M68HC05 Family of microprocessors. This device has 64 bytes of user RAM, 128 bits of personality electronically erasable programmable ROM (PEEPROM), and 928 bytes of user ROM. This device is available in the 16-pin plastic dual in-line package (PDIP), 16-pin small outline integrated circuit (SOIC) package, and 20-pin super small outline

(SSOP) package. A functional block diagram of the MC68HC05K3 is shown in **Figure 1-4**.

1.3 Features

- Low-cost HC05 core
- 16-pin PDIP, 16-pin SOIC package, or 20-pin SSOP
- 928 bytes of user ROM, including eight bytes of user vectors
- 64 bytes of user RAM
- Low-power operation at 1.8 V — V_{DD} minimum (EEPROM read only)
- 128 bits of personality EEPROM (not memory mapped) programmed using CPU software or with on-chip serial programming ROM
- On-chip charge pump for in-circuit programming of the personality EEPROM at 2.7 to 5.5 Vdc
- 8-bit free-running timer
- 4-stage selectable real-time interrupt generator
- 10 bidirectional input/output (I/O) lines including:
  - 8-mA sink capability on four I/O pins (PA7–PA4)
  - Mask option for software programmable pulldowns on all I/O pins
  - Mask option for port interrupts on four I/O pins (PA3–PA0) (keyboard scan feature)
- IRQ interrupt hardware mask, flag bit, and request bit
- Mask option for sensitivity on IRQ interrupt (edge- and level-sensitive or edge-sensitive only)
- On-chip oscillator (mask options for crystal/ceramic resonator oscillator with internal 2-MΩ resistor and 2-pin or 3-pin resistor capacitor (RC) oscillator)

- Mask option for reduced startup delay time with RC oscillator options
- Mask option for computer operating properly (COP) watchdog system
- Power-saving stop mode and wait mode instructions
- Mask option to convert STOP instruction to halt mode
- Illegal address reset
- Internal steering diode and pullup resistor on $\overline{\text{RESET}}$ pin to V_{DD}
- Internal $\overline{\text{RESET}}$ pin pulldown from COP watchdog and ILADR

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

Any reference to voltage, current, or frequency specified in the following sections refers to the nominal values. The exact values and their tolerance or limits are specified in ***Section 11. Electrical Specifications****.*

1.4 Mask Options

The MC68HC05K3 contains these eight mask options:

1. COP watchdog timer (enable or disable)
2. IRQ triggering (edge-sensitive or edge- and level-sensitive)
3. Port A interrupts (enable or disable)
4. Port software programmable pulldowns (enable or disable)
5. STOP instruction (enable or disable)
6. Oscillator type (crystal/ceramic resonator or RC)
7. RC oscillator type (2-pin or 3-pin)
8. RC oscillator startup delay (4064 or 16 f_{OP} cycles)

NOTE: *The startup delay of 16 f_{OP} cycles and the crystal/ceramic resonator oscillator should not be selected together.*

1.5 Pin Assignments

The MC68HC05K3 is available in 16-pin PDIP, 16-pin SOIC, and 20-pin SSOP packages. The pin assignments for these packages are shown in Figure 1-1, Figure 1-2, and Figure 1-3.

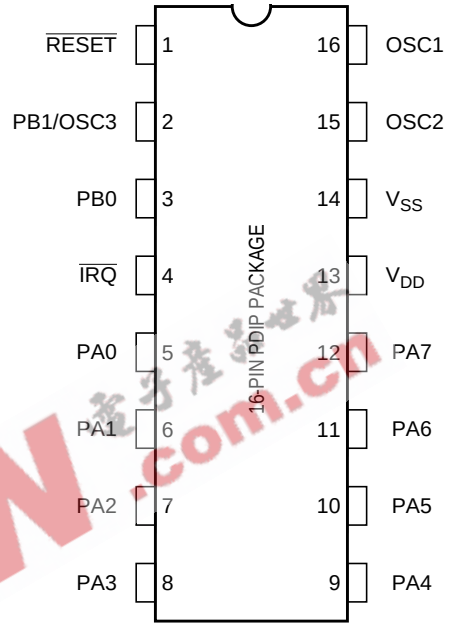


Figure 1-1. Pin Assignments for 16-Pin PDIP

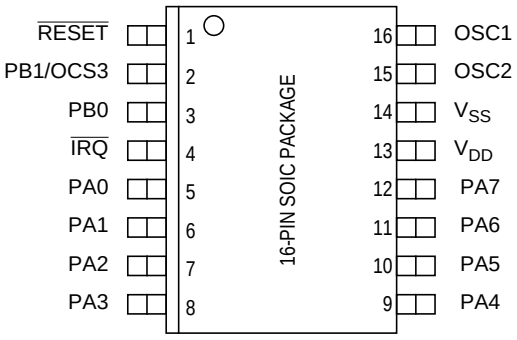


Figure 1-2. Pin Assignments for 16-Pin SOIC

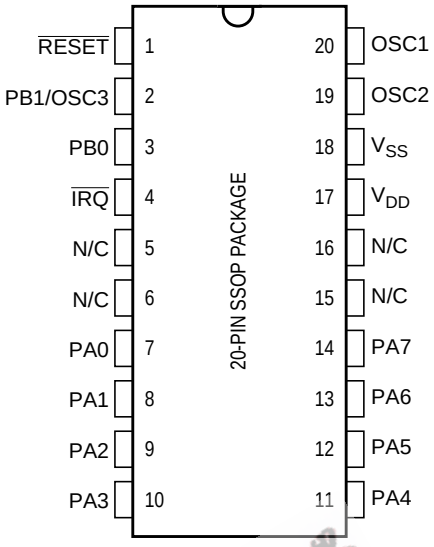


Figure 1-3. Pin Assignments for 20-Pin SSOP

1.6 MCU Structure

The overall block diagram of the MC68HC05K3 is shown in **Figure 1-4**.

1.7 Functional Pin Description

The following paragraphs give a description of the general function of each pin.

1.7.1 V_{DD} and V_{SS}

Power is supplied to the MCU through V_{DD} and V_{SS}. V_{DD} is the positive supply and V_{SS} is ground. The MCU operates from a single power supply.

Rapid signal transitions occur on the MCU pins. The short rise and fall times place very high short-duration current demands on the power supply. To prevent noise problems, special care should be taken to provide good power supply bypassing at the MCU by using bypass capacitors with high-frequency characteristics that are positioned as close to the MCU as possible.

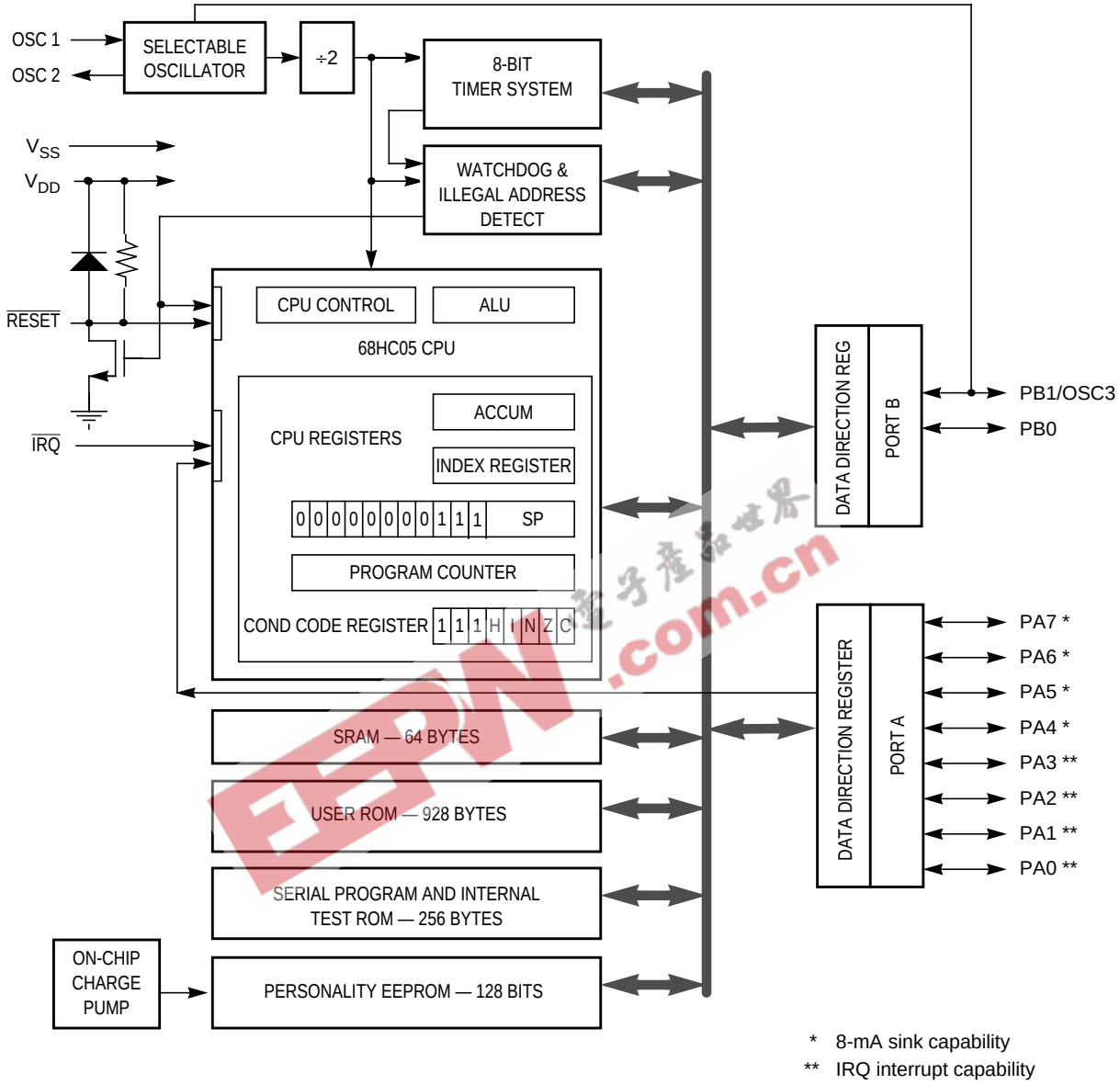


Figure 1-4. MC68HC05K3 Block Diagram

1.7.2 OSC1 and OSC2

The OSC1 and OSC2 pins are the connections for the 2-pin on-chip oscillator. The OSC1 and OSC2 pins also can be used in conjunction with the PB1/OSC3 pin to create a more stable 3-pin RC oscillator.

The OSC1, OSC2, and PB1/OSC3 pins can accept these sets of components:

1. A crystal, as shown in **Figure 1-5(a)**
2. A ceramic resonator, as shown in **Figure 1-5(a)**
3. An external resistor and capacitor using two pins, as shown in **Figure 1-5(b)**
4. An external resistor and capacitor using three pins, as shown in **Figure 1-5(c)**
5. An external clock signal, as shown in **Figure 1-5(d)**

The frequency, f_{OSC}, of the oscillator or external clock source is divided by two to produce the internal operating frequency, f_{OP}. The oscillator type is selected by two mask options.

1.7.2.1 2-Pin Crystal Oscillator

The circuit in **Figure 1-5(a)** shows a typical 2-pin oscillator circuit for an AT-cut, parallel resonant crystal. The crystal manufacturer's recommendations should be followed, since the crystal parameters determine the external component values required to provide maximum stability and reliable startup. The load capacitance values used in the oscillator circuit design should include all stray capacitances. The crystal and components should be mounted as close as possible to the pins for startup stabilization and to minimize output distortion. An internal startup resistor of approximately 2 MΩ is provided between OSC1 and OSC2 when the crystal/ceramic resonator oscillator option is used.

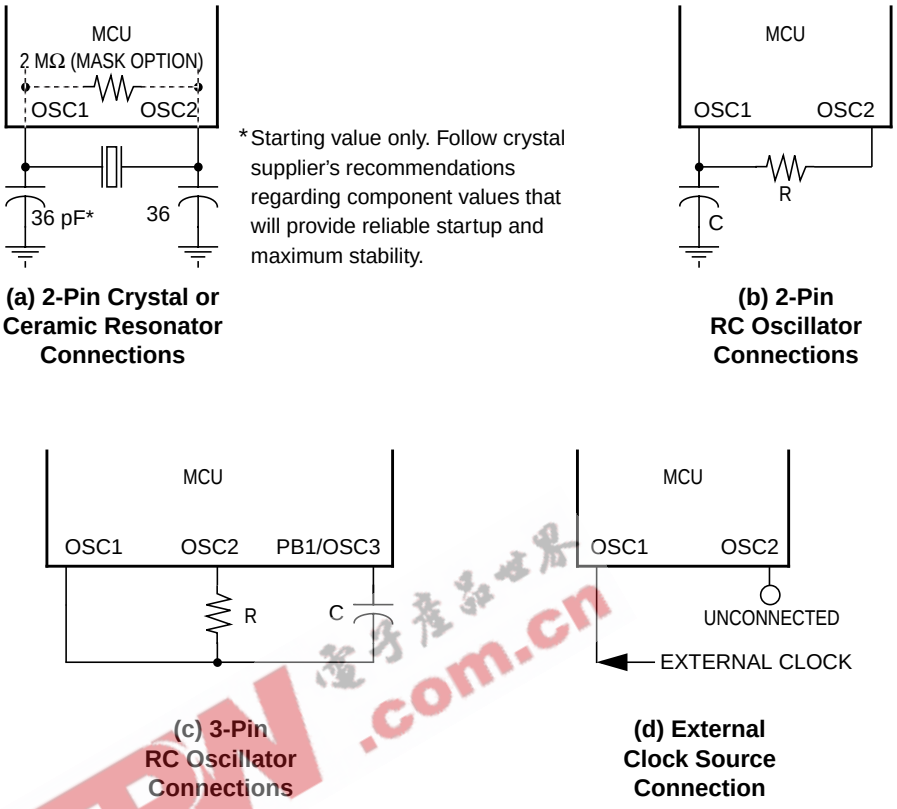


Figure 1-5. Oscillator Connections

1.7.2.2 2-Pin Ceramic Resonator Oscillator

In cost-sensitive applications, a ceramic resonator can be used instead of the crystal. The circuit in **Figure 1-5(a)** is designed for either a crystal or a ceramic resonator. The resonator manufacturer's recommendations should be followed, since the resonator parameters determine the external component values required for maximum stability and reliable starting. The load capacitance values used in the oscillator circuit design should include all stray capacitances. The ceramic resonator and components should be mounted as close as possible to the pins for startup stabilization and to minimize output distortion. An internal startup resistor of approximately 2 MΩ is provided between OSC1 and OSC2 for the crystal/ceramic resonator oscillator mask option.

1.7.2.3 2-Pin RC Oscillators

The 2-pin RC oscillator configuration can be used for very low-cost applications. With this option, a resistor must be connected between the two oscillator pins and a capacitor must be connected from the OSC1 pin to V_{SS}, as shown in **Figure 1-5(b)**. The signal on the OSC2 pin is a square wave and the signal on the OSC1 pin approximates a triangular wave.

The 2-pin RC oscillator is optimized for operation at 500 kHz. This oscillator can be used at higher or lower frequencies with degraded accuracy over temperature, supply voltage, and/or device processing variations. The internal startup resistor of approximately 2 MΩ is **not** connected between OSC1 and OSC2 when the 2-pin RC oscillator mask option is selected.

1.7.2.4 3-Pin RC Oscillator

Another low cost, but more accurate, type of RC oscillator is the 3-pin configuration utilizing the PB1/OSC3 pin. With this option, a resistor must be connected between the OSC1 and OSC2 pins and a capacitor must be connected between the OSC1 and PB1/OSC3 pins, as shown in **Figure 1-5(c)**. This 3-pin RC oscillator is more accurate than the 2-pin RC oscillator with respect to temperature, supply voltage, and/or device processing variations. The signal on the OSC2 and PB1/OSC3 pins is a square wave and the signal on the OSC1 pin approximates a triangular wave.

The 3-pin RC oscillator is optimized for operation at 500 kHz. This oscillator can be used at higher or lower frequencies with degraded accuracy over temperature, supply voltage, and/or device processing variations. The internal startup resistor of approximately 2 MΩ is **not** connected between OSC1 and OSC2 when the 3-pin RC oscillator mask option is selected. The typical external components for a 500-kHz oscillator are a 20-kΩ resistor and a 25- to 30-pF capacitor.

NOTE: *Capacitors used with the RC oscillators should have minimal leakage. Electrolytic or tantalum capacitors should not be used because they degrade the temperature performance of the oscillator due to excessive variation in their leakage.*

1.7.2.5 External Clock

An external clock from another CMOS-compatible device can be connected to the OSC1 input, with the OSC2 input not connected, as shown in **Figure 1-5(d)**. This configuration is possible regardless of whether the oscillator is set up for crystal/ceramic resonator, 2-pin RC, or 3-pin RC operation. However, if the 3-pin RC oscillator is selected, the PB1/OSC3 pin also must be left unconnected.

1.7.3 Reset ($\overline{\text{RESET}}$)

This pin can be used as an input to reset the MCU to a known startup state by pulling the pin to the low state. The $\overline{\text{RESET}}$ pin contains a steering diode to discharge any voltage on the pin to V_{DD} when the power is removed. The $\overline{\text{RESET}}$ pin contains an internal pullup resistor to V_{DD} of approximately 100 kΩ to allow the $\overline{\text{RESET}}$ pin to be left unconnected for low-power applications. The $\overline{\text{RESET}}$ pin contains an internal Schmitt trigger to improve its noise immunity as an input.

The $\overline{\text{RESET}}$ pin has an internal pulldown device that pulls the $\overline{\text{RESET}}$ pin low when there is an internal COP watchdog or an illegal address reset. Refer to **Section 5. Resets**.

1.7.4 Maskable Interrupt Request ($\overline{\text{IRQ}}$)

The $\overline{\text{IRQ}}$ input pin drives the asynchronous IRQ interrupt function of the CPU. The IRQ interrupt function has a mask option to select either negative edge-sensitive triggering or both negative edge-sensitive and low level-sensitive triggering. If the option is selected to include level-sensitive triggering, the $\overline{\text{IRQ}}$ pin requires an external resistor to V_{DD} if “wired-OR” operation is desired. If the $\overline{\text{IRQ}}$ pin is not used, it must be tied to the V_{DD} supply.

NOTE: *Each of the PA0–PA3 I/O pins can be connected through an OR gate to the IRQ interrupt function by a common mask option. This capability allows keyboard scan applications where the transitions or levels on the I/O pins behave the same as the $\overline{\text{IRQ}}$ pin, except that the logic level is*

inverted. The edge or level sensitivity selected by the mask option for the $\overline{IRQ}$ pin also applies to the I/O pins ORed to create an IRQ signal.

The $\overline{IRQ}$ pin contains an internal Schmitt trigger to improve noise immunity. For more details, see **Section 4. Interrupts**.

1.7.5 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 inputs during power-on or reset, except in serial program mode. The four upper-order I/O pins (PA4–PA7) are capable of sinking higher currents. The four lower-order I/O pins (PA0–PA3) can be connected via an internal OR gate to the IRQ interrupt function by a mask option. All the port A pins can have software programmable pulldown devices provided by another mask option. See **Section 7. Parallel Input/Output (I/O)** for more details on the I/O ports.

1.7.6 PB0

The state of the PB0 pin is software programmable and is configured as an input during power-on or reset, except in serial program mode. This pin can have a software programmable pulldown device provided by a mask option. See **Section 7. Parallel Input/Output (I/O)** for more details on the I/O ports.

1.7.7 PB1/OSC3

The state of the PB1/OSC3 pin is software programmable and is configured as an input during power-on or reset, except in serial program mode or when the 3-pin RC oscillator configuration is selected by mask option. This pin can have a software programmable pulldown device provided by a mask option. See **Section 7. Parallel Input/Output (I/O)** for more details on the I/O ports.

General Description

Section 2. Memory Map

2.1 Contents

2.2 Introduction

The MC68HC05K3 has several input/output (I/O) features, 64 bytes of user random-access memory (RAM), 128 bits of user personality electronically erasable programmable read-only memory (PEEPROM), and 928 bytes of user read-only memory (ROM), which are all active in the single-chip mode as shown in **Figure 2-1**.

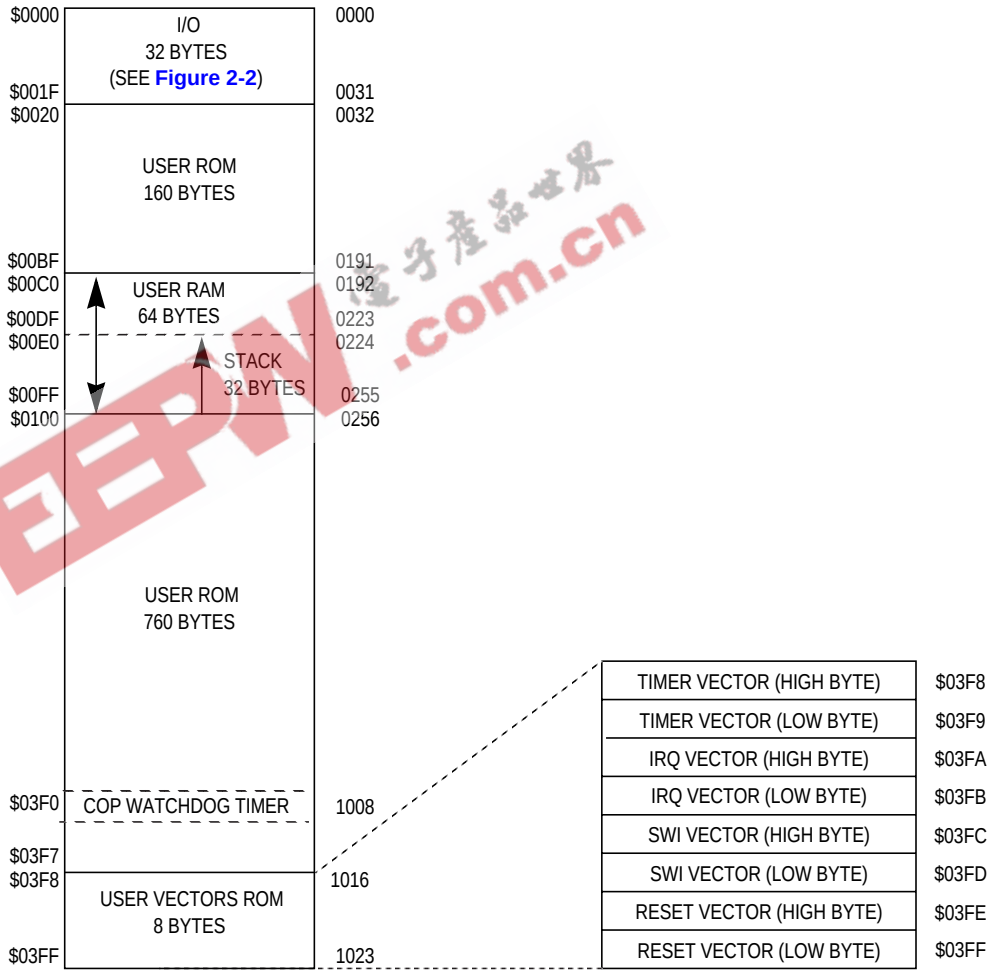


Figure 2-1. MC68HC05K3 Single-Chip Mode Memory Map

2.3 Input/Output and Control Registers

The input/output (I/O) and status/control registers reside in locations $0000–$001F. The overall organization of these registers is shown in Figure 2-2.

The bit assignments for each register are shown in Figure 2-3. Reading unimplemented bits returns unknown states, and writing to unimplemented bits has no effect.

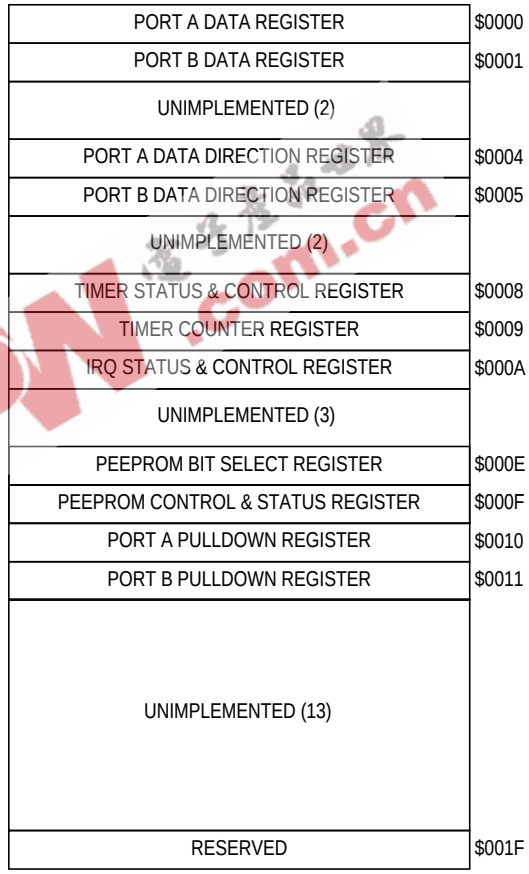


Figure 2-2. MC68HC05K3 I/O Registers Memory Map

Addr.	Name		Bit 7	6	5	4	3	2	1	Bit 0
$0000	Port A Data (PORTA)	Read: / Write:	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
		Reset:	Unaffected by reset							
$0001	Port B Data (PORTB)	Read:	0	0	0	0	0	0	PB1	PB0
		Write:								
		Reset:	Unaffected by reset							
$0002	Unimplemented	Read: / Write:								
$0003	Unimplemented	Read: / Write:								
$0004	Port A Data Direction (DDRA)	Read: / Write:	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
		Reset:	0	0	0	0	0	0	0	0
$0005	Port B Data Direction (DDRB)	Read:	0	0	0	0	0	0	DDRB1	DDRB0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0006	Unimplemented	Read: / Write:								
$0007	Unimplemented	Read: / Write:								
$0008	Timer Status/Control (TSCR)	Read:	TOF	RTIF	TOIE	RTIE	0	0	RT1	RT0
		Write:					TOFR	RTIFR		
		Reset:	0	0	0	0	0	0	1	1

= Unimplemented R = Reserved

Figure 2-3. MC68HC05K3 I/O Registers (Sheet 1 of 4)

Addr.	Name		Bit 7	6	5	4	3	2	1	Bit 0
$0009	Timer Counter (TCNTR)	Read:	TCR7	TCR6	TCR5	TCR4	TCR3	TCR2	TCR1	TCR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$000A	IRQ Status/Control (ISCR)	Read:	IRQE	0	0	0	IRQF	0	0	0
		Write:				R			IRQR	
		Reset:	1	0	0	0	0	0	0	0
$000B	Unimplemented	Read:								
		Write:								
$000C	Unimplemented	Read:								
		Write:								
$000D	Unimplemented	Read:								
		Write:								
$000E	Personality EEPROM Bit Select (PEBSR)	Read:	PEB7	PEB6	PEB5	PEB4	PEB3	PEB2	PDB1	PDB0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$000F	Personality EEPROM Status/Control (PESCR)	Read:	PEDATA	PEBULK	PEPGM	PEBYTE	CPEN	CPCLK	0	PEPCZF
		Write:								
		Reset:	1	0	0	0	0	0	0	1
$0010	Port A Pulldown Inhibit (PDRA)	Read:								
		Write:	PDIA7	PDIA6	PDIA5	PDIA4	PDIA3	PDIA2	PDIA1	PDIA0
		Reset:	0	0	0	0	0	0	0	0
$0011	Port B Pulldown Inhibit (PDRB)	Read:								
		Write:							PDIB1	PDIB0
		Reset:							0	0

= Unimplemented R = Reserved

Figure 2-3. MC68HC05K3 I/O Registers (Sheet 2 of 4)

Addr.	Name		Bit 7	6	5	4	3	2	1	Bit 0
$0012	Unimplemented	Read:								
		Write:								
$0013	Unimplemented	Read:								
		Write:								
$0014	Unimplemented	Read:								
		Write:								
$0015	Unimplemented	Read:								
		Write:								
$0016	Unimplemented	Read:								
		Write:								
$0017	Unimplemented	Read:								
		Write:								
$0018	Unimplemented	Read:								
		Write:								
$0019	Unimplemented	Read:								
		Write:								
$001A	Unimplemented	Read:								
		Write:								
$001B	Unimplemented	Read:								
		Write:								

= Unimplemented R = Reserved

Figure 2-3. MC68HC05K3 I/O Registers (Sheet 3 of 4)

Addr.	Name		Bit 7	6	5	4	3	2	1	Bit 0
$001C	Unimplemented	Read:								
		Write:								
$001D	Unimplemented	Read:								
		Write:								
$001E	Unimplemented	Read:								
		Write:								
$001F	Reserved	Read:	R	R	R	R	R	R	R	R
		Write:								

= Unimplemented R = Reserved

Figure 2-3. MC68HC05K3 I/O Registers (Sheet 4 of 4)

2.4 Random-Access Memory (RAM)

The total RAM consists of 64 bytes (including the stack) at locations $00C0–$00FF. The stack pointer can access 32 locations from $00E0 to $00FF. The stack begins at address $00FF and proceeds down to $00E0. 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 Read-Only Memory (ROM)

There are a total of 928 bytes of user ROM on the chip. This includes 160 bytes in page zero from $0020–$00BF, 760 bytes of user ROM with locations $0100–$03F7 for user program storage, and 8 bytes of user vectors at locations $03F8–$03FF.

Memory Map

Section 3. Central Processor Unit (CPU) Core

3.1 Contents

3.2 Introduction

The MC68HC05K3 has a 1024-byte memory map. Therefore, it uses only the lower 10 bits of the address bus. In the following discussion, the upper six bits of the address bus can be ignored. Also, by using a mask option, the STOP instruction can be converted from acting as the normal STOP instruction. The stack area also is reduced to 32 bytes due to the limited amount of RAM. Therefore, the stack pointer is reduced to only five bits, only decrements down to $00E0, and then wraps around to $00FF. All other instructions and registers behave as described in *M6805 HMOS/M146805 CMOS Family User's Manual,* Motorola document order number M6805UM/AD3.

3.3 Registers

The MCU contains five registers that are hard-wired within the CPU and are not part of the memory map. These five registers are shown in **Figure 3-1**.

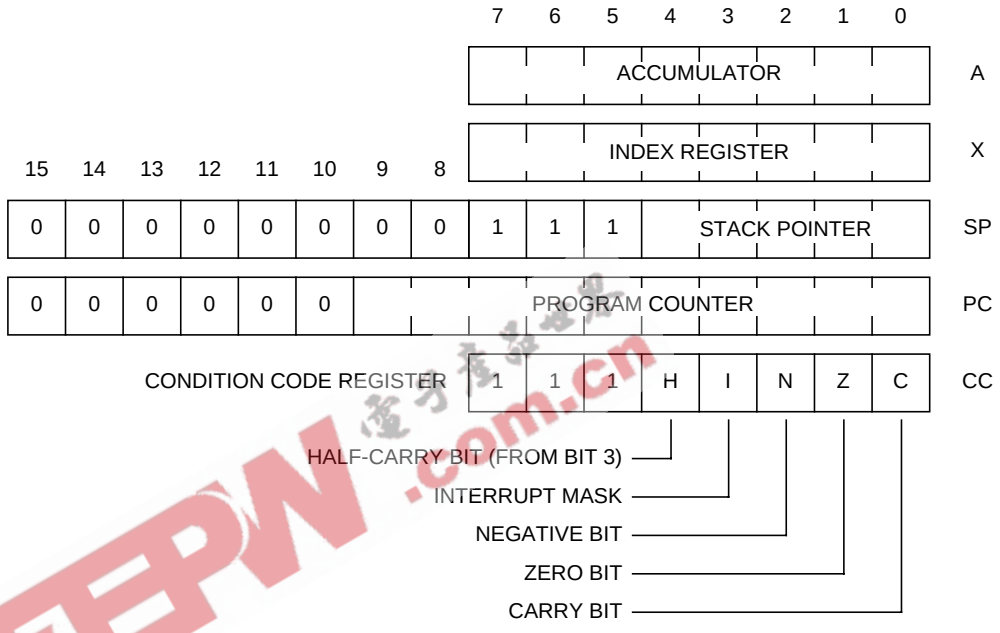


Figure 3-1. M68HC05 Programming Model

For a more complete description of the M68HC05 CPU functions, refer to:

- *M6805 HMOS, M146805 CMOS Family User's Manual,* Motorola order number M6805UM(AD3)
- *HC05 Applications Guide,* Motorola order number M68HC05AG/AD
- *Understanding Small Microcontrollers,* Motorola order number M68HC05TB/D

Any specific differences in the operation of all CPU registers or bits is described in the following sections.

3.3.1 Stack Pointer (SP)

The stack pointer shown in **Figure 3-1** is a 16-bit register internally. In devices with memory maps less than 64 Kbytes, the unimplemented upper address lines are ignored. The stack pointer contains the address of the next free location on the stack. When accessing memory, the 11 most significant bits are permanently set to 00000000111. The five least significant register bits are appended to these 11 fixed bits to produce an address within the range of $00FF to $00E0. Subroutines and interrupts may use up to 32 ($20) locations. If 32 locations are exceeded, the stack pointer wraps around to $00FF and writes over the previously stored information.

3.3.2 Program Counter (PC)

The program counter shown in **Figure 3-1** is a 16-bit register internally. The program counter contains the address of the next instruction or operand to be fetched. The six most significant bits of the program counter are ignored internally and appear as 000000 when stacked onto the RAM.

Central Processor Unit (CPU) Core

Section 4. Interrupts

4.1 Contents

4.2 Introduction

The MCU can be interrupted four different ways:

1. Non-maskable software interrupt instruction (SWI)
2. External asynchronous interrupt ($\overline{IRQ}$)
3. External interrupt via IRQ on PA0–PA3 (enabled by a mask option)
4. Internal timer interrupt (TIMER)

4.3 Central Processor Unit (CPU) Interrupt Processing

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

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

When an interrupt is to be processed, the CPU fetches the address of the appropriate interrupt software service routine from the vector table at locations $03F8–$03FF as defined in **Table 4-1**.

Table 4-1. Vector Addresses for Interrupts and Reset

Register	Flag Name	Interrupts	CPU Interrupts	Vector Addresses
N/A	N/A	Reset	$\overline{\text{RESET}}$	$03FE–$03FF
N/A	N/A	Software	SWI	$03FC–$03FD
ISCR	IRQF	External interrupt	$\overline{\text{IRQ}}$	$03FA–$03FB
TSCR	TOF	Timer overflow	TIMER	$03F8–$03F9
TSCR	RTIF	Real-time interrupt	TIMER	$03F8–$03F9

A return-from-interrupt (RTI) instruction is used to signify when the interrupt software service routine is complete. The RTI instruction causes the register contents to be recovered from the stack and normal processing to resume at the next instruction that was to be executed when the interrupt took place. **Figure 4-1** shows the sequence of events that occurs during interrupt processing. **Figure 4-2** shows the stacking and unstacking order into the RAM that is associated with an interrupt service routine.

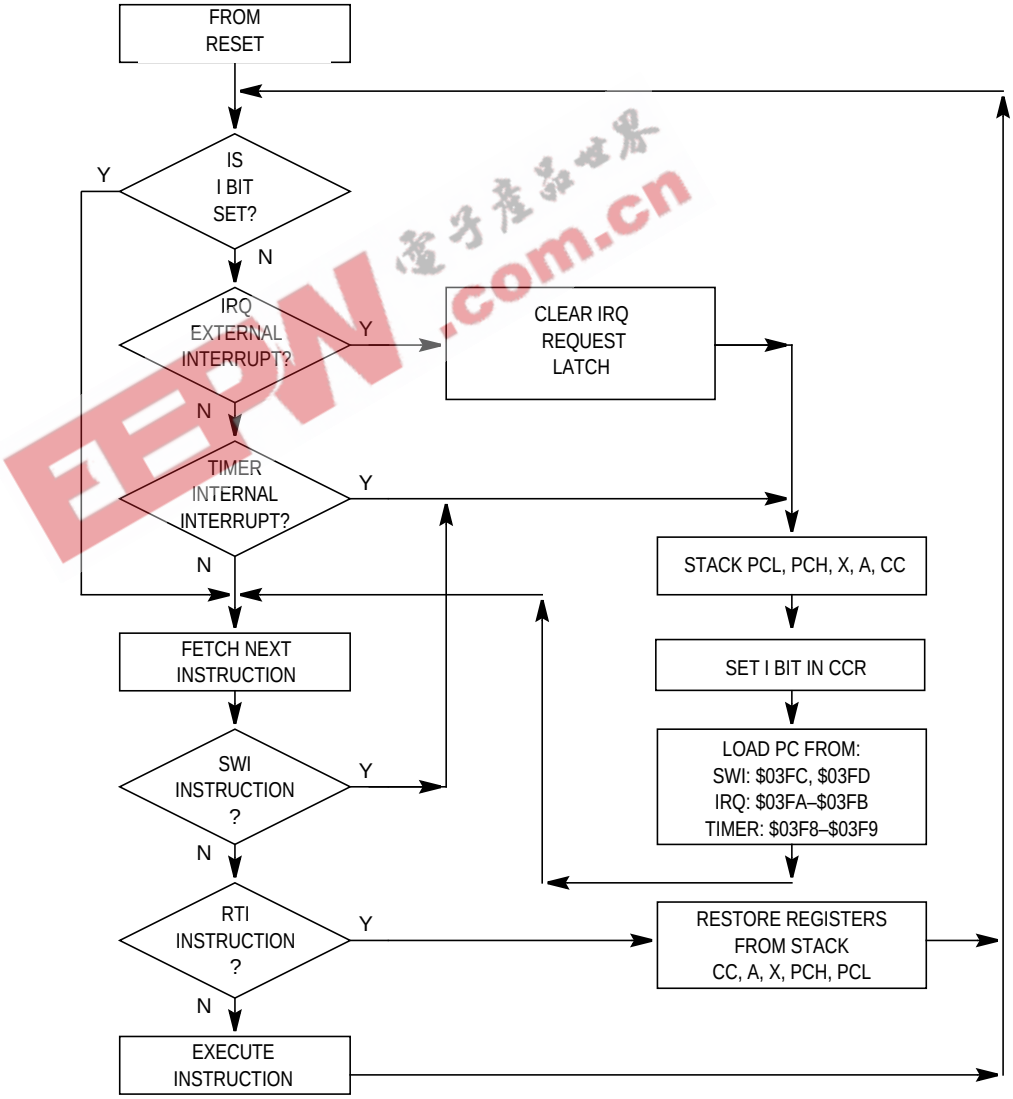


Figure 4-1. Interrupt Processing Flowchart

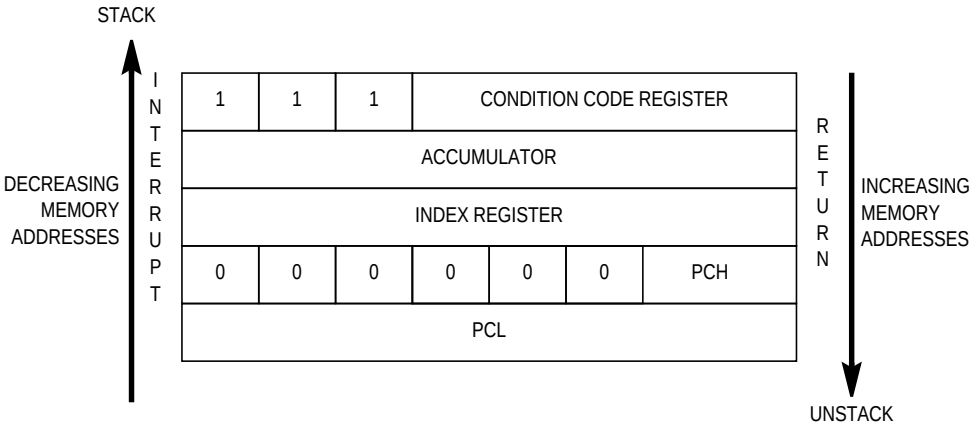


Figure 4-2. Interrupt Stacking Order

4.4 Reset Interrupt Sequence

The reset function is not in the strictest sense an interrupt; however, it is acted upon in a similar manner. A low-level input on the $\overline{\text{RESET}}$ pin or an internally generated reset (RST) signal causes the program to vector to its starting address, which is specified by the contents of memory locations $03FE and $03FF. The I bit in the condition code register also is set. The MCU is configured to a known state during this type of reset, as described in **Section 5. Resets**.

4.5 Software Interrupt (SWI)

The SWI is an executable instruction and a non-maskable interrupt since it is executed regardless of the state of the I bit in the CCR. If the I bit is 0 (interrupts enabled), the SWI instruction executes after interrupts that were pending before the SWI was fetched or before interrupts generated after the SWI was fetched. The interrupt service routine address is specified by the contents of memory locations $03FC and $03FD.

4.6 Hardware Interrupts

All hardware interrupts except RESET are maskable by the I bit in the CCR. If the I bit is set, all hardware interrupts (internal and external) are disabled. Clearing the I bit enables the hardware interrupts. The two types of hardware interrupts are explained here.

4.6.1 External Interrupt ($\overline{IRQ}$)

The $\overline{IRQ}$ pin provides an asynchronous interrupt to the CPU. A block diagram of the IRQ function is shown in **Figure 4-3**.

The $\overline{IRQ}$ pin is one source of an IRQ interrupt, and a mask option is available to enable the four lower order port A pins (PA0–PA3) to act as other IRQ interrupt sources. All of these sources are combined into a single ORing function that is latched by the IRQ latch.

The IRQ latch is set on the falling edge of the $\overline{IRQ}$ pin or on the rising edge of a PA0–PA3 pin, if port A interrupts have been enabled by the mask option.

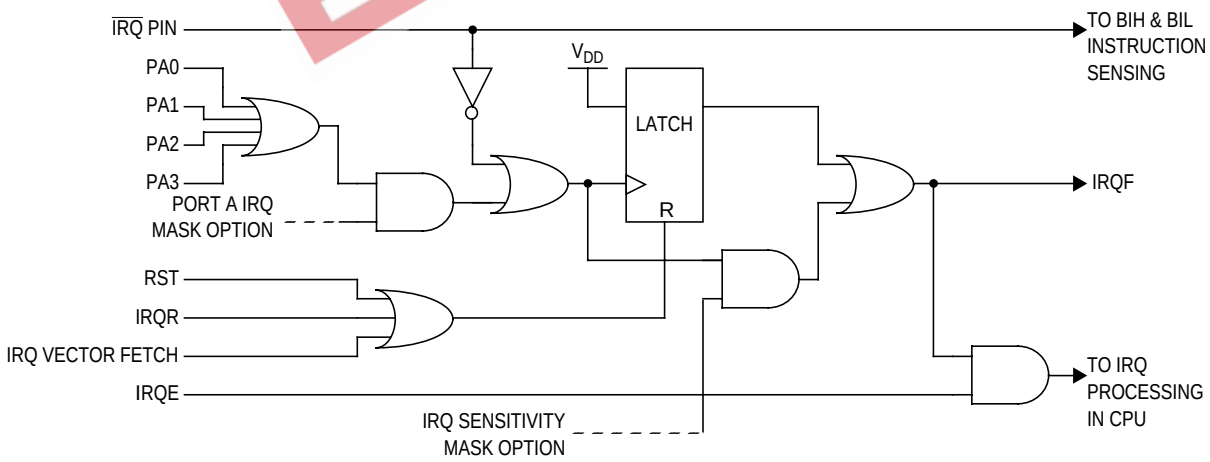


Figure 4-3. IRQ Function Block Diagram

If the mask option for **edge-sensitive only** IRQ is used, only the IRQ latch output can activate an IRQF flag which creates a request to the CPU to generate the IRQ interrupt sequence. This makes the IRQ interrupt sensitive to these cases:

- If the port A interrupts are **disabled** by a mask option, only a falling edge on the $\overline{\text{IRQ}}$ pin initiates an IRQ interrupt.
- If the port A interrupts are **enabled** by a mask option, these conditions initiate an IRQ interrupt:
  - A falling edge on the $\overline{\text{IRQ}}$ pin with all the PA0–PA3 pins at a low level
  - A rising edge on one PA0–PA3 pin with all other PA0–PA3 pins at a low level and the $\overline{\text{IRQ}}$ pin at a high level

If the mask option for **edge- and level-sensitive** IRQ is used, the active high state of the IRQ latch input also can activate an IRQF flag, which creates a request to the CPU to generate the IRQ interrupt sequence. This makes the IRQ interrupt sensitive to these cases:

- If the port A interrupts are **disabled** by a mask option, only these conditions initiate an IRQ interrupt:
  - A low level on the $\overline{\text{IRQ}}$ pin
  - Falling edge on the $\overline{\text{IRQ}}$ pin
- If the port A interrupts are **enabled** by a mask option, these conditions initiate an IRQ interrupt:
  - A low level on the $\overline{\text{IRQ}}$ pin with all the PA0–PA3 pins at a low level
  - Falling edge on the $\overline{\text{IRQ}}$ pin with all the PA0–PA3 pins at a low level
  - High level on any one of the PA0–PA3 pins with the $\overline{\text{IRQ}}$ pin at a high level
  - Rising edge on any PA0–PA3 pin with all other PA0–PA3 pins at a low level and the $\overline{\text{IRQ}}$ pin at a high level

The IRQE enable bit controls whether an active IRQF flag can generate an IRQ interrupt sequence. This interrupt is serviced by the interrupt

service routine located at the address specified by the contents of $03FA and $03FB.

Entering the interrupt service routine automatically clears the IRQ latch. The IRQ interrupt service routine also may clear the IRQ latch by writing a logic 1 to the IRQR acknowledge bit in the ISCR. As long as the output state of the IRQF flag bit is active, the CPU continuously re-enters the IRQ interrupt sequence following an RTI instruction until the active state is removed or the IRQE enable bit is cleared.

4.6.2 IRQ Status/Control Register

The IRQ interrupt function is controlled by the IRQ status/control register (ISCR) located at $000A as shown in Figure 4-4. All unused bits in the ISCR read as logic 0s. A reset clears the IRQF bit and sets the IRQE bit.

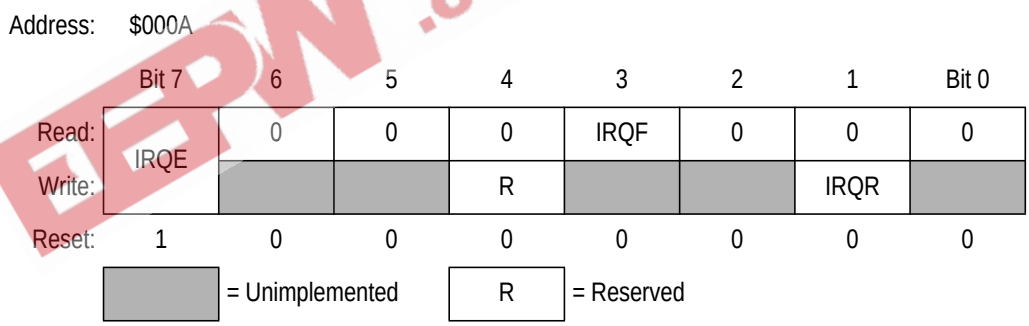


Figure 4-4. IRQ Status/Control Register (ISCR)

IRQR — IRQ Interrupt Acknowledge Bit

The IRQR acknowledge bit clears an IRQ interrupt request by clearing the IRQ latch. If the IRQ latch is set again while in the IRQ service routine (before an RTI instruction is executed), the CPU re-enters the IRQ interrupt service routine unless the IRQ latch is cleared. Writing a logic 1 to the IRQR acknowledge bit clears the IRQ latch. Writing a logic 0 to the IRQR acknowledge bit has no effect on the IRQ latch. The IRQR acknowledge bit always reads as a logic 0.

IRQF — IRQ Interrupt Request Bit

The IRQF flag bit indicates that an IRQ request is pending. Writing to the IRQF flag bit has no effect on it. The IRQF flag bit is cleared automatically when the IRQ vector is fetched and the service routine is entered. The IRQF flag bit also can be cleared by writing a logic 1 to the IRQR acknowledge bit to clear the IRQ latch and also condition the external IRQ sources to be inactive if the edge- and level-sensitive mask option is selected. In this way, any additional setting of the IRQF flag bit while in the service routine can be ignored by clearing the IRQF flag bit just before exiting the service routine. If the IRQF flag bit is set again while in the IRQ service routine, the CPU re-enters the IRQ interrupt sequence unless the IRQF flag bit is cleared. The IRQF flag bit is cleared by reset.

IRQE — IRQ Interrupt Enable Bit

The IRQE bit enables or disables the IRQF flag bit to initiate an IRQ interrupt sequence. If the IRQE enable bit is set, the IRQF flag bit can generate an interrupt sequence. If the IRQE enable bit is cleared, the IRQF flag bit cannot generate an interrupt sequence. Reset sets the IRQE enable bit, thereby enabling IRQ interrupts once the I bit is cleared. Execution of the STOP or WAIT instructions causes the IRQE bit to be set to allow the external IRQ to exit these modes. In addition, reset also sets the I bit, which masks all interrupt sources.

NOTE: *If the I bit is cleared, any instruction that sets the IRQE enable bit when the IRQF flag bit is already set initiates an IRQ interrupt sequence immediately after that instruction.*

4.6.3 Port A Interrupts (PA0–PA3)

The IRQ interrupt also can be triggered by inputs to PA0–PA3 port pins as described in **4.6.1 External Interrupt (IRQ)** if the port interrupts mask option is used. If enabled, the lower four bits of port A can activate the IRQ interrupt function and the interrupt operation is the same as the input to the $\overline{\text{IRQ}}$ pin. The mask option allows all of these input pins to be ORed with the input present on the $\overline{\text{IRQ}}$ pin. All PA0–PA3 pins must be selected as a group and as an additional IRQ interrupt source. All the port A interrupt sources also are controlled by the IRQE enable bit.

NOTE: *The BIH and BIL instructions apply only to the level on the $\overline{IRQ}$ pin itself and not to the output of the logic OR gate with PA0–PA3 pins. The state of the individual port A pins can be checked by reading the appropriate port A pins as inputs.*

NOTE: *If port A interrupts are enabled, the state of PA0–PA3 pins may cause an IRQ interrupt regardless of whether these pins are configured as inputs or outputs. (See* ***Section 7. Parallel Input/Output (I/O)****.)*

4.6.4 Timer Interrupt (TIMER)

The timer interrupt is generated by the 8-bit timer when either a timer overflow or a real-time interrupt has occurred, as described in **Section 8. 8-Bit Timer**. The interrupt flags and enable bits for the timer interrupts are in the timer status/control register (TSCR) located at $0008. The I bit in the CCR must be clear for the timer interrupt to be enabled. Either of these two interrupts vector to the same interrupt service routine located at the address specified by the contents of memory locations $03F8 and $03F9.

Interrupts

Section 5. Resets

5.1 Contents

5.2 Introduction

The MCU can be reset from four sources: one external input and three 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 that drive external pins are reset by the synchronous reset signal (RST) coming from a latch, which is synchronized to the PH2 bus clock and set by any of the four reset sources.

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

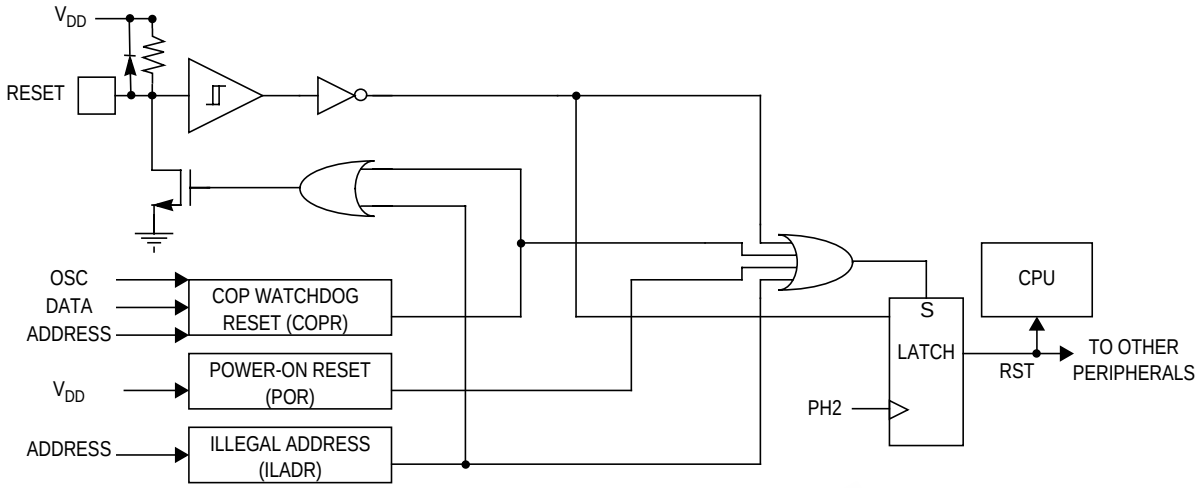


Figure 5-1. Reset Block Diagram

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 noise immunity. This external reset occurs whenever the $\overline{\text{RESET}}$ pin is pulled low and remains in reset until the $\overline{\text{RESET}}$ pin rises to a logic 1. This active low input generates the RST signal and resets the CPU and peripherals.

5.4 Internal Resets

The three internally generated resets are:

- Initial power-on reset (POR) function
- Computer operating properly (COP) watchdog timer reset
- Illegal address detector reset (ILADR)

5.4.1 Power-On Reset (POR)

The internal POR is generated on power-up of the internal CPU 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 (a

“brown-out” condition). After the oscillator becomes active, a mask option selects an oscillator stabilization delay of 16 or 4064 cycles of the internal processor bus clock (PH2).

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

5.4.2 Computer Operating Properly Reset (COPR)

A COP watchdog timer can be enabled by a mask option. The internal COP reset (COPR) is generated automatically by a timeout of the COP watchdog timer. This timeout occurs if the counter in the COP watchdog timer is not reset (cleared) within a specific time by a user program reset sequence. Refer to **8.4 COP Watchdog Timer** for more information on this timeout feature.

The COPR generates the RST signal that resets the CPU and other peripherals. If any other reset function is active at the end of the COPR reset signal, the RST signal remains in the reset condition until the other reset condition(s) end(s).

The COP watchdog reset activates the internal pulldown device connected to the $\overline{\text{RESET}}$ pin for one cycle of the internal processor bus clock, PH2.

5.4.3 Illegal Address Reset (ILADR)

The internal ILADR reset is generated when an instruction opcode fetch occurs from an address in the I/O address area ($0000–$001F). The ILADR generates the RST signal that resets the CPU and other peripherals. If any other reset function is active at the end of the ILADR reset signal, the RST signal remains in the reset condition until the other reset condition(s) end(s). The ILADR reset activates the internal pulldown device connected to the $\overline{\text{RESET}}$ pin for **one** cycle of the internal processor bus clock, PH2.

Resets

Section 6. Operational Modes

6.1 Contents

6.2 Introduction

The MC68HC05K3 is capable of running in one of several operational modes. These modes include:

- Low-power operational modes
  - Stop mode
  - Wait mode
  - Halt mode
- Serial program mode

6.3 Low-Power Modes

The WAIT and STOP/HALT instructions provide two low-power operational modes that reduce the power required for the MCU by stopping various internal clocks and/or the on-chip oscillator. The flow of the stop, halt, and wait modes is shown in **Figure 6-1**.

6.3.1 Stop Mode

The STOP instruction can result in one of two modes of operation depending on its mask option. The mask option can make the STOP instruction operate the same as the STOP instruction in other M68HC05 Family members and place the device in stop mode. Or the mask option can make the STOP instruction behave like a WAIT instruction (except that the restart time involves a delay) and place the device in halt mode.

The mask option enabling the execution of 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 the COP watchdog timer.

When the CPU enters stop mode, the interrupt flags (TOF and RTIF) and the interrupt enable bits (TOFE and RTIE) in the TSCR are cleared by internal hardware to remove any pending timer interrupt requests and to disable any further timer interrupts. Execution of the STOP instruction automatically clears the I bit in the condition code register and sets the IRQE enable bit in the IRQ status/control register so that the IRQ external interrupt is enabled. All other memory and registers, including the other bits in the TSCR, remain unaltered.

The MCU can be brought out of stop mode only by an IRQ external interrupt, an IRQ from port A (if mask option is enabled), or an externally generated RESET. When exiting stop mode, the internal oscillator resumes after an oscillator stabilization delay of either 16 or 4064 cycles (depending on mask option state) of the internal processor clock.

NOTE: *If enabled by a mask option, the STOP instruction causes the oscillator to stop and, therefore, disable the COP watchdog timer. If the COP watchdog timer is used and the part is never intended to enter stop mode, the mask option that should be used is the one that disables the STOP instruction and changes the stop mode to the halt mode. See* ***6.3.4 COP Watchdog Timer Considerations*** *for more details.*

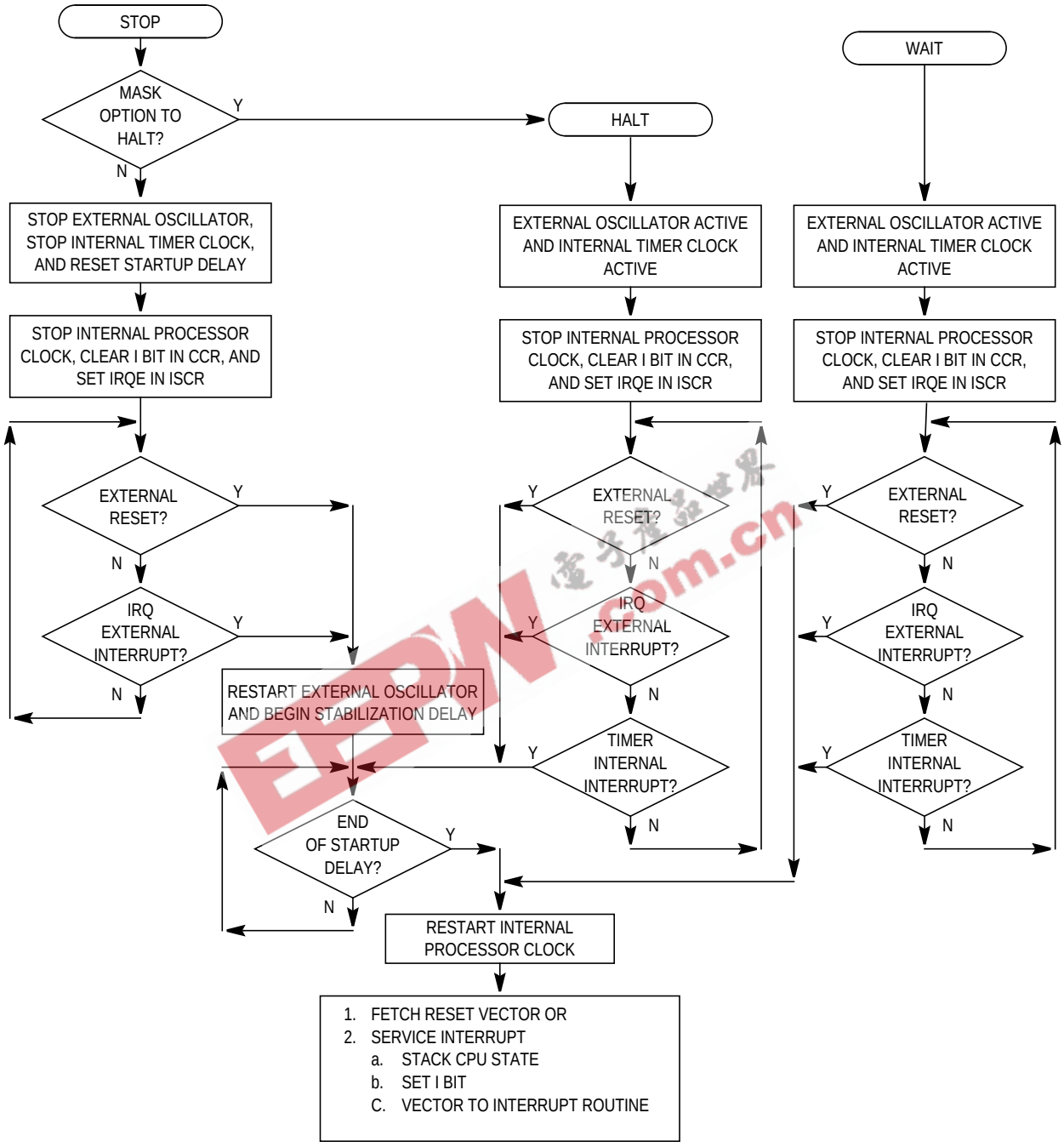


Figure 6-1. Stop/Halt/Wait Flowcharts

6.3.2 Halt Mode

Execution of the STOP instruction with a mask option to disable the stop mode places the MCU in a low-power halt mode, which consumes more power than stop mode. In halt mode, the internal processor clock is halted, suspending all processor and internal bus activity. Internal timer clocks remain active, permitting interrupts to be generated from the timer or a reset to be generated from the COP watchdog timer. Execution of the STOP instruction in the halt mode automatically clears the I bit in the condition code register and sets the IRQE enable bit in the IRQ status/control register so that the IRQ external interrupt is enabled. All other registers, memory, and input/output lines remain in their previous states.

If timer interrupts are enabled, a timer interrupt causes the processor to exit halt mode and resume normal operation. Halt mode also can be exited when an external IRQ or external RESET occurs. When exiting halt mode, the internal processor clock resumes after a variable delay. Depending on the mask option state, the maximum oscillator stabilization delay is 16 or 4064 cycles of the internal processor clock.

Using the mask option to disable the STOP instruction prevents the STOP instruction from halting the oscillator or affecting the COP watchdog timer similar to wait mode. However, the recovery method introduces some startup delay in the processor clock.

NOTE: *Halt mode is not intended for normal use, but is provided to keep the COP watchdog timer active if the STOP instruction opcode is executed inadvertently.*

6.3.3 Wait Mode

The WAIT instruction places the MCU in a low-power wait mode, which consumes more power than stop mode. In wait mode, the internal processor clock is halted, suspending all processor and internal bus activity. Internal timer clocks remain active, permitting interrupts to be generated from the timer or a reset to be generated from the COP watchdog timer. Execution of the WAIT instruction automatically clears the I bit in the condition code register and sets the IRQE enable bit in the

IRQ status/control register so that the IRQ external interrupt is enabled. All other registers, memory, and input/output lines remain in their previous states.

If timer interrupts are enabled, a timer interrupt causes the processor to exit wait mode and resume normal operation. Thus, the timer can be used to generate a periodic exit from wait mode. Wait mode also is exited when an external IRQ or RESET occurs.

6.3.4 COP Watchdog Timer Considerations

If the COP watchdog timer is enabled by the mask option, any execution of the STOP instruction (either intentional or inadvertent due to the CPU being disturbed) causes the oscillator to halt and prevent the COP watchdog timer from timing out unless the STOP instruction is disabled by a mask option.

If the mask option is selected to enable the COP watchdog timer, the COP resets the MCU when it times out. Therefore, it is recommended that the mask option be selected to disable the COP watchdog for a system that must have intentional uses of the wait mode for periods longer than the COP timeout period.

6.4 PEEPROM Serial Programming Mode

The internal personality EEPROM (PEEPROM) can be erased, read, or programmed through the application of serial data patterns to the $\overline{\text{IRQ}}$ and PB0 pins, if the PEEPROM serial programming mode is selected following reset. Refer to **9.6 PEEPROM Serial Programming** for details.

Operational Modes

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

7.1 Contents

7.2 Introduction

In single-chip mode, 10 bidirectional input/output (I/O) lines are arranged as one 8-bit I/O port (port A) and one 2-bit I/O port (port B). The individual bits in these ports are programmable as either inputs or outputs under software control by the data direction registers (DDRs). All port A and port B I/O pins have individual software programmable pulldown devices which can be enabled by a mask option. Some port A pins also have the additional properties of sinking higher current or acting as additional IRQ interrupt input sources. One of the port B pins also may be used as an output for a 3-pin resistor capacitor (RC) oscillator option.

7.3 Port A

Port A is an 8-bit bidirectional port that shares four of its pins with the IRQ interrupt system, as shown in Figure 7-1. Each port A pin is controlled by the corresponding bits in a data direction register, a data register, and a pulldown register.

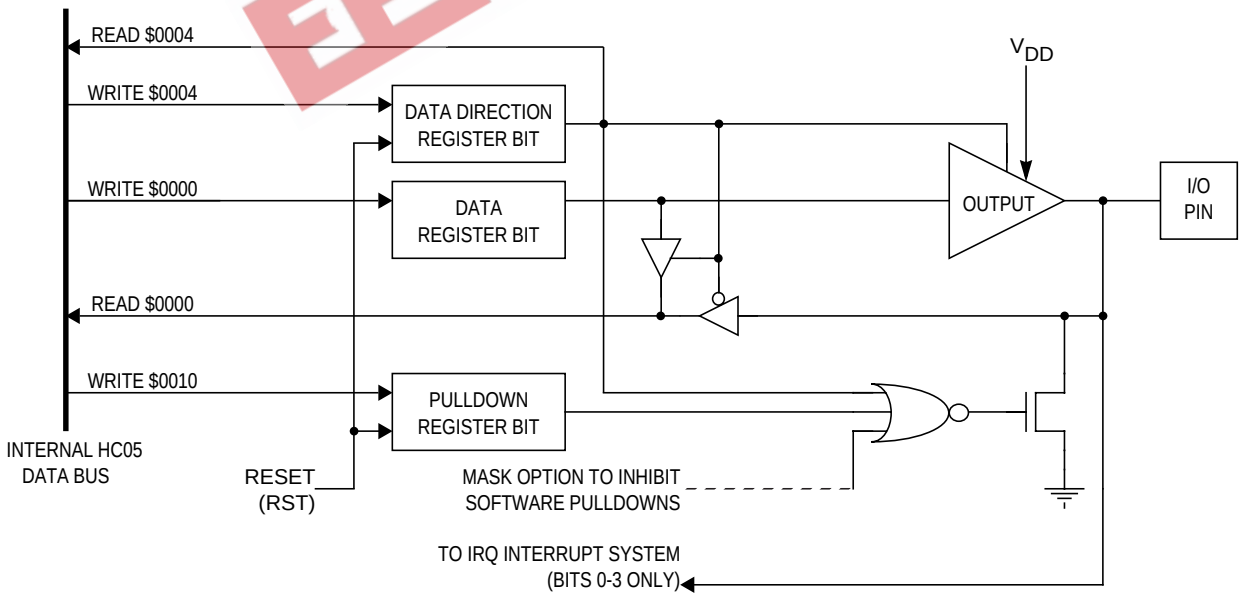


Figure 7-1. Port A I/O Circuitry

The port A data register is located at address $0000. The port A data direction register (DDRA) is located at address $0004. The port A pulldown register (PDRA) is located at address $0010. Reset clears both the DDRA and the PDRA. The port A data register is unaffected by reset.

7.3.1 Port A Data Register

Each port A I/O pin has a corresponding bit in the port A data register. When a port A pin is programmed as an output, the state of the corresponding data register bit determines the state of the output pin. When a port A pin is programmed as an input, any read of the port A data register returns the logic state of the corresponding I/O pin, and any write to the port A data register is saved in the data register, but is not applied to the corresponding I/O pin. The port A data register is unaffected by reset. The port A data register is indeterminant after initial power-up.

7.3.2 Port A Data Direction Register

Each port A I/O pin may be programmed as an input by clearing the corresponding bit in the DDRA or programmed as an output by setting the corresponding bit in the DDRA. When a DDRA bit is set, the corresponding pulldown device is disabled. The DDRA can be accessed at address $0004. The DDRA is cleared by reset.

7.3.3 Port A Pulldown Inhibit Register

All port A I/O pins have software programmable pulldown devices which may be enabled by a mask option. If enabled by mask option, the software programmable pulldowns are activated by clearing their corresponding bit in the PDRA or disabled by setting the corresponding bit in the PDRA. If disabled by a mask option, all pulldowns are disabled. A pulldown on an I/O pin can be activated only if the I/O pin is programmed as an input.

The PDRA is a write-only register and any reads of location $0010 return undefined results. Since reset clears both the DDRA and the PDRA, all pins initialize as inputs with the pulldown devices active (if enabled by mask option).

Address: $0010

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

= Unimplemented

Figure 7-2. Port A Pulldown Inhibit Register (PDRA)

7.3.4 Port A Light-Emitting Diode (LED) Drive Capability

The outputs of port A pins 4–7 are capable of sinking high current for light-emitting diode (LED) drive capability.

7.3.5 Port A I/O Pin Interrupts

The inputs for the lower four bits of port A can be connected through an OR gate to the IRQ latched input to the CPU by a mask option. When connected as an alternate source of an IRQ interrupt, the port A input pins behave the same as the $\overline{\text{IRQ}}$ pin itself, except that their active state is a logic 1 or a rising edge. The normal $\overline{\text{IRQ}}$ pin has an active state that is a logic 0 or a falling edge depending on the mask option.

If the mask option for edge- and level-sensitive interrupts and the mask option for port A interrupts are both used, the presence of a logic 1 on any one of the lower four port A pins causes an IRQ interrupt request. If the mask option for edge-sensitive-only interrupts and the mask option for port A interrupts are both used, the occurrence of a rising edge on any one of the PA0–PA3 pins causes an IRQ interrupt request, as long as the other PA0–PA3 pins are at a low level. As long as any one of the PA0–PA3 IRQ inputs remains at a logic 1 level, or the $\overline{\text{IRQ}}$ remains at a logic 0 level, the other PA0–PA3 IRQ inputs are effectively ignored. Port

interrupts will be generated with the above PA0–PA3 I/O state regardless of whether the port is configured as an input or output.

NOTE: *The BIH and BIL instructions apply only to the level on the $\overline{IRQ}$ pin itself and not to the internal IRQ input to the CPU. Therefore, BIH and BIL cannot be used to test the state of the lower four port A input pins as a group. Each port A interrupt pin can be tested by reading the port A data register at $0000.*

7.4 Port B

Port B is a 2-bit bidirectional port that shares one of its pins with the RC oscillator as shown in **Figure 7-3**. Each port B pin is controlled by the corresponding bits in a data direction register, a data register, and a pulldown register.

The port B data register is located at address $0001. The port B data direction register (DDRB) is located at address $0005, and the port B pulldown register (PDRB) is located at address $0011. Reset clears both the DDRB and the PDRB. The port B data register is unaffected by reset. The port B data register is indeterminant after initial powerup.

7.4.1 Port B Data Register

Each port B I/O pin has a corresponding bit in the port B data register. When a port B pin is programmed as an output, the state of the corresponding data register bit determines the state of the output pin. When a port B pin is programmed as an input, any read of the port B data register returns the logic state of the corresponding I/O pin, and any write to the port B data register is saved in the data register, but is not applied to the corresponding I/O pin. Unused bits 2–7 are always read as logic 0s, and any write to these bits is ignored. The port B data register is unaffected by reset. The port B data register is indeterminant after initial power-up.

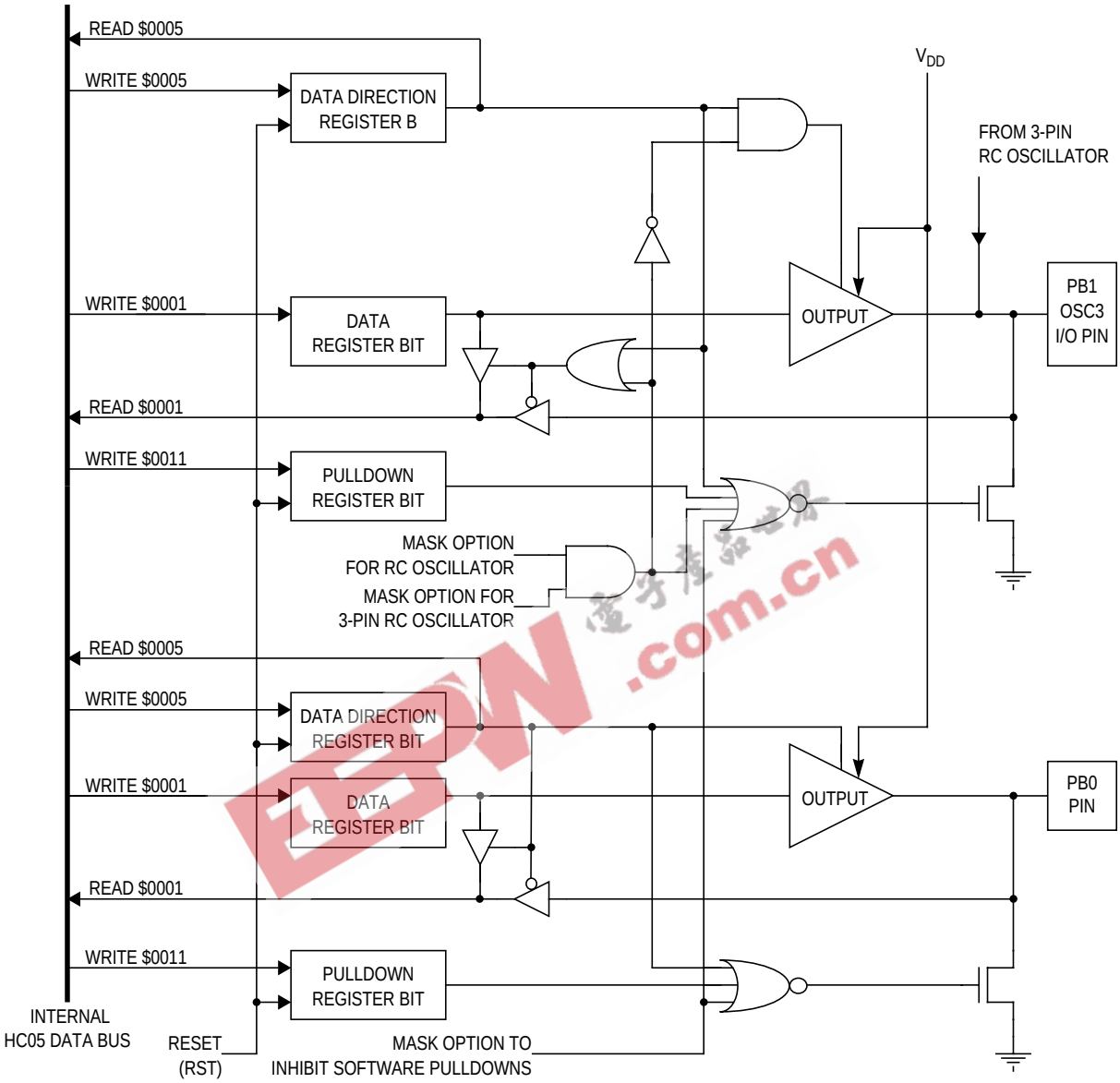


Figure 7-3. Port B I/O Circuitry

7.4.2 Port B Data Direction Register

Each port B I/O pin may be programmed as an input by clearing the corresponding bit in the DDRB or programmed as an output by setting the corresponding bit in the DDRB. When a DDRB bit is set, the corresponding pulldown device is disabled. The DDRB can be accessed at address $0005. Unused bits 2–7 are always read as logic 0s, and any write to these bits is ignored. The DDRB is cleared by reset.

7.4.3 Port B Pulldown Inhibit Register

Each port B I/O pin has a software programmable pulldown device which can be enabled by a mask option. If enabled by a mask option, the software programmable pulldowns are activated by clearing the corresponding bit in the PDRB or disabled by setting the corresponding bit in the PDRB. If disabled by a mask option, all pulldowns are disabled. A pulldown on an I/O pin can be activated only if the I/O pin is programmed as an input.

The PDRB is a write-only register and any reads of location $0011 return undefined results. Since reset clears both the DDRB and the PDRB, all pins initialize as inputs with the pulldown devices active (if enabled by mask option).

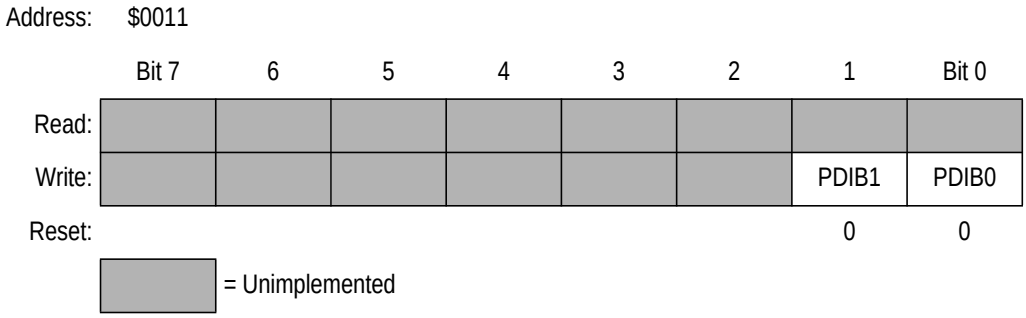


Figure 7-4. Port B Pulldown Inhibit Register (PDRB)

7.4.4 Port B with 3-Pin RC Oscillator

The PB1/OSC3 pin may be used as an output from a 3-pin RC oscillator when the mask option for a 3-pin RC oscillator is used. In this case, the following conditions apply:

- The PB1 data register bit can be used as a read/write storage location without affecting the oscillator. PB1 is unaffected by reset.
- The DDRB1 data direction bit can be used as a read/write storage location without affecting the oscillator. DDRB1 is cleared by reset.
- The software programmable pulldown on PB1/OSC3 is disabled, regardless of the mask option selection for the software programmable pulldowns or the state of PDRB1.

7.5 I/O Port Programming

All I/O pins can be programmed as inputs or outputs, with or without pulldown devices.

7.5.1 Pin Data Direction

The direction of a pin is determined by the state of its corresponding bit in the associated port data direction register (DDR). A 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.

The data direction bits DDRB0, DDRB1, and DDRA0–DDRA7 are read/write bits that can be manipulated with read-modify-write instructions. At power-on or reset, all DDRs are cleared, which configures all port pins as inputs. If the mask option for software programmable pulldowns is selected, all pins initially power-up with their software programmable pulldowns enabled.

7.5.2 Output Pin

When an I/O pin is programmed as an output pin, the state of the corresponding data register bit determines the state of the pin. The state of the data register bits can be altered by writing to address $0000 for port A and address $0001 for port B. Reads of the corresponding data register bit at address $0000 or $0001 return the state of the data register bit, not the state of the I/O pin itself. Therefore, bit manipulation is possible on all pins programmed as outputs.

All pins programmed as outputs have their pulldown devices disabled regardless of the selected mask option for software programmable pulldowns or the state of their PDR bits.

7.5.3 Input Pin

When an I/O pin is programmed as an input pin, the state of the pin can be determined by reading the corresponding data register bit. Any writes to the corresponding data register bit for an input pin is saved by the register bit, but not applied to the corresponding I/O pin until the pin is later programmed to be an output.

If the corresponding bit in the pulldown register is clear (and the mask option for software programmable pulldowns is selected), the input pin also has an activated pulldown device.

Read-modify-write instructions, such as bit manipulation, should not be used on the pulldown registers, since they are write-only.

7.5.4 I/O Pin Transitions

A “glitch” can be generated on an I/O pin when changing it from an input to an output unless the data register is first pre-conditioned to the desired state before changing the corresponding DDR bit from a 0 to a 1.

If the mask option for software programmable pulldowns is selected, a floating input can be avoided by first clearing the pulldown register bit before changing the corresponding DDR from a 1 to a 0. This ensures that the pulldown device is activated on the pin as the I/O pin changes from a driven output to a pulled low input.

7.5.5 I/O Pin Truth Tables

Every pin on port A and PB0 on port B may be programmed as an input or an output under software control, as shown in **Table 7-1** and **Table 7-2**. All port I/O pins also may have software programmable pulldown devices selected by a mask option. The PB1/OSC3 pin on port B also can be programmed as an input or an output under software control, but it has special considerations when selected by a mask option as an output for the 3-pin RC oscillator, as shown in **Table 7-3**. Otherwise, PB1/OSC3 behaves the same as PB0.

Table 7-1. Port A Pin Functions

Software Prog. Pulldown Mask Option*	PDIAx	DDRAx	I/O Pin Mode	Access to PDRA at $0010		Access to DDRA at $0004	Access to Data Register at $0000	
				Read	Write	Read/Write	Read	Write
1	X	0	In, Hi-Z	U	PDIA0–PDIA7	DDRA0–DDRA7	I/O pin	X
1	X	1	OUT	U	PDIA0–PDIA7	DDRA0–DDRA7	PA0–PA7	PA0–PA7
0	0	0	In, Pulldown	U	PDIA0–PDIA7	DDRA0–DDRA7	I/O pin	X
0	0	1	Out	U	PDIA0–PDIA7	DDRA0–DDRA7	PA0–PA7	PA0–PA7
0	1	0	In, Hi-Z	U	PDIA0–PDIA7	DDRA0–DDRA7	I/O pin	X
0	1	1	Out	U	PDIA0–PDIA7	DDRA0–DDRA7	PA0–PA7	PA0–PA7

Notes:
X is don't care state
U is an undefined state

*1 = pulldowns disabled, 0 = pulldowns enabled

Table 7-2. PB0 Pin Functions

Software Prog. Pulldown Mask Option*	PDIB0	DDRB0	I/O Pin Mode	Access to PDRB at $0011		Access to DDRB at $0005	Access to Data Register at $0001	
				Read	Write	Read/Write	Read	Write
1	X	0	In, Hi-Z	U	PDIB0	DDRB0	I/O pin	X
1	X	1	Out	U	PDIB0	DDRB0	PB0	PB0
0	0	0	In, pulldown	U	PDIB0	DDRB0	I/O pin	X
0	0	1	Out	U	PDIB0	DDRB0	PB0	PB0
0	1	0	In, Hi-Z	U	PDIB0	DDRB0	I/O pin	X
0	1	1	Out	U	PDIB0	DDRB0	PB0	PB0

Notes:
X is don't care state
U is an undefined state

*1 = pulldowns disabled, 0 = pulldowns enabled

Table 7-3. PB1/OSC3 Pin Functions

Mask Option (3-Pin)	Software Prog. Pulldown Mask Option*	PDIB1	DDRB1	I/O Pin Mode	Access to PDRB at $0011		Access to DDRB at $0005	Access to Data Register at $0001	
					Read	Write	Read/Write	Read	Write
0	0	1	0	In, Hi-Z	U	PDIB1	DDRB1	I/O pin	X
0	0	1	1	Out	U	PDIB1	DDRB1	PB1	PB1
0	0	0	0	In, Pulldown	U	PDIB1	DDRB1	I/O pin	X
0	0	0	1	Out	U	PDIB1	DDRB1	PB1	PB1
0	0	0	0	In, Hi-Z	U	PDIB1	DDRB1	I/O pin	X
0	0	0	1	Out	U	PDIB1	DDRB1	PB1	PB1
1	X	X	X	RC OSCOUT	U	PDIB1	DDRB1	PB1	PB1

Notes:
X is don't care state
U is an undefined state

*1 = pulldowns disabled, 0 = pulldowns enabled

Parallel Input/Output (I/O)

Section 8. 8-Bit Timer

8.1 Contents

8.2 Introduction

The timer for this device is an 8-bit ripple counter. The features include timer overflow (TOF), power-on reset (POR), real-time interrupt (RTI), and computer operating properly (COP) watchdog timer. This timer is powered down in the stop mode to reduce STOP I_{DD}.

As shown in **Figure 8-1**, the timer is driven by the timer clock, NTF1, divided by four (4). NTF1 has the same phase and frequency as the processor bus clock, PH2, but is not stopped by the wait or halt modes. 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 (TCNTR) 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 or f_{OP}/16, followed by two more stages, with the resulting clock (f_{OP}/16,384) driving the real-time interrupt (RTI) circuit.

The RTI circuit consists of three divider stages with a one-of-four selector. The output of the RTI circuit is further divided by eight to drive

the optional COP watchdog timer circuit, which can be enabled by a mask option. The RTI rate selector bits and the RTI and TOF enable bits and flags are located in the timer control and status register at location $08. The clock frequency that drives the RTI circuit is $f_{OP}/2^{14}$ (or $f_{OP}/16{,}384$) with three additional divider stages giving a maximum interrupt period of $f_{OP}/2^{17}$ (or $f_{OP}/131{,}072$).

The power-on cycle clears the entire counter chain and begins clocking the counter. After 4064 or 16 cycles (depending on mask option), the power-on reset circuit is released, which again clears the counter chain and allows the device to come out of reset. At this point, if $\overline{\text{RESET}}$ is not asserted, the timer starts counting up from 0 and normal device operation begins. If $\overline{\text{RESET}}$ is asserted at any time during operation, the counter chain is cleared.

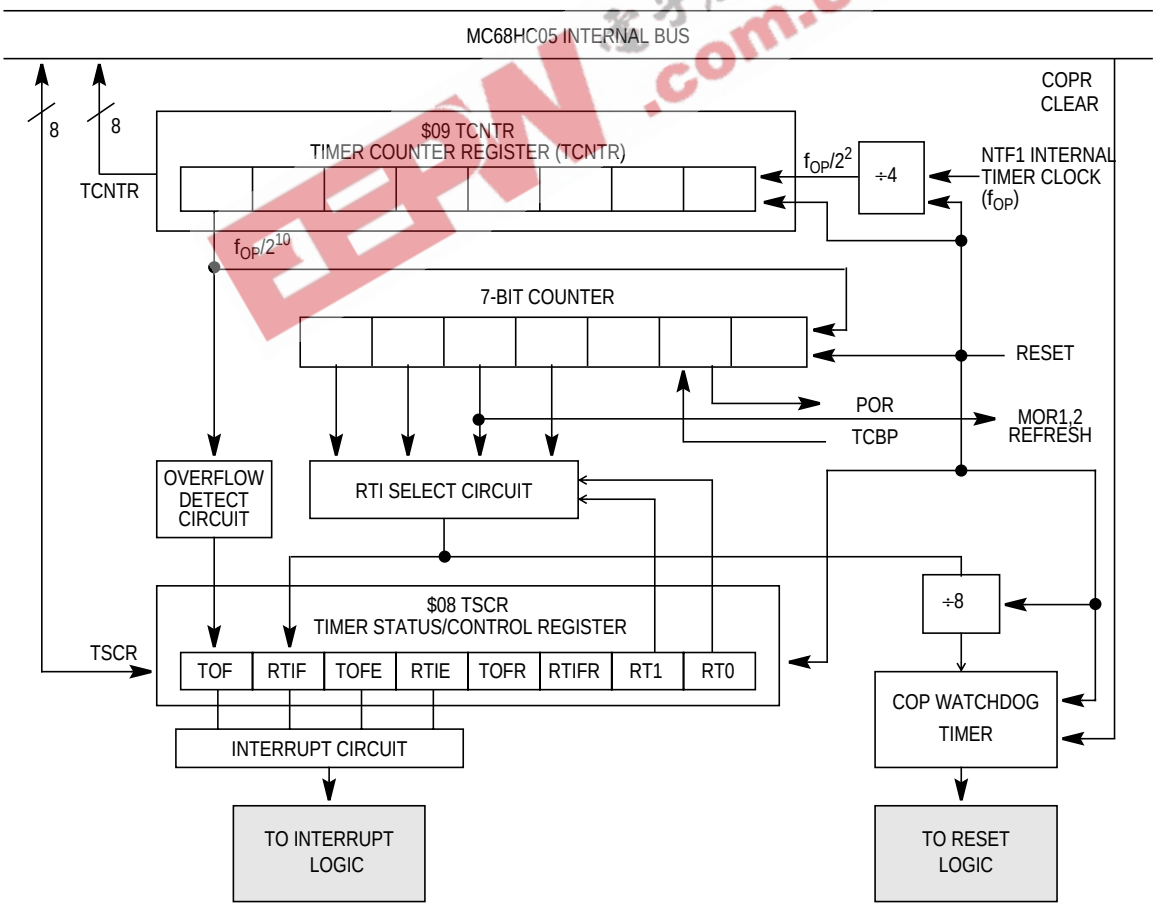


Figure 8-1. Timer Block Diagram

8.3 Timer Registers

The 8-bit timer contains two registers:

- Timer counter register
- Timer status/control register

8.3.1 Timer Counter Register

The timer counter register is a read-only register that contains the current value of the 8-bit ripple counter at the beginning of the timer chain. This counter is clocked at f_{OP} divided by 4 and can be used for various functions including a software input capture. Extended time periods can be attained using the TOF function to increment a temporary RAM storage location thereby simulating a 16-bit (or more) counter. The value of each bit of the TCNTR is shown in Figure 8-2. This register is cleared by reset.

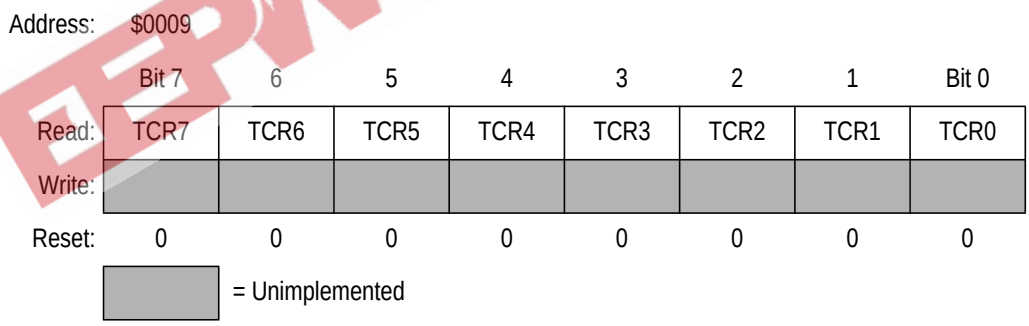


Figure 8-2. Timer Counter Register (TCNTR)

8.3.2 Timer Status/Control Register

The TSCR contains the timer interrupt flag, the timer interrupt enable bits, and the real-time interrupt rate select bits. Bit 2 and bit 3 are write-only bits that read as logical 0s. **Figure 8-3** shows the value of each bit in the TSCR following reset.

Address: $0008

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TOF	RTIF	TOIE	RTIE	0	0	RT1	RT0
Write:					TOFR	RTIFR		
Reset:	0	0	0	0	0	0	1	1

= Unimplemented

Figure 8-3. Timer Status/Control Register (TSCR)

TOF — Timer Overflow Bit

The TOF is a read-only flag bit that is set when the 8-bit ripple counter rolls over from $FF to $00. A timer interrupt request is generated if TOF is set when TOIE is also set. The TOF flag bit is reset by writing a logical 1 to the TOFR acknowledge bit. Writing to the TOF flag bit has no effect on its value. This bit is cleared by reset.

RTIF — Real-Time Interrupt Flag Bit

The RTIF is a read-only flag bit that is set when the output of the chosen (one-of-four selection) real-time interrupt stage goes active. A timer interrupt request is generated if RTIF is set when RTIE is also set. The RTIF flag bit is reset by writing a logical 1 to the RTIFR acknowledge bit. Writing to the RTIF flag bit has no effect on its value. This bit is cleared by reset.

TOIE — Timer Overflow Interrupt Enable Bit

The TOIE is an enable bit that allows generation of a timer interrupt. When the TOIE enable bit is set, the TIMER Interrupt is generated when the TOF flag bit is set. This bit is cleared by reset.

RTIE — Real-Time Interrupt Enable Bit

The RTIE is an enable bit that allows the generation of a timer interrupt. When the RTIE enable bit is set and the RTIF flag bit is set, the timer interrupt is generated. The RTIE bit is cleared by reset.

TOFR — Timer Overflow Acknowledge Bit

The TOFR is an acknowledge bit that resets the TOF flag bit. Writing a logical 1 to the TOFR clears the TOF flag bit. Reading the TOFR always returns a logical 0. This bit is unaffected by reset.

RTIFR — Real-Time Interrupt Acknowledge Bit

The RTIFR is an acknowledge bit that resets the RTIF flag bit. Writing a logical 1 to the RTIFR clears the RTIF flag bit. Reading the RTIFR always returns a logical 0. This bit is unaffected by reset.

RT1:RT0 — Real-Time Interrupt Rate Select Bit

The RT0 and RT1 control bits select one-of-four taps for the real-time interrupt circuit. **Table 8-1** shows the available interrupt rates with several f_{OP} values. Both the RT0 and RT1 control bits are set by reset, 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 can be missed or an additional RTIF can be generated. To avoid problems, the COP should be cleared just prior to changing RTI taps.

Table 8-1. RTI Rates and COP Reset Times

RT1 and RT0	RTI Rate	RTI Period (f_{OP} = 2 MHz)	COP Timeout Period (± 1 RTI Period)	Minimum COP Timeout Period (f_{OP} = 2 MHz)
0 0	$f_{OP} \div 2^{14}$	8.2 ms	8 x RTI period	57.3 ms
0 1	$f_{OP} \div 2^{15}$	16.4 ms	8 x RTI period	114.7 ms
1 0	$f_{OP} \div 2^{16}$	32.8 ms	8 x RTI period	229.4 ms
1 1	$f_{OP} \div 2^{17}$	65.5 ms	8 x RTI period	458.8 ms

8.4 COP Watchdog Timer

The computer operating properly (COP) watchdog timer function is implemented on this device by using the output of the RTI circuit and further dividing it by eight. The minimum COP reset times are listed in **Table 8-1**.

If the COP circuit times out, an internal reset is generated and the reset vector is fetched. Preventing a COP timeout is done by writing a logical 0 to the COPC bit at address $03F0 as shown in **Figure 8-4**. The COPR register is shared with a user EEPROM byte. This address location is not affected by any reset signals. Reading this location returns the user EEPROM byte. When the COPC is cleared, only the final four bits used to count eight RTI cycles are cleared. The COP watchdog timer can be enabled/disabled by a mask option.

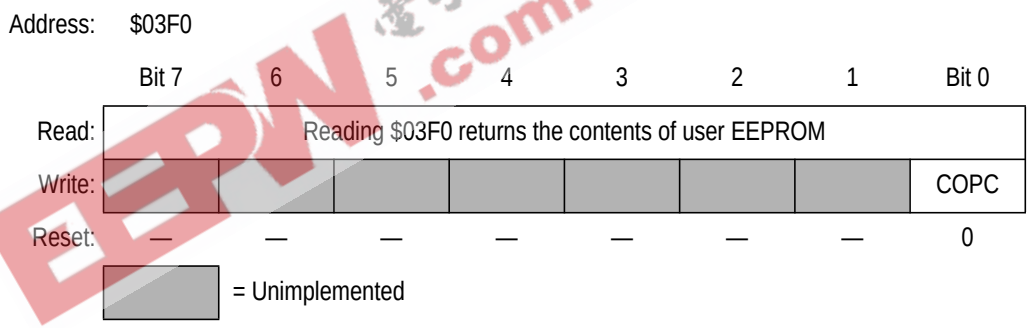


Figure 8-4. COPR Watchdog Timer Location

8.5 Operating During Stop Mode

The timer system is cleared when going into stop mode. When STOP is exited by an external interrupt or an external RESET, the internal oscillator resumes, followed by a 16- or 4064-cycle internal processor oscillator stabilization delay. The timer system counter is then cleared and operation resumes. If the STOP instruction is disabled by mask option to create the halt mode, the effects on the timer are as described in **8.6 Operating During Wait Mode**.

8.6 Operating During Wait Mode

The CPU clock halts during wait mode, but the timer remains active. If interrupts are enabled, a timer interrupt or custom periodic interrupt causes the processor to exit wait mode.

8-Bit Timer

Section 9. Personality EEPROM (PEEPROM)

9.1 Contents

9.2 Introduction

The MC68HC05K3 contains a 128-bit personality EEPROM (PEEPROM) for storage of variables or user data. These 128 bits are provided as a simple EEPROM array and control logic that requires serial reading of the data. The PEEPROM may be accessed via software programmed into the user ROM through two registers that directly interface with the PEEPROM array. The actual implementation of the software varies depending on customer requirements. The PEEPROM array is arranged as 16 bytes (rows) with a separate column select for each bit (column) in a byte. The column select connects the bit to a

single sense amplifier as shown in the block diagram of the PEEPROM module in **Figure 9-1**.

An on-chip charge pump is provided to allow programming and erasure of the personality EEPROM if the supply voltage to the V_{DD} pin is at least 2.7 Vdc.

NOTE: *Programming and erasure of the personality EEPROM may only be performed if V_{DD} greater than or equal to 2.7 Vdc.*

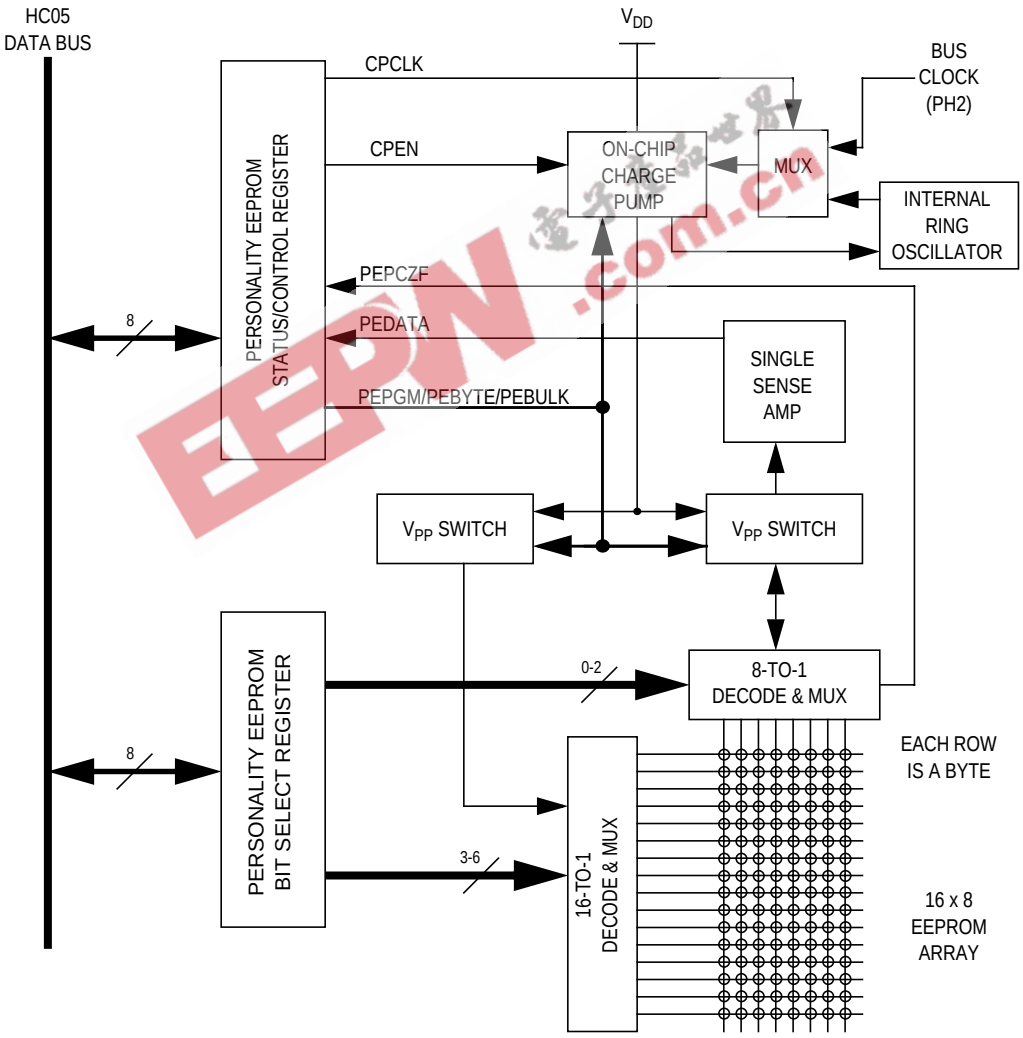


Figure 9-1. Personality EEPROM Block Diagram

9.3 PEEPROM Registers

Two register locations are used to support the EEPROM array. These are the bit select and status/control registers.

9.3.1 PEEPROM Bit Select Register

The PEEPROM bit select register (PEBSR) is located at $000E and contains the enable signals for the rows and columns to access the bits in the EEPROM array. **Figure 9-2** shows the placement of these bits. The output of this register is connected to two decoders, one for the array column and one for the array row.

A byte in the PEEPROM is defined by the upper four bits in the 7-bit address in the PEBSR (PEB3–PEB6) and the bit within that byte is defined by the lower three bits in the 7-bit address in the PEBSR (PEB0–PEB2). The upper bit in the PEBSR (PEB7) may be used as a storage location. All of the bits in the PEBSR register are cleared by reset.

Address: $000E

	Byte (Row) of PEEPROM				Bit (Column in Byte (Row) of PEEPROM		
	3	2	1	0	2	1	0
	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PEB7	PEB6	PEB5	PEB4	PEB3	PEB2	PEB1	PEB0
Reset:	0	0	0	0	0	0	0	0

Figure 9-2. PEEPROM Bit Select Register (PEBSR)

9.3.2 PEEPROM Status/Control Register

The PEEPROM status/control register (PESCR) is located at $000F and contains five user bits, as shown in **Figure 9-3**. Bit 1 is unimplemented and always reads as a logic 0. The states of all bits except PEPCZF and PEDATA are cleared by reset. The PEPCZF is set by reset; and the state of the PEDATA bit following reset is dependent on the stored data in bit 0 of the PEEPROM array.

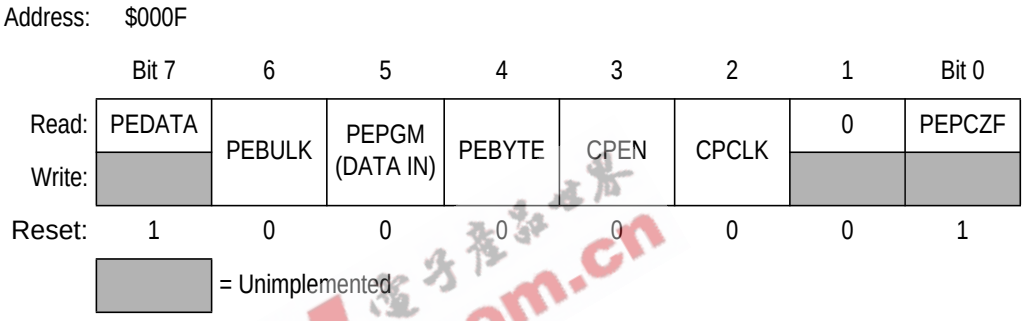


Figure 9-3. PEEPROM Status/Control Register (PESCR)

PEPCZF — PEEPROM Column Zero Flag Bit

The PEPCZF is a flag bit that is set to a logical 1 when the first column (COL0) of the EEPROM array is selected. If any other column is selected, the PEPCZF flag bit is cleared. This flag bit can be used to reduce the software code required to access one byte of the PEEPROM. The PEPCZF is set following a reset, since the first column is selected by the reset of the PEBSR. The software code given in **Table 9-1** is suggested for reading one byte from the PEEPROM.

Table 9-1. Software to Read PEEPROM

```
pebsr     equ     $000e
pescr     equ     $000f
ram       equ     $000c

          lda     #$xy              ; xy is base addresses and should start on
          sta     pebsr             ; a first column (i.e., $00, $08, $10, $18, etc).
          clr     ram               ; clear ram location used for final result

peep_rd   rol     pescr             ; c = pedata (c = carry bit)
          ror     ram               ; ram = c
          inc     pebsr             ; go to next bit in array.
          brclr   0,pescr,peep_rd   ; care data here, loop until all bytes read
                                    ; peep_rd loop ends when PEPCZF = 1.
                                    ; At end of loop, ram contains one row of PEEP data.
```

CPCLK — Charge Pump Clock Source Bit

The CPCLK bit is a read/write bit that controls the source of the clock for the charge pump. When the CPCLK bit is set, the charge pump is driven by the PH2 bus clock. When the CPCLK bit is cleared, the charge pump is driven from an internal ring oscillator. The CPCLK bit is cleared when the device is in reset.

In systems where the desired PH2 clock rate is below 1 MHz, the CPCLK bit should be cleared to enable the internal ring oscillator. Otherwise, the charge pump does not attain sufficient program/erase voltage because the clock source is too slow.

CPEN — Charge Pump Enable Bit

The CPEN bit is a read/write bit to control the on-chip charge pump for programming and erasure of the personality EEPROM. **This charge pump is only intended for use at V_{DD} supply voltages greater than or equal to 2.7 Vdc.** The charge pump is activated when both the CPEN bit is set and one of the program or erase bits is also set (PEPGM, PEBYTE, or PBULK). The charge pump supplies the required programming voltage to the personality EEPROM array. Once activated, and after startup time t_{CP}, the charge pump continues to operate until all the program and erase bits are cleared.

NOTE: *If the personality EEPROM is read while the CPEN bit is set, the data is unknown.*

The charge pump must always be used to program or erase bits in the personality EEPROM. The CPEN bit is cleared when the device is in reset.

NOTE: *Setting the CPEN bit can activate the charge pump. However, all the PEPGM, PEBYTE, PEBULK, and CPEN bits must be cleared to de-activate the charge pump. If the charge pump is left running, the overall device I_{DD} current increases.*

PEBYTE — PEEPROM Byte Erase Bit

The PEBYTE bit is a read/write bit to control the switches that apply the internally provided charge pump programming voltage to a row in the PEEPROM array that is to be erased. When the PEBYTE bit is set to a logical 1, a logical 0 is stored to all bits in the same row of the PEEPROM array, as specified by the upper four bits of the 7-bit address in the PEBSR.

The PEBYTE bit should only be set if the PEPGM and PEBULK bits are cleared. If both the PEBYTE and PEBULK bits are set, the PEEPROM is bulk erased. The PEBYTE bit is cleared when the device is reset.

PEPGM — PEEPROM Program Control Bit

The PEPGM bit is a read/write bit to control the switches that apply the internally provided charge pump programming voltage to the device in the PEEPROM array that is to be programmed. When the PEPGM bit is set to a logical 1, a logical 1 is stored to the PEEPROM array element specified by the address in the PEBSR. Since the state of the PEPGM bit determines the state of the programmed bit in the PEEPROM array, the PEPGM bit is similar to a DATA IN bit.

The PEPGM bit should be set only if the PEBYTE and PEBULK bits are cleared. The PEPGM bit is cleared when the device is reset.

NOTE: *Only one of the PEPGM, PEBYTE, or PEBULK bits should be set at any one time.*

Always clear the PEPGM bit before altering the addressing bits in the PEBSR. Otherwise, intermediate locations may be affected if the programming voltage is present.

PEBULK — PEEPROM Bulk Erase Bit

The PEBULK bit is a read/write bit to control the switches that apply an internally provided programming voltage to all the bits in the PEEPROM array that are to be erased. When the PEBULK bit is set to a logical 1, a logical 0 is stored to all bits of the PEEPROM array regardless of the bit address specified in the PEBSR.

The PEBULK bit should only be set if the PEBYTE and PEPGM bits are cleared. If both the PEBYTE and PEBULK bits are set, the personality EEPROM is bulk erased. The PEBULK bit is cleared when the device is reset.

PEDATA — PEEPROM Data Bit

The PEDATA bit is a read-only bit that reflects the state of the PEEPROM sense amplifier. The state of the PEDATA bit is only meaningful when the PEBYTE, PEPGM, PEBULK, and CPEN control bits are all 0. The state of the PEDATA bit following a reset is dependent on the stored data in bit 0 of the PEEPROM array.

9.4 PEEPROM Programming

The PEEPROM can be programmed using a Motorola programmer or in the user application **if the V_{DD} supply source is at least 2.7 Vdc**. In the latter case, the programming software must be provided in the user ROM and use some external pins in either a serial or parallel method for data transfer and/or access. Each bit of the PEEPROM can be programmed using this step-by-step procedure:

1. Write the desired bit location to be programmed into the PEBSR located at $000E.
2. Set the PEPGM and CPEN bits in the PESCR located at $000F.
3. Wait for a t_{EPGM} time delay.
4. Clear the PEPGM and CPEN bits.

The PEEPROM is then ready to be set up for another bit of data for programming.

The programming of a PEEPROM bit only requires access of that bit through the PEBSR followed by setting the PEPGM and CPEN bits in the PESCR. Do not access any bits that are to be left unprogrammed (erased) until all the PEPGM, PEBYTE, PEBULK, and CPEN bits in the PESCR are cleared. Always clear the PEPGM, PEBYTE, PEBULK, and CPEN bits before altering the PEBSR.

9.5 PEEPROM Read Access

The contents of the PEEPROM are read in this sequence:

1. Write the desired bit location to be read into the PEBSR located at $000E.
2. Read the state of the PEDATA bit in the PESCR located at $000F.
3. Store the state of the PEDATA bit into RAM or a register.
4. Select another bit by changing the PEBSR.
5. Continue reading and storing the PEDATA bit states until all the required PEEPROM data has been accessed.

Reading the PEEPROM is easiest when each row in the PEEPROM array is mapped to contain one byte of data. Selecting a column zero bit selects the first bit in the row; and incrementing the PEEPROM bit select register (PEBSR) selects the next (column 1) bit from the same row. Incrementing the PEBSR seven more times selects the remaining bits of the row and carries over to select column zero of the next row, thereby setting the column zero flag, PEPCZF in the PESCR. The number of increments per row can be controlled by looping on a test of the PEPCZF flag bit.

The complete array can be easily accessed by starting with $007F for the PEBSR and decrementing the PEBSR after each access of the PEDATA bit. The decrement sequence can end when the contents of the PEBSR are 0.

NOTE: *One byte of data from the PEEPROM can be re-created in the PEBSR itself. This can be done if the read routine builds the 8-bit data byte in the index register or the accumulator and then transfers that result to the PEBSR when completed. Subsequent reads of the PEBSR quickly yield that retrieved data byte.*

9.6 PEEPROM Serial Programming

The MC68HC05K3 can be programmed, read, or bulk erased in a serial fashion using the $\overline{\text{RESET}}$, PB0 and $\overline{\text{IRQ}}$ pins in PEEPROM serial programming mode.

- The $\overline{\text{RESET}}$ pin is used to begin and end sequences and must be able to toggle from 0 Vdc to V_{DD}.
- PB0 serves as the data pin and must be able to toggle from 0 Vdc to V_{DD}.
- $\overline{\text{IRQ}}$ is used as the clock/select line and must be able to toggle from 0 Vdc to 2 x V_{DD}.

NOTE: *When not in serial programming mode, do not operate the device with more than V_{DD} on the $\overline{IRQ}$ pin.*

9.6.1 Serial Programming Connections

The required schematic considerations are shown in **Figure 9-4**. The serial programming connections can be shared with the application if certain considerations are met.

1. The application circuitry connected to the PB0 pin must allow this pin to be used as an input and not driven from some active source within the application other than the MC68HC05K3. This pin is driven to a logic high or low level by either an external source or by the MC68HC05K3 itself.
2. The application circuitry connected to the $\overline{\text{IRQ}}$ pin must be capable of being driven from an external source to a voltage, V_{SELCKH}.
3. The application circuitry connected to the $\overline{\text{RESET}}$ pin must allow this pin to be used as an input and not driven from some active source within the application other than the MC68HC05K3.
4. To use the on-chip charge pump for programming/erasure, the application circuitry must be capable of being supplied with a V_{DD} source of at least 2.7 Vdc.
5. The diagram in **Figure 9-4** shows the 3-pin RC oscillator connections. This circuit also works with the 2-pin crystal and RC oscillators and the PB1/OSC3 pin can be left unconnected in those cases.

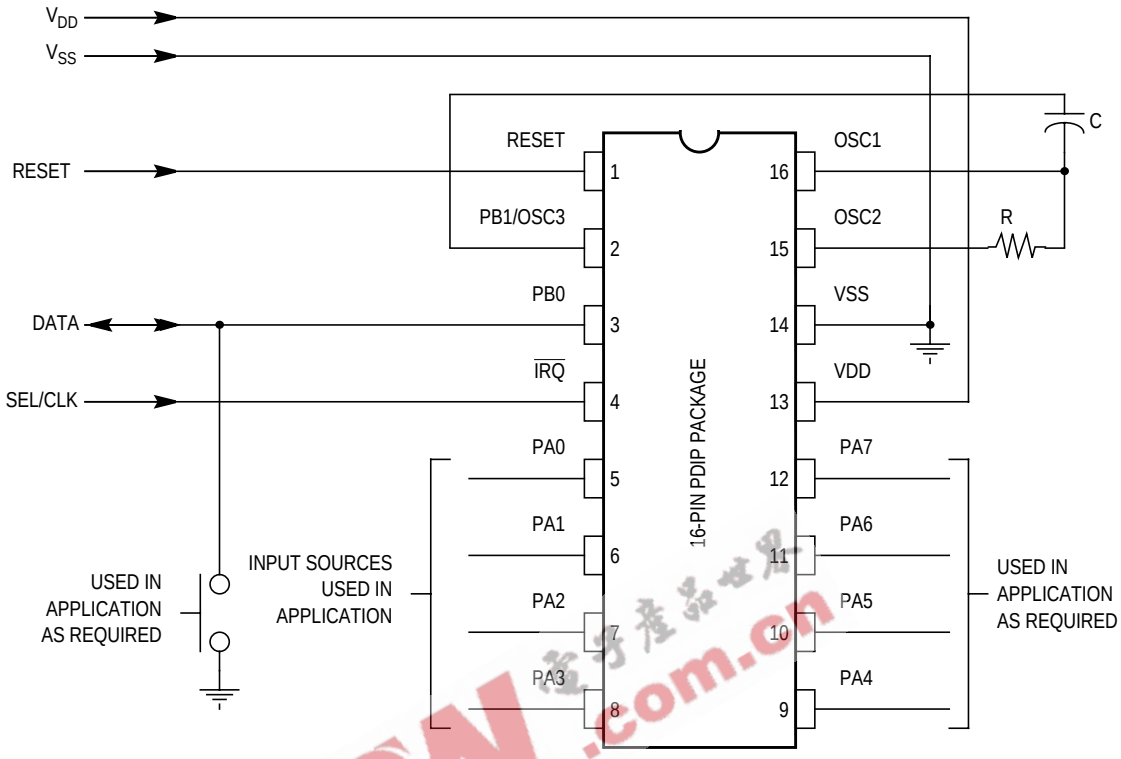


Figure 9-4. Serial Programming Connections

9.6.2 Multiple Devices in Serial Program Mode

If the preceding rules in **9.6.1 Serial Programming Connections** are met, multiple MC68HC05K3 devices can be connected in parallel during serial programming. All the parallel devices can share the same power supplies for the V_{DD} and V_{SS}; and they may all share the same DATA and RESET signals. The only signal that must be routed individually to each device is the SEL/CLK signal. Programming time can be reduced by setting the data for each unit and then clocking its SEL/CLK low and remaining low until the data has been received by all other devices. Then raise the SEL/CLK line high for all units after the required programming time.

NOTE: *Direct connection of the $\overline{RESET}$ pin to the V_{DD} supply should be avoided because as an internal reset source, such as a COP watchdog reset or an illegal address, reset turns on an internal pulldown device connected*

to the $\overline{RESET}$ pin. This device may be capable of heavily loading the V_{DD} supply source.

9.6.3 PEEPROM Serial Programming Mode Entry

The sequence for entry into the personality EEPROM serial programming mode to program, erase, or read data is:

1. Connect all parallel device modules.
2. Apply 0 Vdc to the DATA, RESET, and SEL/CLK signal lines.
3. Apply a voltage V_{DDS} to the VDD suppy line.
4. Apply V_{SELCKH} to the SEL/CLK signal line.
5. Wait for the initial startup delay (16 or 4064 internal PH2 processor clock cycles, depending on mask option). Then raise the RESET signal line to V_{DDS}.
6. Do not raise the DATA signal line to V_{DDS} until after a time, t_{RDH}.
7. During the select sequence, the SEL/CLK signal is toggled between 0 Vdc and V_{SELCKH}. The duration between transitions is specified by t_{CLKHI} and t_{CLKLO}.

The device is now monitoring the $\overline{IRQ}$ and PB0 pins to determine the type of procedure to execute. The state of the PB0 pin is examined following each high-to-low transition of the $\overline{IRQ}$ pin, as shown in **Figure 9-5**.

The state of the PB0 pin during the second and third falling edges of SEL/CLK determines the type of operation. If the operation is to program or verify the contents of the personality EEPROM, an additional 128 falling edges are required to store or retrieve data. The type of internal operation to be executed is defined in **Table 9-2**. The state of PB0 on the first falling edge of SEL/CLK is of no consequence; however, PB0 must remain low for a minimum of 10 internal clock cycles, t_{RDH}, following negation of $\overline{RESET}$.

The time delays required for various PEEPROM serial program mode operations are given in **Table 9-3**.

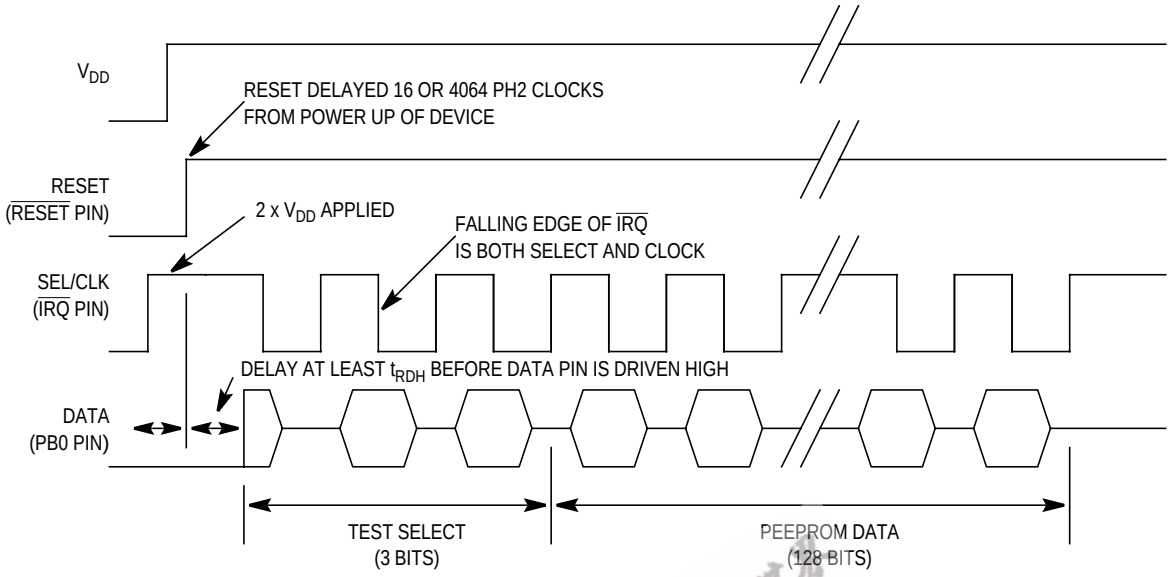


Figure 9-5. PEEPROM Serial Programming Mode Data Format

Table 9-2. PEEPROM Serial Programming Mode Operations

First Bit	Second Bit	Third Bit	Internal Test Operation
X	0	0	Serial program personality EEPROM contents
X	0	1	Serial retrieval of personality EEPROM contents
X	1	0	Unassigned
X	1	1	Bulk erasure of EEPROM contents

9.7 Serial Programming Sequence

The programming of the internal personality EEPROM always begins at bit address 00 of the 128-bit array. The signal timing on the $\overline{IRQ}$ and PB0 pins is shown in **Figure 9-6**, **Figure 9-7**, and **Figure 9-8**. The personality EEPROM is serially programmed by using this sequence:

1. Follow the serial programming mode entry routine as specified in **9.6.3 PEEPROM Serial Programming Mode Entry**.
2. Keep the DATA signal at 0 Vdc throughout the select sequence.
3. Clock the SEL/CLK signal line to 0 Vdc and back to V_{SELCKH} (first select bit).
4. Clock the SEL/CLK signal line to 0 Vdc and back to V_{SELCKH} (second select bit).
5. Clock the SEL/CLK signal line to 0 Vdc and back to V_{SELCKH} (third select bit).
6. The device is now ready to receive the data to be programmed into the 128 bits of the personality EEPROM. During the next 128 clocks of the SEL/CLK signal line, the data to be stored into each bit (starting with location $00) must be present on the DATA signal line prior to the falling edge of the SEL/CLK signal line to 0 Vdc. The time that the SEL/CLK signal line stays at 0 Vdc is determined by the EEPROM programming time (t_{EPGM}).
7. On the 128th data bit, rather than drive the SEL/CLK signal high to complete the programming sequence, drive the $\overline{RESET}$ pin to 0 V after a time, t_{EPGM}. This must be done to prevent the part from entering an unknown state. If all 128 bits do not need to be programmed, the $\overline{RESET}$ pin may be driven to 0 V after the last bit has been programmed. This will complete the programming sequence.
8. While $\overline{RESET}$ is held low, the pins can be conditioned for the next sequence.

This completes the serial programming sequence. The device can now be verified by going to the serial data readout sequence or bulk erased by going to the bulk erase sequence.

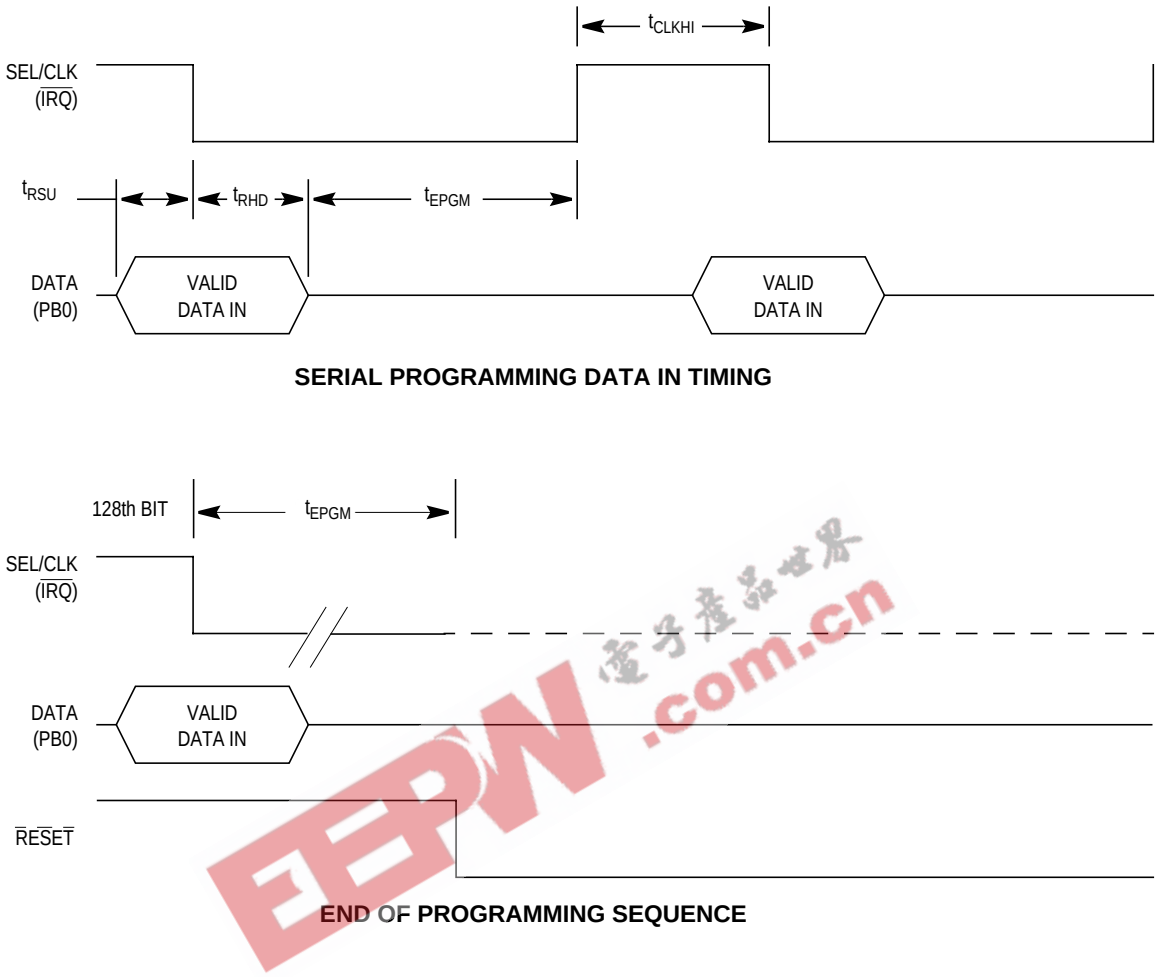


Figure 9-6. PEEPROM Serial Programming Data In Timing

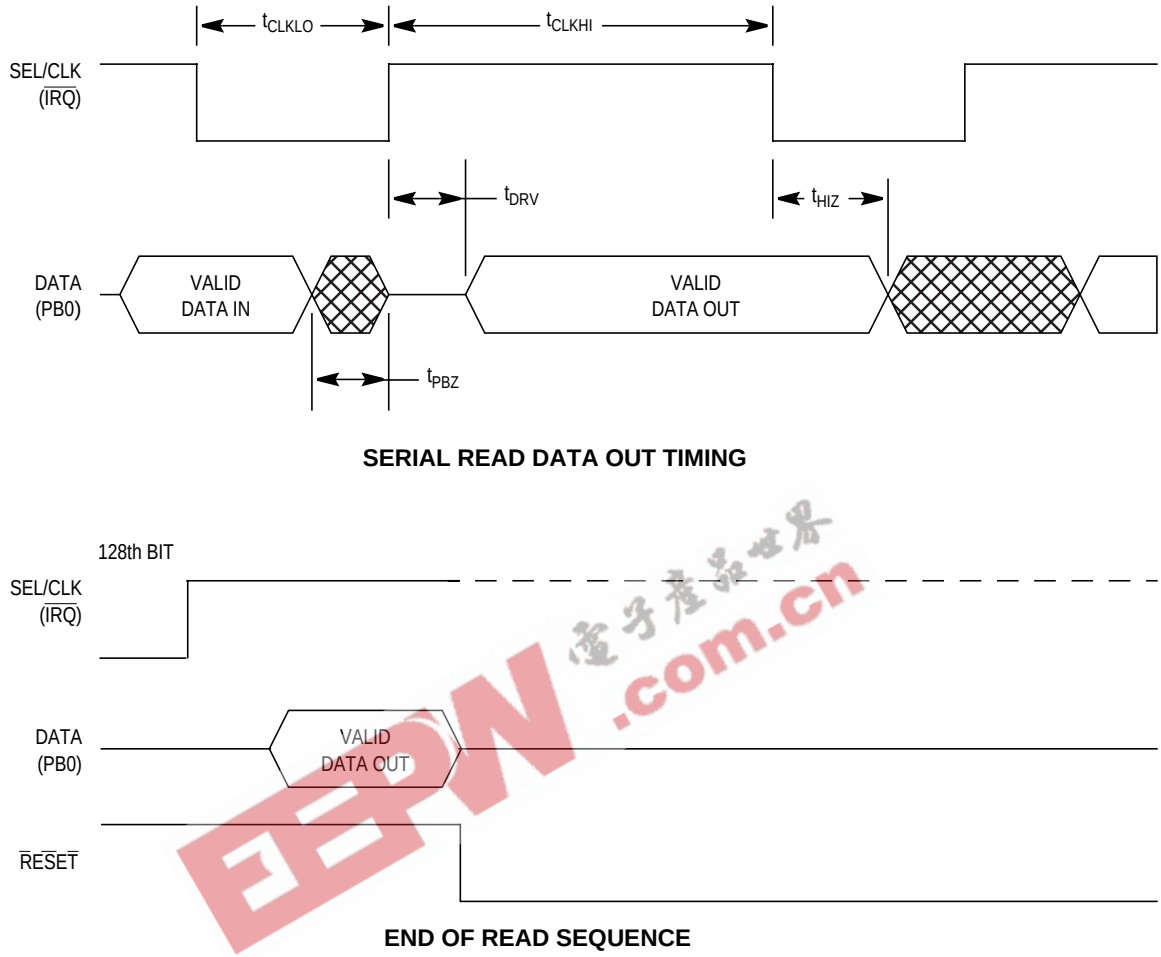


Figure 9-7. PEEPROM Serial Read Data Out Timing

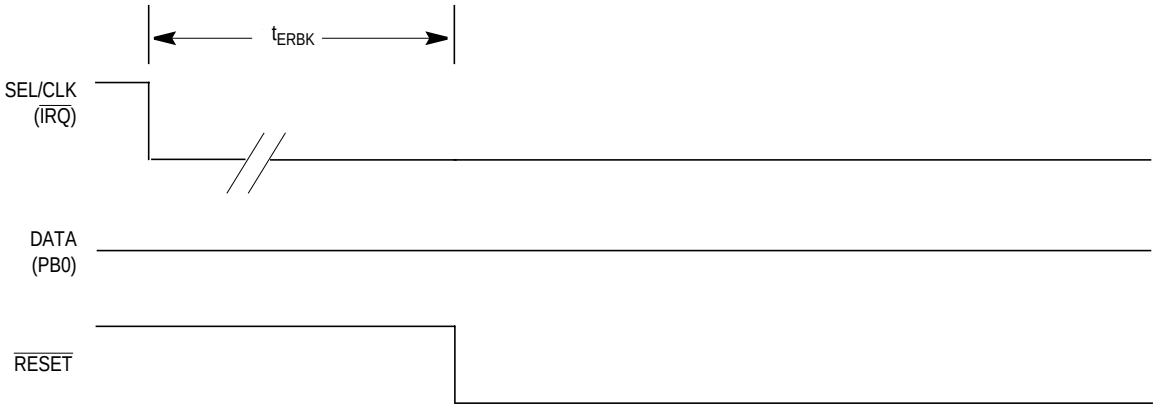


Figure 9-8. Bulk Erase Sequence

Table 9-3. Internal Test Time Delays

Symbol	Description	Min	Max	Time
V_{DDS}	Serial mode V_{DD} voltage	2.7	5.5	V
V_{SELCKH}	SEL/CLK high voltage	1.5 x V_{DD}	2.0 x V_{DD}	V
t_{RDH}	RESET rising edge to 1st rising edge of data	10	—	f_{OP} cycles
t_{CLKHI}	SEL/CLK active high time	32	—	f_{OP} cycles
t_{CLKLO}	SEL/CLK active low time (non-programming/erasing)	32	See Note	f_{OP} cycles
t_{RSU}	Read data in setup before falling edge of SEL/CLK	0	—	f_{OP} cycles
t_{RHD}	Read data in hold after falling edge of SEL/CLK	16	—	f_{OP} cycles
t_{EPGM}	Programming time for EEPROM cell	10	15	ms
t_{RCLK}	RESET rising edge to falling edge of SEL/CLK	5	—	f_{OP} cycles
t_{DRV}	Data output drive after rising edge of SEL/CLK	4	8	f_{OP} cycles
t_{HIZ}	Data output hi-Z after rising edge of SEL/CLK	—	8	f_{OP} cycles
t_{PBZ}	Data hi-Z before rising edge of SEL/CLK	0	—	f_{OP} cycles
t_{ERBK}	Bulk erase time for EEPROM array	30	30	ms

Note:
If computer operating properly (COP) watchdog timer is enabled through mask option, the maximum allowable interval between any two successive high-to-low transitions of SEL/CLK is 7 x 2^{17} f_{OP} cycles.

9.8 Serial Data Readout Sequence

The read sequence of the internal personality EEPROM always begins at bit address 00 of the 128-bit array. The signal timing on the $\overline{IRQ}$ and PB0 pins is shown in **Figure 9-7**. The personality EEPROM can be serially read out by using this sequence:

1. Follow the serial programming mode entry routine as specified in **9.6.3 PEEPROM Serial Programming Mode Entry**.
2. Keep the DATA signal at 0 Vdc.
3. Clock the SEL/CLK signal line to 0 Vdc and back to V_{SELCKH} (first select bit).
4. Clock the SEL/CLK signal line to 0 Vdc and back to V_{SELCKH} (second select bit).
5. Raise the DATA signal line to V_{DDS}.
6. Clock the SEL/CLK signal line to 0 Vdc and back to V_{SELCKH} (third select bit).
7. The device is now ready to transmit the data that has been programmed or erased into the 128 bits of the personality EEPROM. During the next 128 clocks of the SEL/CLK signal line, the stored data for each bit (starting with location $00) is presented on the DATA signal line prior to the falling edge of the SEL/CLK signal line to 0 Vdc. The data is valid within t_{DRV} internal cycles after the rising edge of SEL/CLK and is valid when the SEL/CLK signal line falls for a time, t_{HIZ}.
8. On the 128th data bit, rather than drive the SEL/CLK signal low to complete the read sequence, drive the $\overline{RESET}$ pin to 0 V. This must be done to prevent the part from entering an unknown state. If all 128 bits do not need to be verified, the $\overline{RESET}$ pin may be driven to 0 V after the last bit has been verified.
9. While $\overline{RESET}$ is held low, the pins can be conditioned for the next sequence.

This completes the serial data readout sequence, and the device can now be completely erased by going to the bulk erase sequence if the verification sequence fails to read the desired data.

9.9 Serial Bulk Erase Sequence

The signal timing on the IRQ and PB0 pins is shown in **Figure 9-8**. The personality EEPROM can be bulk erased by using this sequence:

1. Follow the serial programming mode entry routine as specified in **9.6.3 PEEPROM Serial Programming Mode Entry**.
2. Keep the DATA signal at 0 Vdc.
3. Clock the SEL/CLK signal line to 0 Vdc and back to V_{SELCKH} (first select bit).
4. Raise the DATA signal to V_{DDS}.
5. Clock the SEL/CLK signal line to 0 Vdc and back to V_{SELCKH} (second select bit).
6. Clock the SEl/CLK signal line to 0 Vdc. Do not bring the SEL/CLK signal line back to V_{SELCKH} as in other select sequences (third select bit). This starts the bulk erase sequence.
7. After a time, t_{ERBK}, drive the $\overline{\text{RESET}}$ pin to 0 V to complete the bulk erase sequence. This must be done to prevent the part from entering an unkown state.
8. While $\overline{\text{RESET}}$ is held low, the pins can be conditioned for the next sequence.

This completes the bulk erase sequence. The device can now be programmed by going to the serial programming sequence or verified by going to the serial verification sequence.

Section 10. Instruction Set

10.1 Contents

10.2 Introduction

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

10.3 Addressing Modes

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

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

10.3.1 Inherent

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

10.3.2 Immediate

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

10.3.3 Direct

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

10.3.4 Extended

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

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

10.3.5 Indexed, No Offset

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

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

10.3.6 Indexed, 8-Bit Offset

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

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

10.3.7 Indexed,16-Bit Offset

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

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

10.3.8 Relative

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

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

10.4 Instruction Types

The MCU instructions fall into the following five categories:

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

10.4.1 Register/Memory Instructions

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

Table 10-1. Register/Memory Instructions

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

10.4.2 Read-Modify-Write Instructions

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

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

Table 10-2. Read-Modify-Write Instructions

Instruction	Mnemonic
Arithmetic Shift Left (Same as LSL)	ASL
Arithmetic Shift Right	ASR
Bit Clear	BCLR(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)

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

10.4.3 Jump/Branch Instructions

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

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

Table 10-3. Jump and Branch Instructions

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

10.4.4 Bit Manipulation Instructions

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

Table 10-4. Bit Manipulation Instructions

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

10.4.5 Control Instructions

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

Table 10-5. Control Instructions

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

10.5 Instruction Set Summary

Table 10-6. Instruction Set Summary (Sheet 1 of 6)

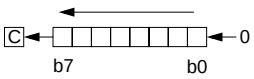
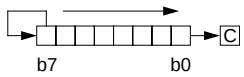
Source Form	Operation	Description	Effect on CCR					Address Mode	Opcode	Operand	Cycles
			H	I	N	Z	C				
ADC #*opr* ADC *opr* ADC *opr* ADC *opr*,X ADC *opr*,X ADC ,X	Add with Carry	A ← (A) + (M) + (C)	↕	—	↕	↕	↕	IMM DIR EXT IX2 IX1 IX	A9 B9 C9 D9 E9 F9	ii dd hh ll ee ff ff	2 3 4 5 4 3
ADD #*opr* ADD *opr*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

Table 10-6. Instruction Set Summary (Sheet 2 of 6)

Source Form	Operation	Description	Effect on CCR					Address Mode	Opcode	Operand	Cycles
			H	I	N	Z	C				
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 10-6. Instruction Set Summary (Sheet 3 of 6)

Source Form	Operation	Description	Effect on CCR					Address Mode	Opcode	Operand	Cycles
			H	I	N	Z	C				
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 10-6. Instruction Set Summary (Sheet 4 of 6)

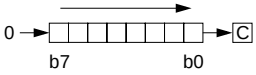
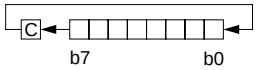
Source Form	Operation	Description	Effect on CCR					Address Mode	Opcode	Operand	Cycles
			H	I	N	Z	C				
JSR *opr* JSR *opr* JSR *opr*,X JSR *opr*,X JSR ,X	Jump to Subroutine	PC ← (PC) + n (n = 1, 2, or 3) Push (PCL); SP ← (SP) – 1 Push (PCH); SP ← (SP) – 1 PC ← Effective Address	—	—	—	—	—	DIR EXT IX2 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 10-6. Instruction Set Summary (Sheet 5 of 6)

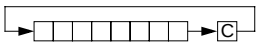
Source Form	Operation	Description	Effect on CCR					Address Mode	Opcode	Operand	Cycles
			H	I	N	Z	C				
ROR *opr* RORA RORX ROR *opr*,X ROR ,X	Rotate Byte Right through Carry Bit		—	—	↕	↕	↕	DIR INH INH IX1 IX	36 46 56 66 76	dd ff 	5 3 3 6 5
RSP	Reset Stack Pointer	SP ← $00FF	—	—	—	—	—	INH	9C		2
RTI	Return 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 10-6. Instruction Set Summary (Sheet 6 of 6)

Source Form	Operation	Description	Effect on CCR					Address Mode	Opcode	Operand	Cycles
			H	I	N	Z	C				
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 10-7. Opcode Map

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

INH = InherentREL = Relative
IMM = ImmediateIX = Indexed, No Offset
DIR = DirectIX1 = Indexed, 8-Bit Offset
EXT = ExtendedIX2 = Indexed, 16-Bit Offset

LSB of Opcode in Hexadecimal

MSB / LSB	0
0	5 BRSET0 3 DIR

MSB of Opcode in Hexadecimal

Number of Cycles
Opcode Mnemonic
Number of Bytes/Addressing Mode

Section 11. Electrical Specifications

11.1 Contents

11.2 Introduction

This section contains electrical and timing specifications.

11.3 Maximum Ratings

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

The MCU contains circuitry to protect the inputs against damage from high static voltages; however, do not apply voltages higher than those shown in the table below. Keep V_{IN} and V_{OUT} within the range $V_{SS} \le (V_{IN}$ or $V_{OUT}) \le V_{DD}$. 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
Input voltage ($\overline{IRQ}$ pin only)	V_{IN}	2 x V_{DD}	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* ***11.6 5.0-Volt DC Electrical Characteristics*** *and* ***11.7 3.0-Volt DC Electrical Characteristics*** *for guaranteed operating conditions.*

11.4 Operating Range

Characteristic	Symbol	Value	Unit
Operating temperature range MC68HC05K3 (standard) MC68HC05K3 (extended)	T_A	T_L to T_H 0 to +70 –40 to +85	°C
Supply voltage range for internal charge pump operation	V_{DDCP}	2.7 to 5.5	V

11.5 Thermal Characteristics

Characteristic	Symbol	Value	Unit
Thermal resistance PDIP SOIC	θ_{JA}	 100 140	°C/W

11.6 5.0-Volt DC Electrical Characteristics

Characteristic[1]	Symbol	Min[2]	Typ[3]	Max	Unit
Output high voltage (I_{Load} = –0.8 mA) PA7–PA0, PB1/OSC3, PB0	V_{OH}	V_{DD} –0.8	—	—	V
Output low voltage PA3–PA0, PB1/OSC3, PB0 (I_{Load} = 1.6 mA) PA7–PA4 (I_{Load} = 8.0 mA)	V_{OL}	— —	— —	0.4 0.4	V
Input high voltage PA0–PA7, PB0, PB1/OSC3, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IH}	0.7 x V_{DD}	—	V_{DD}	V
Input low voltage PA0–PA7, PB0, PB1/OSC3, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IL}	V_{SS}	—	0.2 x V_{DD}	V
Supply current (f_{OP} = 2 MHz)[4], [5], [6], [7], [8] Run Wait Stop 25 °C 0 °C to +70 °C (standard) –40 °C to +85 °C (extended)	I_{DD}	 — — — — —	 — — 100 — —	 5.0 3.0 200 400 500	 mA mA nA nA nA
I/O ports hi-Z leakage current PA0–PA7, PB0–PB1 (without pulldowns activated)	I_{IL}	—	0.2	1	µA
Input pulldown current PA0–PA7, PB0–PB1	I_{IL}	50	100	200	µA
Input current $\overline{IRQ}$ and OSC1 $\overline{RESET}$ (V_{In} = V_{IH}) $\overline{RESET}$ (V_{In} = V_{IL})	I_{In}	— — —	— 15 50	1 — —	µA
$\overline{RESET}$, internal pulldown device	I_{In}	1.0	4.0	8.0	mA
Capacitance Ports (as input or output) $\overline{RESET}$, $\overline{IRQ}$, OSC1, OSC2	C_{Out} C_{In}	— —	— —	12 8	pF
Crystal/ceramic resonator Oscillator mode internal resistor OSC1 to OSC2	R_{OSC}	1.0	2.0	3.0	MΩ

Notes:

1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = –40 °C to +85 °C, unless otherwise noted.
2. All values shown reflect average measurements.
3. Typical values at midpoint of voltage range, 25 °C only
4. Wait I_{DD}: Only timer system active
5. Run (operating) I_{DD}, wait I_{DD}: Measured using external square wave clock source to OSC1, all inputs 0.2 Vdc from rail; no DC loads, less than 50 pF on all outputs, C_L = 20 pF on OSC2.
6. Wait, stop I_{DD}: All ports configured as inputs, V_{IL} = 0.2 Vdc, V_{IH} = V_{DD} –0.2 Vdc
7. Stop I_{DD} measured with OSC1 = V_{DD}, $\overline{RESET}$ open
8. Wait I_{DD} is affected linearly by the OSC2 capacitance.

11.7 3.0-Volt DC Electrical Characteristics

Characteristic[1]	Symbol	Min[2]	Typ[3]	Max	Unit
Output high voltage (I_{Load} = –0.4 mA) PA7–PA0, PB1/OSC3, PB0	V_{OH}	V_{DD} –0.3	—	—	V
Output low voltage PA3–PA0, PB1/OSC3, PB0 (I_{Load} = 0.4 mA) PA7–PA4 (I_{Load} = 4.0 mA)	V_{OL}	— —	— —	0.3 0.3	V
Input high voltage PA0–PA7, PB0, PB1/OSC3, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IH}	0.7 x V_{DD}	—	V_{DD}	V
Input low voltage PA0–PA7, PB0, PB1/OSC3, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IL}	V_{SS}	—	0.2 x V_{DD}	V
Supply current (f_{OP} = 1 MHz)[4], [5], [6], [7], [8] Run Wait Stop 25 °C 0 °C to +70 °C (standard) –40 °C to +85 °C (extended)	I_{DD}	 — — — — —	 — — 50 — —	 2.0 0.75 100 175 200	 mA mA nA nA nA
I/O ports hi-Z leakage current PA0–PA7, PB0–PB1 (without individual pulldown activated)	I_{IL}	—	0.1	1	µA
Input pulldown current PA0–PA7, PB0–PB1	I_{IL}	25	50	100	µA
Input current $\overline{IRQ}$, OSC1 $\overline{RESET}$ (V_{In} = V_{IH}) $\overline{RESET}$ (V_{In} = V_{IL})	I_{In}	— — —	— 10 30	1 — —	µA
$\overline{RESET}$, internal pulldown device	I_{In}	0.2	2.0	4.0	mA
Capacitance Ports (as input or output) $\overline{RESET}$, $\overline{IRQ}$, OSC1, OSC2	C_{Out} C_{In}	— —	— —	12 8	pF
Crystal/ceramic resonator Oscillator mode internal resistor OSC1 to OSC2	R_{OSC}	1.0	2.0	3.0	MΩ

Notes:

1. V_{DD} = 3.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = –40 °C to +85 °C, unless otherwise noted
2. All values shown reflect average measurements
3. Typical values at midpoint of voltage range, 25 °C only
4. Wait I_{DD}: Only timer system active
5. Run (operating) I_{DD}, wait I_{DD}: Measured using external square wave clock source to OSC1, all inputs 0.2 Vdc from rail; no DC loads, less than 50 pF on all outputs, C_L = 20 pF on OSC2
6. Wait, stop I_{DD}: All ports configured as inputs, V_{IL} = 0.2 Vdc, V_{IH} = V_{DD} –0.2 Vdc
7. Stop I_{DD} measured with OSC1 = V_{DD}, $\overline{RESET}$ open
8. Wait I_{DD} is affected linearly by the OSC2 capacitance

11.8 5.0-Volt Control Timing

Characteristic[1]	Symbol	Min	Max	Unit
Frequency of operation 3-pin RC oscillator option 2-pin RC oscillator option Crystal oscillator option External clock source	f_{OSC}	 0.1 0.1 0.5 DC	 1.25 2.7 4.0 4.0	MHz
Internal operating frequency RC oscillator ($f_{OSC} \div 2$) Crystal oscillator ($f_{OSC} \div 2$) External clock ($f_{OSC} \div 2$)	f_{OP}	 0.5 1.0 DC	 1.0 2.0 2.0	MHz
Cycle time ($1 \div f_{OP}$)	t_{cyc}	500	—	νs
RC oscillator stabilization time	t_{RCON}	—	1	ms
Crystal oscillator startup time (crystal oscillator option)	t_{OXON}	—	100	ms
Stop recovery startup time (crystal oscillator option)	t_{ILCH}	—	100	ms
$\overline{\text{RESET}}$ pulse width low	t_{RL}	1.5	—	t_{cyc}
Timer resolution[2]	t_{RESL}	4.0	—	t_{cyc}
$\overline{\text{IRQ}}$ interrupt pulse width low (edge-triggered)	t_{ILIH}	125	—	ns
$\overline{\text{IRQ}}$ interrupt pulse period	t_{ILIL}	[3]	—	t_{cyc}
PA0–PA3 interrupt pulse width high (edge-triggered)	t_{IHIL}	125	—	ns
PA0–PA3 interrupt pulse period	t_{IHIH}	[3]	—	t_{cyc}
OSC1 pulse width	t	90	—	ns
2-pin RC oscillator frequency combined stability[4] f_{OSC} = 500 kHz	Δf_{OSC}	—	±35	%
3-pin RC oscillator frequency combined stability[4] f_{OSC} = 500 kHz	Δf_{OSC}	—	±25	%
PEEPROM bit programming time	t_{EPGM}	—	10	ms
PEEPROM byte erase time	t_{ERBT}	—	10	ms
PEEPROM bulk erase time	t_{ERBK}	—	30	ms
PEEPROM charge pump startup time	t_{CP}	—	1	ms

Notes:
1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = –40 °C to +85 °C, unless otherwise noted
2. The 2-bit timer prescaler is the limiting factor in determining timer resolution.
3. The minimum period, t_{ILIL} or t_{IHIH}, should not be less than the number of cycles it takes to execute the interrupt service routine plus 19 t_{cyc}.
4. Effects of processing, temperature, and supply voltage (including tolerances of external 1% R and 2% C)

11.9 3.0-Volt Control Timing

Characteristic[1]	Symbol	Min	Max	Unit
Frequency of operation 3-pin RC oscillator option 2-pin RC oscillator option Crystal oscillator option External clock source	f_{OSC}	 0.1 0.1 0.4 DC	 0.6 0.7 2.0 2.0	MHz
Internal operating frequency RC oscillator ($f_{OSC} \div 2$) Crystal oscillator ($f_{OSC} \div 2$) External clock ($f_{OSC} \div 2$)	f_{OP}	 — — DC	 0.35 1.0 1.0	MHz
Cycle time ($1 \div f_{OP}$)	t_{cyc}	1.0	—	µs
RC oscillator stabilization time	t_{RCON}	—	1	ms
Crystal oscillator startup time (crystal oscillator option)	t_{OXON}	—	100	ms
Stop recovery startup time (crystal oscillator option)	t_{ILCH}	—	100	ms
$\overline{\text{RESET}}$ pulse width low	t_{RL}	1.5	—	t_{cyc}
Timer resolution[2]	t_{RESL}	4.0	—	t_{cyc}
$\overline{\text{IRQ}}$ interrupt pulse width low (edge-triggered)	t_{ILIH}	125	—	ns
$\overline{\text{IRQ}}$ interrupt pulse period	t_{ILIL}	[3]	—	t_{cyc}
PA0–PA3 interrupt pulse width high (edge-triggered)	t_{IHIL}	125	—	ns
PA0–PA3 interrupt pulse period	t_{IHIH}	[3]	—	t_{cyc}
OSC1 pulse width	t	180	—	ns
2-pin RC oscillator frequency combined stability[4] f_{OSC} = 500 kHz	Δf_{OSC}	—	±35	%
3-pin RC oscillator frequency combined stability[4] f_{OSC} = 500 kHz	Δf_{OSC}	—	±15	%
PEEPROM bit programming time	t_{EPGM}	—	15	ms
PEEPROM byte erase time	t_{ERBT}	—	15	ms
PEEPROM bulk erase time	t_{ERBK}	—	30	ms
PEEPROM charge pump startup time	t_{CP}	—	1	ms

Notes:
1. V_{DD} = 3.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = –40 °C to +85 °C, unless otherwise noted
2. The 2-bit timer prescaler is the limiting factor in determining timer resolution.
3. The minimum period, t_{ILIL} or t_{IHIH}, should not be less than the number of cycles it takes to execute the interrupt service routine plus 19 t_{cyc}.
4. Effects of processing, temperature, and supply voltage (including tolerances of external 1% R and 2% C)

11.10 1.8-Volt Control Timing (PEEPROM Read Only)

Characteristic[1]	Symbol	Min	Max	Unit
Frequency of operation 3-pin RC oscillator option 2-pin RC oscillator option Crystal oscillator option External clock source	f_{OSC}	 0.1 0.1 0.4 DC	 0.6 0.7 1.0 1.0	MHz
Internal operating frequency RC oscillator ($f_{OSC} \div 2$) Crystal oscillator ($f_{OSC} \div 2$) External clock ($f_{OSC} \div 2$)	f_{OP}	 — — DC	 350 500 500	kHz
Cycle time ($1 \div f_{OP}$)	t_{cyc}	2.0	—	µs
RC oscillator stabilization time	t_{RCON}	—	1	ms
Crystal oscillator startup time (crystal oscillator option)	t_{OXON}	—	100	ms
Stop recovery startup time (crystal oscillator option)	t_{ILCH}	—	100	ms
$\overline{\text{RESET}}$ pulse width low	t_{RL}	1.5	—	t_{cyc}
Timer resolution[2]	t_{RESL}	4.0	—	t_{cyc}
$\overline{\text{IRQ}}$ interrupt pulse width low (edge-triggered)	t_{ILIH}	125	—	ns
$\overline{\text{IRQ}}$ interrupt pulse period	t_{ILIL}	[3]	—	t_{cyc}
PA0–PA3 interrupt pulse width high (edge-triggered)	t_{IHIL}	125	—	ns
PA0–PA3 interrupt pulse period	t_{IHIH}	[3]	—	t_{cyc}
OSC1 pulse width	t	360	—	ns
2-pin RC oscillator frequency combined stability[4] f_{OSC} = 500 kHz	Δf_{OSC}	—	±35	%
3-pin RC oscillator frequency combined stability[4] f_{OSC} = 500 kHz	Δf_{OSC}	—	±15	%

Notes:
1. V_{DD} = 1.8 Vdc minimum, V_{SS} = 0 Vdc, T_A = –40 °C to +85 °C, unless otherwise noted
2. The 2-bit timer prescaler is the limiting factor in determining timer resolution.
3. The minimum period, t_{ILIL} or t_{IHIH}, should not be less than the number of cycles it takes to execute the interrupt service routine plus 19 t_{cyc}.
4. Effects of processing, temperature, and supply voltage (including tolerances of external 1% R and 2% C)

Section 12. Mechanical Specifications

12.1 Contents

12.2 Introduction

The MC68HC05K3 is available in these packages:

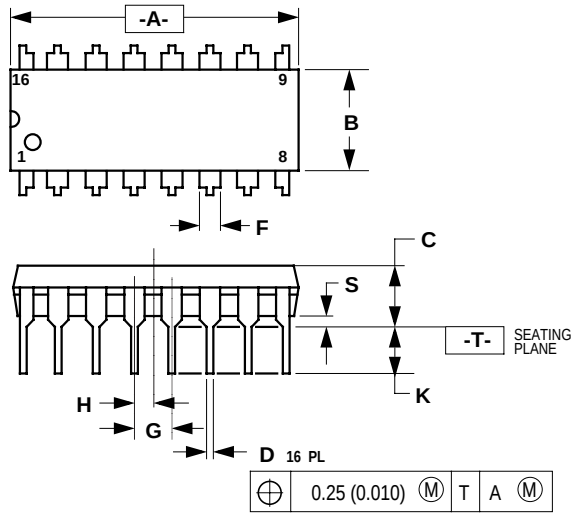
- Plastic dual in-line package (PDIP)
- Small outline integrated circuit (SOIC)
- Super small outline package (SSOP)

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

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

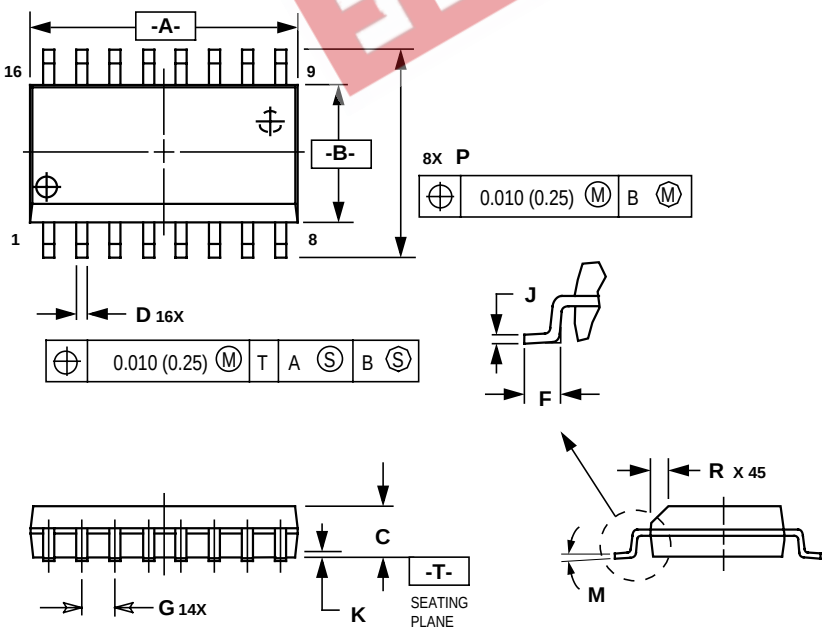
Follow Mfax or wwweb on-line instructions to retrieve the current mechanical specifications.

12.3 Dual In-Line Package (Case 648)



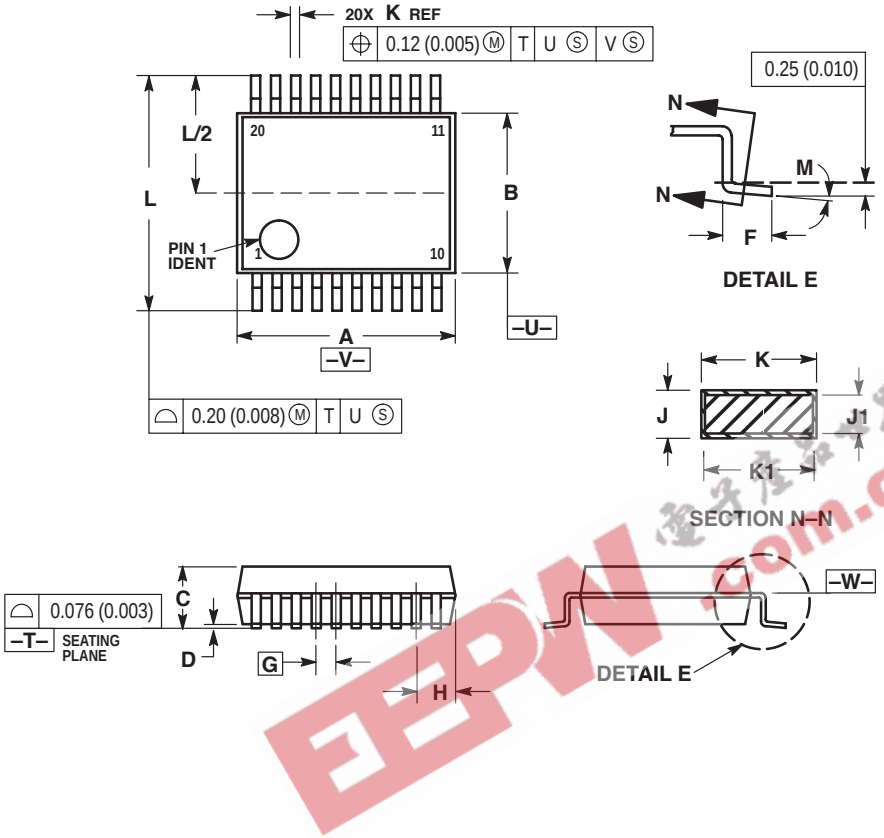
DIM	INCHES		MILLIMETERS	
	MIN	MAX	MIN	MAX
A	0.740	0.770	18.80	19.55
B	0.250	0.270	6.35	6.85
C	0.145	0.175	3.69	4.44
D	0.015	0.021	0.39	0.53
F	0.040	0.70	1.02	1.77
G	0.100 BSC		2.54 BSC	
H	0.050 BSC		1.27 BSC	
J	0.008	0.015	0.21	0.38
K	0.110	0.130	2.80	3.30
L	0.295	0.305	7.50	7.74
M	0°	10°	0°	10°
S	0.020	0.040	0.51	1.01

12.4 Small Outline Integrated Circuit (Case 751)



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

12.5 Super Small Outline Package (Case 940C)



NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: MILLIMETER.
3. DIMENSION A DOES NOT INCLUDE MOLD FLASH, PROTRUSIONS OR GATE BURRS. MOLD FLASH OR GATE BURRS SHALL NOT EXCEED 0.15 (0.006) PER SIDE.
4. DIMENSION B DOES NOT INCLUDE INTERLEAD FLASH OR PROTRUSION. INTERLEAD FLASH OR PROTRUSION SHALL NOT EXCEED 0.15 (0.006) PER SIDE.
5. DIMENSION K DOES NOT INCLUDE DAMBAR PROTRUSION/INTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL BE 0.13 (0.005) TOTAL IN EXCESS OF K DIMENSION AT MAXIMUM MATERIAL CONDITION. DAMBAR INTRUSION SHALL NOT REDUCE DIMENSION K BY MORE THAN 0.07 (0.002) AT LEAST MATERIAL CONDITION.
6. TERMINAL NUMBERS ARE SHOWN FOR REFERENCE ONLY.
7. DIMENSION A AND B ARE TO BE DETERMINED AT DATUM PLANE –W–.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	7.07	7.33	0.278	0.288
B	5.20	5.38	0.205	0.212
C	1.73	1.99	0.068	0.078
D	0.05	0.21	0.002	0.008
F	0.63	0.95	0.024	0.037
G	0.65 BSC		0.026 BSC	
H	0.59	0.75	0.023	0.030
J	0.09	0.20	0.003	0.008
J1	0.09	0.16	0.003	0.006
K	0.25	0.38	0.010	0.015
K1	0.25	0.33	0.010	0.013
L	7.65	7.90	0.301	0.311
M	0 °	8 °	0 °	8 °

Mechanical Specifications

Section 13. Ordering Information

13.1 Contents

13.2 Introduction

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

13.3 MCU Ordering Forms

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

- A current MCU ordering form that is **completely filled out** (Contact your Motorola sales office for assistance.)
- A copy of the customer specification if the customer specification deviates from the Motorola specification for the MCU
- Customer's application program on one of the media listed in **13.4 Application Program Media**

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

13.4 Application Program Media

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 in the United States and/or other countries..
3. PC-DOS is a trademark of International Business Machines Corporation.

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

If the memory map has two user ROM areas with the same 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.

13.5 ROM Program Verification

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

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

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

Check the listing verify file thoroughly, then complete and sign the listing verify form and return the listing verify form to Motorola. The signed

listing verify form constitutes the contractual agreement for the creation of the custom mask.

13.6 MC Order Numbers

Table 13-1 shows the MC order numbers for the available package types.

Table 13-1. MC Order Numbers

MC Order Number	Operating Temperature Range
MC68HC05K3P (standard)	–0 °C to 70 °C
MC68HC05K3CP (extended)	–40 °C to 85 °C
MC68HC05K3DW (standard)	–0 °C to 70 °C
MC68HC05K3CDW (extended)	–40 °C to 85 °C
MC68HC05K3CSD	–40 °C to 85 °C

Notes:
P = Plastic dual in-line package
DW = Small outline integrated circuit (SOIC) package
SD = Super small outline package (SSOP)

How to reach us:

USA/EUROPE/Locations Not Listed: Motorola Literature Distribution, P.O. Box 5405, Denver, Colorado 80217, 1-800-441-2447 or 1-303-675-2140. Customer Focus Center, 1-800-521-6274

JAPAN: Motorola Japan Ltd.: SPD, Strategic Planning Office, 141, 4-32-1 Nishi-Gotanda, Shinagawa-ku, Tokyo, Japan. 03-5487-8488

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

Mfax™, Motorola Fax Back System: RMFAX0@email.sps.mot.com; http://sps.motorola.com/mfax/; TOUCHTONE, 1-602-244-6609; US and Canada ONLY, 1-800-774-1848

HOME PAGE: http://motorola.com/sps/

MOTOROLA

MC68HC05K3/D