

HCO5

Freescale Semiconductor, Inc.

MC68HC05P8

TECHNICAL
DATA





Freescale Semiconductor, Inc.


Freescale Semiconductor, Inc.

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



MC68HC05P8

HCMOS MICROCONTROLLER UNIT

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

©MOTOROLA INC., 1990

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



Freescale Semiconductor, Inc.

Freescale Semiconductor, Inc.

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

TABLE OF CONTENTS

Paragraph Number	Title	Page Number
---------------------	-------	----------------

**Section 1
Introduction**

**Section 2
Pin Descriptions**

2.1	V _{DD} and V _{SS}	2-1
2.2	OSC1 and OSC2 (Oscillator Inputs)	2-2
2.2.1	Crystal (XTAL)	2-2
2.2.2	Ceramic Resonator.....	2-2
2.2.3	External Clock	2-3
2.3	V _{RH} and V _{RL}	2-4
2.4	RESET	2-4
2.5	$\overline{\text{IRQ}}$ (Interrupt Request).....	2-4

**Section 3
Parallel I/O**

3.1	I/O Port Function.....	3-1
3.2	Port A.....	3-2
3.3	Port B.....	3-3
3.4	Port D	3-3

**Section 4
Central Processor Unit**

4.1	CPU Registers.....	4-1
4.1.1	Accumulator (A).....	4-2
4.1.2	Index Register (X)	4-2
4.1.3	Stack Pointer (SP).....	4-3
4.1.4	Program Counter (PC).....	4-3
4.1.5	Condition Code Register (CCR).....	4-4
4.1.5.1	Half-Carry Bit (H).....	4-4

**TABLE OF CONTENTS
(Continued)**

Paragraph Number	Title	Page Number
4.1.5.2	Interrupt Mask (I)	4-5
4.1.5.3	Negative Bit (N)	4-5
4.1.5.4	Zero Bit (Z).....	4-5
4.1.5.5	Carry/Borrow Bit (C)	4-5
4.2	Arithmetic/Logic Unit (ALU) and CPU Control.....	4-6
4.3	Addressing Modes.....	4-6
4.3.1	Inherent.....	4-7
4.3.2	Immediate.....	4-8
4.3.3	Direct.....	4-8
4.3.4	Extended	4-10
4.3.5	Indexed, No Offset	4-10
4.3.6	Indexed, 8-Bit Offset	4-11
4.3.7	Indexed, 16-Bit Offset.....	4-11
4.3.8	Relative	4-12
4.4	Instruction Set.....	4-13
4.4.1	Register/Memory Instructions	4-14
4.4.2	Read-Modify-Write Instructions	4-15
4.4.3	Jump/Branch Instructions.....	4-15
4.4.4	Bit Manipulation Instructions.....	4-17
4.4.5	Control Instructions.....	4-17
4.4.6	Instruction Set Summary	4-18
4.4.7	Opcode Map	4-23
4.5	Low-Power Modes.....	4-24
4.5.1	STOP Mode	4-24
4.5.2	WAIT Mode.....	4-25

Section 5

Resets and Interrupts

5.1	Resets	5-1
5.1.1	Power-On Reset (POR).....	5-2
5.1.2	External $\overline{\text{RESET}}$ Input	5-2
5.1.3	Computer Operating Properly (COP) Reset.....	5-2
5.1.4	Illegal Address Reset	5-3
5.2	Interrupts.....	5-3
5.2.1	External Interrupt.....	5-7
5.2.2	Software Interrupt (SWI).....	5-8
5.2.3	Timer Interrupt	5-8

**TABLE OF CONTENTS
(Continued)**

Paragraph Number	Title	Page Number
---------------------	-------	----------------

**Section 6
Memory**

6.1	Memory Map.....	6-1
6.1.1	ROM.....	6-1
6.1.2	RAM.....	6-1
6.1.3	EEPROM.....	6-4
6.1.3.1	Programming Register (PROG).....	6-4
6.1.3.2	EEPROM Erasure	6-6
6.1.3.3	EEPROM Programming.....	6-7
6.1.3.4	Minimizing EEPROM Erase/Program Cycles	6-7
6.1.3.5	Low-Voltage Programming Inhibit (LVPI) Circuit.....	6-8
6.2	Data-Retention Mode.....	6-8

**Section 7
Timer**

7.1	Timer Status and Control Register (TSCR).....	7-2
7.2	COP Timer.....	7-3
7.3	Timer Count Register (TCR).....	7-4
7.4	Timer During WAIT Mode.....	7-4
7.5	Timer During STOP Mode.....	7-4

**Section 8
Analog-to-Digital Converter**

8.1	Conversion Process.....	8-1
8.2	A/D Status and Control Register (ADSCR).....	8-2
8.3	A/D Data Register (ADDR)	8-3
8.4	A/D Converter During WAIT Mode.....	8-4
8.5	A/D Converter During STOP Mode	8-4

**Section 9
Self-Check Mode**

9.1	Self-Check Circuit.....	9-1
9.2	Self-Check Results.....	9-1

**TABLE OF CONTENTS
(Concluded)**

Paragraph Number	Title	Page Number
---------------------	-------	----------------

Section 10

Electrical Specifications

10.1	Maximum Ratings	10-1
10.2	Thermal Characteristics.....	10-1
10.3	Power Considerations	10-2
10.4	DC Electrical Characteristics ($V_{DD} = 5.0$ Vdc).....	10-3
10.5	DC Electrical Characteristics ($V_{DD} = 3.3$ Vdc).....	10-4
10.6	A/D Converter Characteristics.....	10-7
10.7	Control Timing ($V_{DD} = 5.0$ Vdc).....	10-8
10.8	Control Timing ($V_{DD} = 3.3$ Vdc).....	10-10

Section 11

Mechanical Specifications

11.1	DIP (P Suffix)	11-1
11.2	SOIC (DW Suffix).....	11-2

Section 12

Ordering Information

12.1	ROM Pattern Media	12-1
12.1.1	Flexible Disks	12-1
12.1.2	EPROMs	12-2
12.2	ROM Pattern Verification	12-2
12.2.1	Verification Media.....	12-2
12.2.2	ROM Verification Units (RVUs).....	12-2
12.3	MC Order Numbers	12-2

LIST OF FIGURES

Figure Number	Title	Page Number
1-1	MC68HC05P8 Block Diagram	1-2
2-1	Pin Assignments	2-1
2-2	Crystal/Ceramic Resonator Connections.....	2-3
2-3	External Clock Source Connections.....	2-3
3-1	Parallel Port I/O Circuit.....	3-2
3-2	Port A Data Register and DDRA.....	3-2
3-3	Port B Data Register and DDRB.....	3-3
3-4	Port D Data Register.....	3-3
4-1	CPU Block Diagram	4-1
4-2	Programming Model.....	4-2
4-3	Accumulator (A).....	4-2
4-4	Index Register (X)	4-3
4-5	Stack Pointer (SP).....	4-3
4-6	Program Counter (PC).....	4-4
4-7	Condition Code Register (CCR).....	4-4
4-8	STOP Function Flowchart	4-24
4-9	WAIT Function Flowchart.....	4-26
5-1	COP Control Register.....	5-3
5-2	Interrupt Stacking Order	5-4
5-3	Reset and Interrupt Flowchart.....	5-6
5-4	$\overline{\text{IRQ}}$ Mask Option Logic	5-7
6-1	MC68HC05P8 Memory Map	6-2
6-2	I/O and Control Register Summary.....	6-3
6-3	Programming Register (PROG).....	6-4
7-1	Timer Block Diagram.....	7-1
7-2	Timer Status and Control Register (TSCR).....	7-2
7-3	Timer Counter Register (TCR).....	7-4

**LIST OF FIGURES
(Concluded)**

Figure Number	Title	Page Number
8-1	A/D Status and Control Register (ADSCR)	8-2
8-2	A/D Data Register (ADDR)	8-3
9-1	Self-Check Circuit.....	9-2
10-1	Test Load.....	10-2
10-2	Typical High-Side Driver Characteristics.....	10-5
10-3	Typical Low-Side Driver Characteristics.....	10-5
10-4	Typical Supply Current vs Clock Frequency	10-6
10-5	Maximum Supply Current vs Clock Frequency.....	10-6
10-6	STOP Recovery Timing	10-9
10-7	External Interrupt Timing	10-9
10-8	Power-On Reset Timing.....	10-11
10-9	External Reset Timing	10-11
10-10	LVPI Timing.....	10-12
11-1	Case 710-02 Dimensions	11-1
11-2	Case 751F-02 Dimensions.....	11-2

LIST OF TABLES

Table Number	Title	Page Number
3-1	I/O Pin Functions.....	3-2
4-1	Inherent Addressing Instructions.....	4-7
4-2	Immediate Addressing Instructions.....	4-8
4-3	Direct Addressing Instructions.....	4-9
4-4	Extended Addressing Instructions.....	4-10
4-5	Indexed Addressing Instructions.....	4-12
4-6	Relative Addressing Instructions.....	4-13
4-7	Register/Memory Instructions.....	4-14
4-8	Read-Modify-Write Instructions.....	4-15
4-9	Jump and Branch Instructions.....	4-16
4-10	Bit Manipulation Instructions.....	4-17
4-11	Control Instructions.....	4-17
4-12	Instruction Set.....	4-19
4-13	Opcode Map.....	4-23
6-1	Erasing Modes.....	6-5
7-1	RTI and COP Reset Rates ($f_{op} = 2$ MHz).....	7-3
8-1	A/D Input Selection.....	8-3
9-1	Self-Check Results.....	9-1
10-1	Maximum Ratings.....	10-1
10-2	Thermal Characteristics.....	10-1
10-3	DC Electrical Characteristics ($V_{DD} = 5.0$ Vdc).....	10-3
10-4	DC Electrical Characteristics ($V_{DD} = 3.3$ Vdc).....	10-4
10-5	A/D Converter Characteristics.....	10-7
10-6	Control Timing ($V_{DD} = 5.0$ Vdc).....	10-8
10-7	Control Timing ($V_{DD} = 3.3$ Vdc).....	10-10
12-1	MC Order Numbers.....	12-2



SECTION 1 INTRODUCTION

The MC68HC05P8 high-density complementary metal-oxide semiconductor (HCMOS) microcontroller unit (MCU) is a member of the popular M68HC05 Family of microcontrollers. This high-performance, low-cost MCU is a complete system on a single chip. The MCU features include the following:

- Memory-Mapped Input/Output (I/O)
- 2064 Bytes of On-Chip ROM
- 32 Bytes of EEPROM
- 112 Bytes of On-Chip RAM
- 16 Bidirectional I/O Lines and Four Input-Only Lines
- Fully Static Operation (No Minimum Clock Speed)
- On-Chip Oscillator
- 15-Stage Multifunctional Timer
- Real-Time Interrupt Circuit
- Four-Channel 8-bit Analog-to-Digital (A/D) Converter
- Self-Check Mode
- Power-Saving STOP, WAIT, and Data-Retention Modes
- Single 3.3-Volt to 5.0-Volt Supply (2-Volt Data-Retention Mode)
- 8 × 8 Unsigned Multiply Instruction
- Illegal Address Reset
- 28-Pin Dual-in-Line Package (DIP) or Small Outline Integrated Circuit (SOIC)

The following four mask options are available:

- Low-Voltage Programming Inhibit Circuit (Enable or Disable)
- Edge-Sensitive or Edge- and Level-Sensitive External Interrupt Trigger
- STOP Instruction (Enable or Disable)
- Computer Operating Properly (COP) Watchdog Timer (Enable or Disable)

Figure 1-1 shows the structure of the MC68HC05P8 MCU.

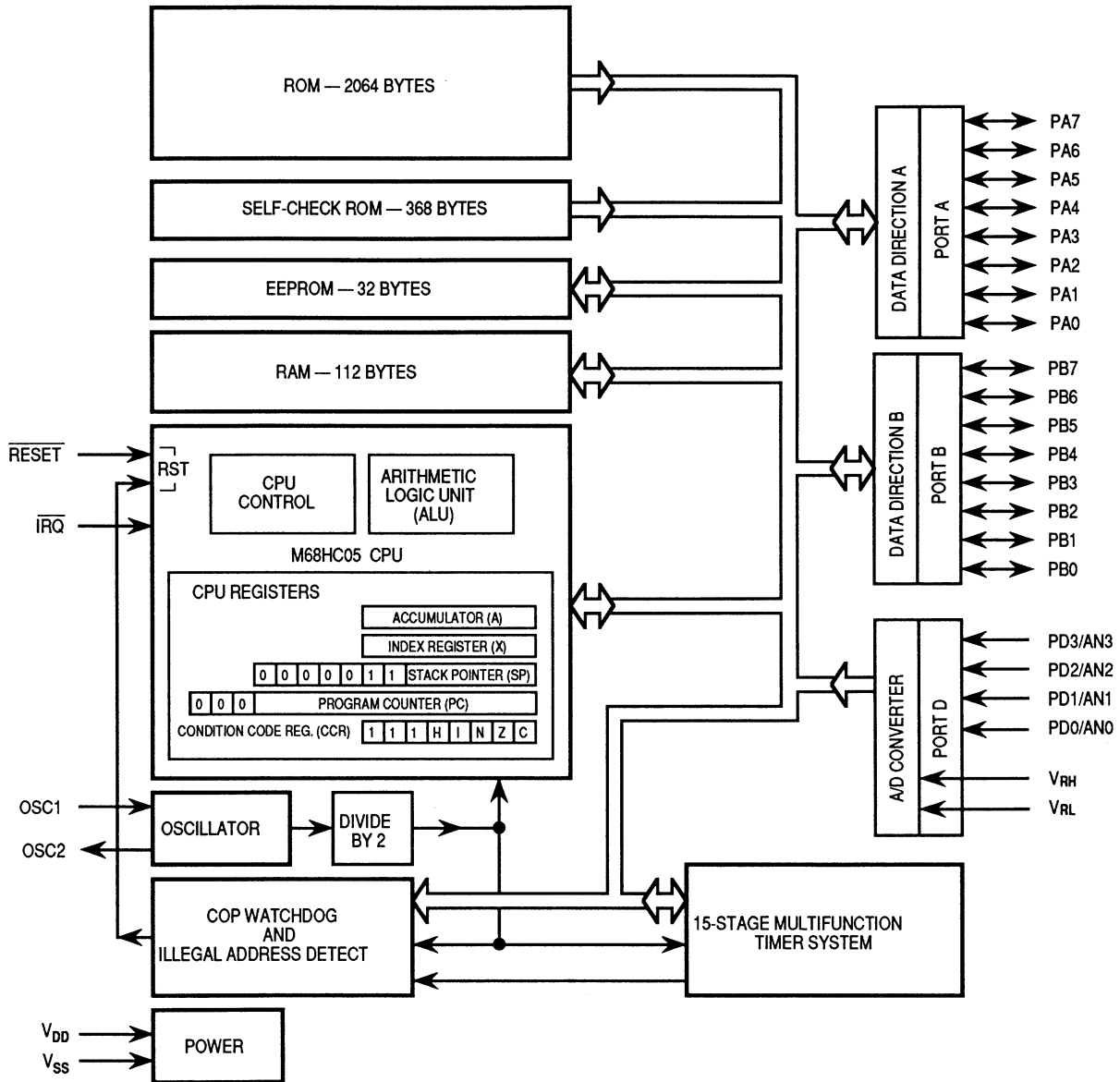


Figure 1-1. MC68HC05P8 Block Diagram

SECTION 2 PIN DESCRIPTIONS

This section describes the functions of the MC68HC05P8 pins. Figure 2-1 shows the pin assignments.

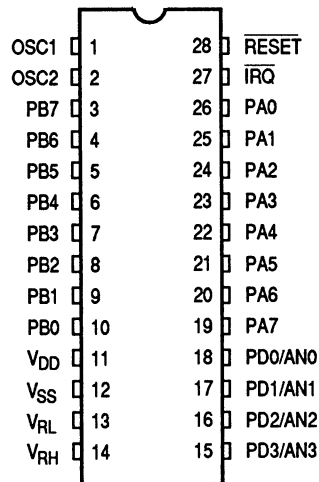


Figure 2-1. Pin Assignments

2.1 V_{DD} and V_{SS}

Power is supplied to the MCU through V_{DD} and V_{SS}. V_{DD} is the power supply, and V_{SS} is ground. The MCU operates from a single 5-volt (nominal) power supply.

Very fast 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 must be taken to provide good power supply bypassing at the MCU. Bypass capacitors should have good high-frequency characteristics and be as close to the MCU as possible. Bypassing requirements vary, depending on how heavily the MCU pins are loaded.

2.2 OSC1 and OSC2 (Oscillator Inputs)

OSC1 and OSC2 are the control connections for the on-chip oscillator. The OSC1 and OSC2 pins can accept the following:

- A crystal (see Figure 2-2)
- A ceramic resonator (see Figure 2-2)
- An external clock signal connected to OSC1 (see Figure 2-3)

2.2.1 Crystal (XTAL)

The circuit in Figure 2-2 shows a typical crystal oscillator circuit for an AT-cut, parallel resonant crystal. The crystal supplier's recommendations should be followed, since the crystal parameters determine the external component values required to provide maximum stability and reliable start-up. The load capacitance values used in the oscillator circuit design should include all stray layout capacitances. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and start-up stabilization time.

2.2.2 Ceramic Resonator

A ceramic resonator can be used in place of the crystal in cost-sensitive applications. The circuit in Figure 2-2 can be used for a ceramic resonator. The resonator manufacturer's recommendations should be followed, since the resonator parameters determine the external component values required to provide maximum stability and reliable starting. The load capacitance values used in the oscillator circuit design should include all stray layout capacitances.

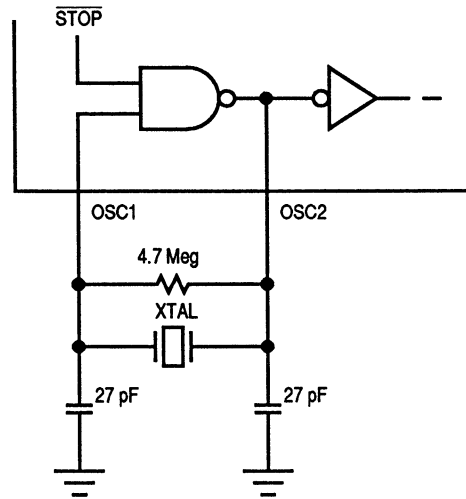


Figure 2-2. Crystal/Ceramic Resonator Connections

2.2.3 External Clock

An external clock from another CMOS-compatible device can be connected to the OSC1 input, with the OSC2 pin not connected, as shown in Figure 2-3.

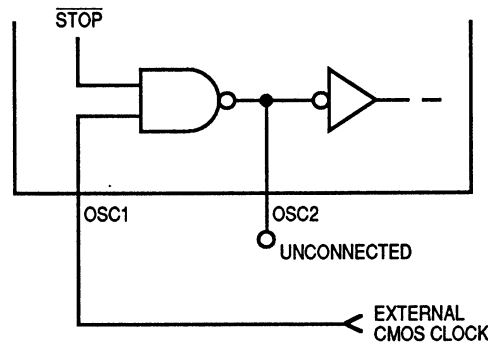


Figure 2-3. External Clock Source Connections

2.3. V_{RH} and V_{RL}

V_{RH} is the positive (high) reference voltage for the A/D converter. V_{RL} is the negative (low) reference voltage for the A/D converter. V_{RH} and V_{RL} should be isolated from V_{DD} and V_{SS} .

Very fast 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 must be taken to provide good power supply bypassing at the MCU. Bypass capacitors should have good high-frequency characteristics and be as close to the MCU as possible. Bypassing requirements vary, depending on how heavily the MCU pins are loaded.

2.4 \overline{RESET}

A logical zero on the \overline{RESET} pin forces the MCU to a known start-up state. (See **5.1 Resets** for more information.)

NOTE

The mask-optional low-voltage programming inhibit (LVPI) system protects against unintentional writes to the EEPROM during power-down. (See **6.1.3.5 Low-Voltage Programming Inhibit (LVPI) Circuit**.) If the LVPI mask option was not selected when ordering the MCU, the \overline{RESET} pin can be held low during power-down to prevent unpredictable writes to the EEPROM.

2.5 \overline{IRQ} (External Interrupt Request)

The \overline{IRQ} pin allows the application of asynchronous external interrupts to the MCU. Two different external interrupt triggering sensitivities are available. The factory-set mask options are the following:

- Negative edge-sensitive triggering only
- Both negative edge-sensitive triggering and low level-sensitive triggering

See **5.2 Interrupts** for more details concerning interrupts.

The \overline{IRQ} pin is also used in changing operating modes. (See **9.1 Self-Check Circuit**.)

SECTION 3 PARALLEL I/O

This section describes the three parallel I/O ports.

3.1 I/O Port Function

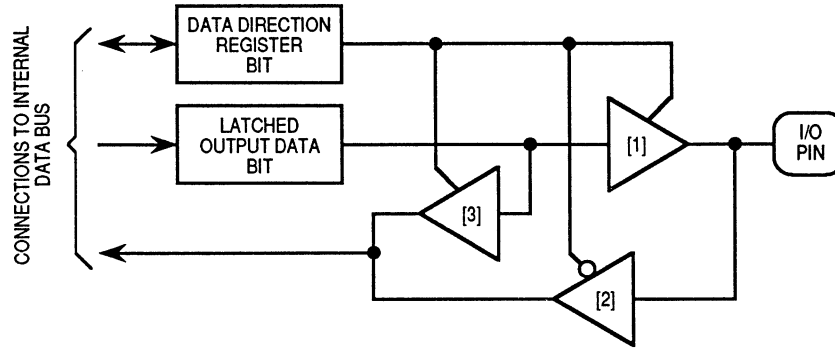
The MCU has 16 I/O pins that form two 8-bit I/O ports and four input-only pins that form a 4-bit input-only port. Each of the 16 I/O pins is programmable as an input or an output. Data direction is determined by the contents of the data direction register (DDR) for the port. Writing a logical one to a DDR bit enables the output buffer for that pin; a logical zero disables the output buffer. On reset, all DDR bits are initialized to logical zero to put the pins in the input mode.

NOTE

Any unused inputs and I/O pins should be connected to an appropriate logical level (e.g., either V_{DD} or V_{SS}). Although the I/O ports do not require termination for proper operation, termination is recommended to reduce the possibility of electrostatic damage.

A reset does not initialize the three port data registers. The port data registers for ports A, B, and D are at addresses \$00, \$01, and \$03, respectively. To avoid undefined levels, the data registers should be written before writing the DDR bits.

When a pin is programmed to be an output, reading the associated port bit actually reads the value of the output data latch and not the voltage on the pin itself. When a pin is programmed as an input, reading the port bit reads the voltage level on the I/O pin. The output data latch can always be written, regardless of the state of its DDR bit. (See Figure 3-1 for typical port circuitry, and Table 3-1 for a summary of I/O pin functions.)



- [1] Output buffer. Enables latched output to drive pin when DDR bit is 1 (output mode).
- [2] Input buffer. Enabled when DDR bit is 0 (input mode).
- [3] Input buffer. Enabled when DDR bit is 1 (output mode).

Figure 3-1. Parallel Port I/O Circuit

Table 3-1. I/O Pin Functions

R/W	DDR	I/O Pin Functions
0	0	The I/O pin is in input mode. Data is written into the output data latch.
0	1	Data is written into the output data latch, which drives the I/O pin.
1	0	The state of the I/O pin is read.
1	1	The I/O pin is in output mode. The output data latch is read.

NOTE: R/W is an internal signal.

3.2 Port A

PA7–PA0 form an 8-bit general-purpose bidirectional I/O port. The contents of data direction register A (DDRA) determine whether each pin is an input or an output. Figure 3-2 shows the port A data register and DDRA.

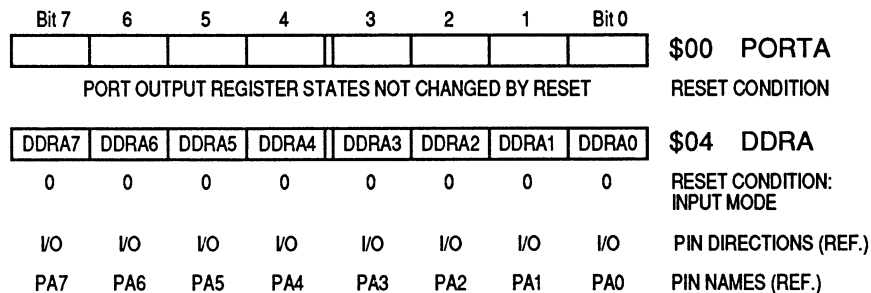


Figure 3-2. Port A Data Register and DDRA

3.3 Port B

PB7–PB0 form an 8-bit general-purpose bidirectional I/O port. The contents of data direction register B (DDRB) determine whether each pin is an input or an output. Figure 3-6 shows the port B data register and DDRB.

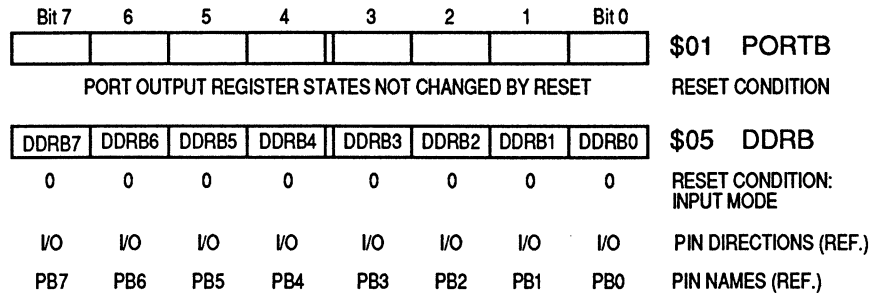


Figure 3-3. Port B Data Register and DDRB

3.4 Port D

PD3/AN3 (PORTD3/Analog Input 3)–PD0/AN0 form a 4-bit input-only port. When the A/D converter is enabled, one of these pins is the analog input to the A/D converter. The values of CH1 and CH0 in the A/D status and control register (ADSCR) select one of the four as the analog input pin. A digital read of this port while the A/D converter is enabled results in a read of logical zero from the selected analog input pin. A digital read of the remaining three pins gives their correct digital values. (See SECTION 8 ANALOG-TO-DIGITAL CONVERTER.)

When the A/D converter is disabled, PD3/AN3–PD0/AN0 are general-purpose digital inputs. A reset turns off the A/D converter and configures port D as a digital input.

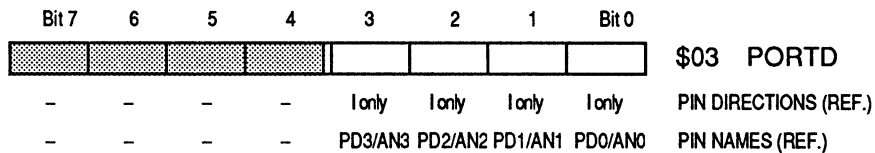


Figure 3-4. Port D Data Register



SECTION 4 CENTRAL PROCESSOR UNIT

This section describes the registers, instruction set, and addressing modes of the M68HC05 central processor unit (CPU). The STOP and WAIT modes are initiated by software instructions and are also described in this section.

The M68HC05 CPU executes all instructions of the earlier M6805 and M146805 instruction sets and is upgraded to include an 8 × 8 bit unsigned multiply instruction.

4.1 CPU Registers

The CPU contains the following five registers:

- Accumulator (A)
- Index register (X)
- Stack pointer (SP)
- Program counter (PC)
- Condition code register (CCR)

The CPU registers are hard-wired within the CPU and are not part of the memory map. Figure 4-1 is a block diagram of the MC68HC05 CPU.

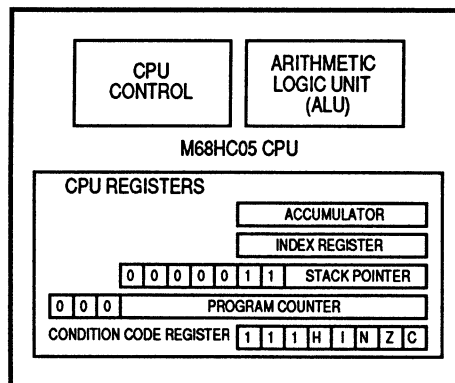


Figure 4-1. CPU Block Diagram

Figure 4-2 shows the five CPU registers.

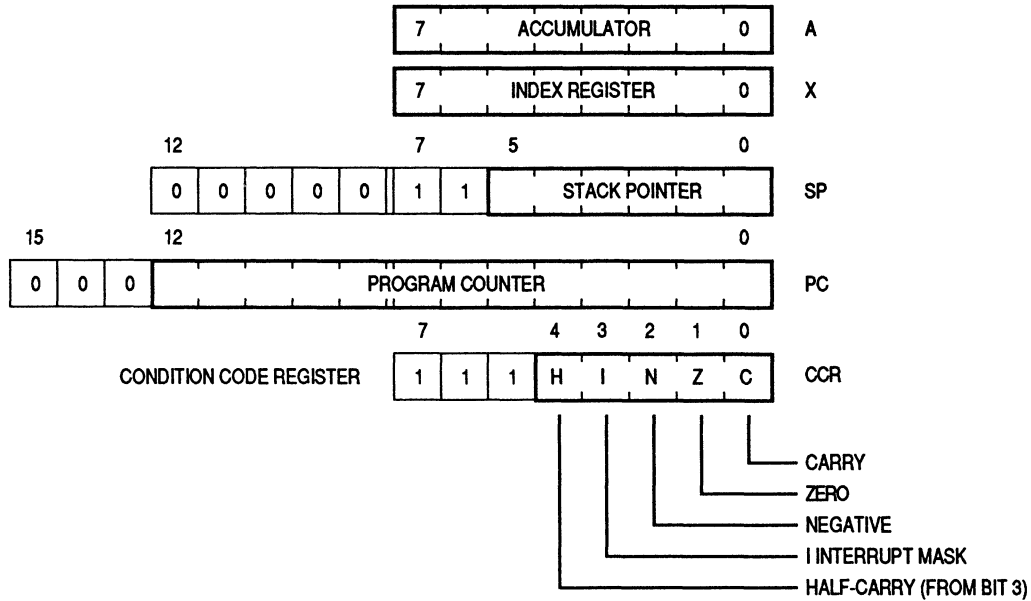


Figure 4-2. Programming Model

4.1.1 Accumulator (A)

The accumulator is a general-purpose 8-bit register. The CPU uses the accumulator to hold operands and results of arithmetic and nonarithmetic operations. (See Figure 4-3.)

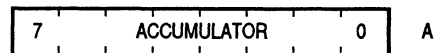


Figure 4-3. Accumulator (A)

4.1.2 Index Register (X)

The index register is an 8-bit register that can perform two functions:

- Indexed addressing
- Temporary storage

In indexed addressing with no offset, the index register contains the low byte of the operand address, and the high byte is assumed to be \$00. In indexed

addressing with an 8-bit offset, the CPU finds the operand address by adding the index register contents to an 8-bit immediate value. In indexed addressing with a 16-bit offset, the CPU finds the operand address by adding the index register contents to a 16-bit immediate value. (See **4.3 Addressing Modes.**)

The index register can also serve as an auxiliary accumulator for temporary storage. (See Figure 4-4.)



Figure 4-4. Index Register (X)

4.1.3 Stack Pointer (SP)

The stack pointer is a 13-bit register that contains the address of the next free location on the stack. During a reset or after the reset stack pointer (RSP) instruction, the stack pointer contents are set to \$FF. The address in the stack pointer is decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

When accessing memory, the seven most significant bits of the stack pointer are permanently set to 000011. (See Figure 4-5.) These seven bits are appended to the six least significant register bits to produce an address within the range of \$FF-\$C0. Subroutines and interrupts may use up to 64 locations. If 64 locations are exceeded, the stack pointer wraps around and writes over the previously stored information. A subroutine call occupies two locations on the stack; an interrupt uses five locations.

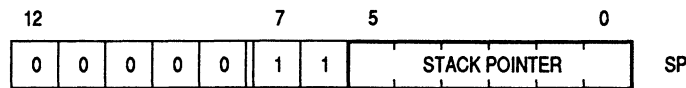


Figure 4-5. Stack Pointer (SP)

4.1.4 Program Counter (PC)

The program counter is a 13-bit register that contains the address of the next instruction or operand to be fetched. Since addresses are often 16-bit values, the program counter may be thought of as having three additional upper bits that are always zeros. (See Figure 4-6.)



Figure 4-6. Program Counter (PC)

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

4.1.5 Condition Code Register (CCR)

The condition code register is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. A fifth bit is the interrupt mask. (See Figure 4-7.) These bits can be individually tested by a program, and specific actions can be taken as a result of their states. The condition code register should be thought of as having three additional upper bits that are always ones.

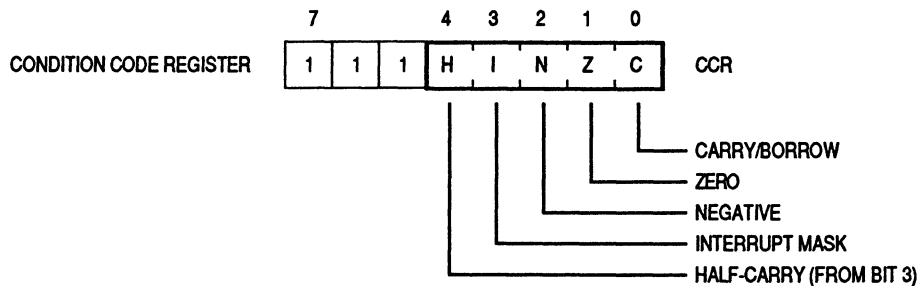


Figure 4-7. Condition Code Register (CCR)

The following paragraphs explain the functions of the lower five bits of the condition code register.

4.1.5.1 Half-Carry Bit (H)

When the half-carry bit is set, it means that a carry occurred between bits 3 and 4 of the accumulator during the last ADD or ADC operation. The half-carry bit is required for binary-coded decimal (BCD) arithmetic operations.

4.1.5.2 Interrupt Mask (I)

When the interrupt mask is set, timer interrupts and external interrupts are disabled. Interrupts are enabled when the interrupt mask is cleared. When an interrupt occurs, the interrupt mask is automatically set after the CPU registers are saved on the stack, but before the interrupt vector is fetched. If an interrupt occurs while the interrupt mask is set, the interrupt is latched. Normally, the interrupt is processed as soon as the interrupt mask is cleared.

A return from interrupt (RTI) instruction pulls the CPU registers from the stack, restoring the interrupt mask to its cleared state. After any reset, the interrupt mask is set and can only be cleared by a software instruction.

4.1.5.3 Negative Bit (N)

The negative bit is set when the result of the last arithmetic operation, logical operation, or data manipulation was negative. (Bit 7 of the result was a logical one.)

The negative bit can also be used to check an often-tested flag by assigning the flag to bit 7 of a register or memory location. Loading the accumulator with the contents of that register or location then sets or clears the negative bit according to the state of the flag.

4.1.5.4 Zero Bit (Z)

The zero bit is set when the result of the last arithmetic operation, logical operation, or data manipulation was zero.

4.1.5.5 Carry/Borrow Bit (C)

The carry/borrow bit is set when a carry out of bit 7 of the accumulator occurred during the last arithmetic operation, logical operation, or data manipulation. The carry/borrow bit is also set or cleared during bit test and branch instructions and during shifts and rotates.

4.2 Arithmetic/Logic Unit (ALU) and CPU Control

The ALU performs the arithmetic and logical operations defined by the instruction set.

The binary arithmetic circuits decode the instruction and set up the ALU for the desired function. Most binary arithmetic is based on the addition algorithm, and subtraction is carried out as negative addition. Multiplication is not performed as a discrete instruction but as a chain of addition and shift operations within the ALU. The multiply instruction (MUL) requires 11 internal processor cycles to complete this chain of operations.

The CPU control circuitry sequences the logic elements of the ALU to carry out the required operations.

4.3 Addressing Modes

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

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

4.3.1 Inherent

The inherent addressing mode is used for instructions with no operand (e.g., STOP) and for some of the instructions that act on data in the CPU registers (e.g., CLRA). No memory address is required for inherent instructions. Inherent instructions are one byte long. Table 4-1 lists the instructions that can be used in the inherent addressing mode.

Table 4-1. Inherent Addressing Instructions

Instruction	Mnemonic
Arithmetic Shift Left	ASLA, ASLX
Arithmetic Shift Right	ASRA, ASRX
Clear Carry Bit	CLC
Clear Interrupt Mask	CLI
Clear	CLRA, CLRX
Complement	COMA, COMX
Decrement	DECA, DECX
Increment	INCA, INCX
Logical Shift Left	LSLA, LSLX
Logical Shift Right	LSRA, LSRX
Multiply	MUL
Negate	NEGA, NEGX
No Operation	NOP
Rotate Left through Carry	ROLA, ROLX
Rotate Right through Carry	RORA, RORX
Reset Stack Pointer	RSP
Return from Interrupt	RTI
Return from Subroutine	RTS
Set Carry Bit	SEC
Set Interrupt Mask	SEI
Enable \overline{IRQ} and Stop Oscillator	STOP
Software Interrupt	SWI
Transfer Accumulator to Index Register	TAX
Test for Negative or Zero	TSTA, TSTX
Transfer Index Register to Accumulator	TXA
Enable Interrupt and Halt Processor	WAIT

4.3.2 Immediate

The immediate addressing mode is used for instructions that contain a value to be used in an operation with the value in the accumulator or index register. No memory address is required for immediate instructions. The operand is contained in the byte immediately following the opcode. These are two-byte instructions, one for the opcode and one for the immediate data byte. Table 4-2 lists the instructions that can be used in the immediate addressing mode.

Table 4-2. Immediate Addressing Instructions

Instruction	Mnemonic
Add with Carry	ADC
Add	ADD
Logical AND	AND
Bit Test Memory with Accumulator	BIT
Compare Accumulator with Memory	CMP
Compare Index Register with Memory	CPX
Exclusive OR Memory with Accumulator	EOR
Load Accumulator from Memory	LDA
Load Index Register from Memory	LDX
Inclusive OR	ORA
Subtract with Carry	SBC
Subtract	SUB

4.3.3 Direct

The direct addressing mode is used to access data within the first 256 bytes of memory with a single two-byte instruction. In the direct addressing mode, the low byte of the operand's address is contained in the byte following the opcode. The high byte of the address is assumed to be \$00. Most direct instructions take two bytes, one for the opcode and one for the operand's address. BRSET and BRCLR are three-byte instructions that use direct addressing to access the operand and relative addressing to specify a branch destination. Table 4-3 lists the instructions that can be used in the direct addressing mode.

Table 4-3. Direct Addressing Instructions

Instruction	Mnemonic
Add with Carry	ADC
Add	ADD
Logical AND	AND
Arithmetic Shift Left	ASL
Arithmetic Shift Right	ASR
Clear Bit in Memory	BCLR
Bit Test Memory with Accumulator	BIT
Branch if Bit n Is Clear	BRCLR
Branch if Bit n Is Set	BRSET
Set Bit in Memory	BSET
Clear	CLR
Compare Accumulator with Memory	CMP
Complement	COM
Compare Index Register with Memory	CPX
Decrement	DEC
Exclusive OR Memory with Accumulator	EOR
Increment	INC
Jump	JMP
Jump to Subroutine	JSR
Load Accumulator from Memory	LDA
Load Index Register from Memory	LDX
Logical Shift Left	LSL
Logical Shift Right	LSR
Negate	NEG
Inclusive OR	ORA
Rotate Left through Carry	ROL
Rotate Right through Carry	ROR
Subtract with Carry	SBC
Store Accumulator in Memory	STA
Store Index Register in Memory	STX
Subtract	SUB
Test for Negative or Zero	TST

NOTE: ASL = LSL

4.3.4 Extended

The extended addressing mode is used to access data in any memory location with a single three-byte instruction. In the extended addressing mode, the high and low bytes of the operand's address are contained in the two bytes following the opcode. Extended instructions take three bytes, one for the opcode and two for the operand's address.

When using the Motorola assembler, the user need not specify whether an instruction uses direct or extended addressing. The assembler automatically selects the shortest form of the instruction. Table 4-4 lists the instructions that can be used in the extended addressing mode.

Table 4-4. Extended Addressing Instructions

Instruction	Mnemonic
Add with Carry	ADC
Add	ADD
Logical AND	AND
Bit Test Memory with Accumulator	BIT
Compare Accumulator with Memory	CMP
Compare Index Register with Memory	CPX
Exclusive OR Memory with Accumulator	EOR
Jump	JMP
Jump to Subroutine	JSR
Load Accumulator from Memory	LDA
Load Index Register from Memory	LDX
Inclusive OR	ORA
Subtract with Carry	SBC
Store Accumulator in Memory	STA
Store Index Register in Memory	STX
Subtract	SUB

4.3.5 Indexed, No Offset

The indexed, no offset addressing mode is used to access data with variable addresses within the first 256 memory locations. The CPU finds the low byte of the operand's conditional address by reading the contents of the index register. The high byte is assumed to be \$00. These instructions are only one byte long. The indexed, no offset mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location. Table 4-5 lists the instructions that can be used in the indexed, no offset addressing mode.

4.3.6 Indexed, 8-Bit Offset

The indexed, 8-bit offset addressing mode is used to access data with variable addresses within the first 511 memory locations. The CPU finds the operand's conditional address by adding the unsigned contents of the index register to the unsigned byte following the opcode. This addressing mode is useful for selecting the kth element in an n-element table. The table may begin anywhere within the first 256 memory locations and could extend as far as location 510 (\$1FE). With this two-byte instruction, k typically would be in the index register, and the address of the beginning of the table would be in the byte following the opcode. Table 4-5 lists the instructions that can be used in the indexed, 8-bit offset addressing mode.

4.3.7 Indexed, 16-Bit Offset

The indexed, 16-bit offset addressing mode is used to access data with variable addresses at any location in memory. The CPU finds the operand's conditional address by adding the unsigned contents of the 8-bit index register to the 16-bit unsigned word formed by the two bytes following the opcode. The first byte after the opcode is the high byte of the 16-bit offset; the second byte is the low byte. This addressing mode can be used in a manner similar to indexed, 8-bit offset, but this three-byte instruction allows tables to be anywhere in memory.

As with direct and extended addressing, the Motorola assembler determines the shortest form of indexed addressing. Table 4-5 lists the instructions that can be used in the indexed, 16-bit offset addressing mode.

Table 4-5. Indexed Addressing Instructions

Instruction	Mnemonic	No Offset	8-Bit Offset	16-Bit Offset
Add with Carry	ADC	√	√	√
Add	ADD	√	√	√
Logical AND	AND	√	√	√
Arithmetic Shift Left	ASL	√	√	
Arithmetic Shift Right	ASR	√	√	
Bit Test Memory with Accumulator	BIT	√	√	√
Clear	CLR	√	√	
Compare Accumulator with Memory	CMP	√	√	√
Complement	COM	√	√	
Compare Index Register with Memory	CPX	√	√	√
Decrement	DEC	√	√	
Exclusive OR Memory with Accumulator	EOR	√	√	√
Increment	INC	√	√	
Jump	JMP	√	√	√
Jump to Subroutine	JSR	√	√	√
Load Accumulator from Memory	LDA	√	√	√
Load Index Register from Memory	LDX	√	√	√
Logical Shift Left	LSL	√	√	
Logical Shift Right	LSR	√	√	
Negate	NEG	√	√	
Inclusive OR	ORA	√	√	√
Rotate Left through Carry	ROL	√	√	
Rotate Right through Carry	ROR	√	√	
Subtract with Carry	SBC	√	√	√
Store Accumulator in Memory	STA	√	√	√
Store Index Register in Memory	STX	√	√	√
Subtract	SUB	√	√	√
Test for Negative or Zero	TST	√	√	

4.3.8 Relative

The relative addressing mode is used only for branch instructions and bit test and branch instructions. The CPU finds the conditional branch destination by adding the signed byte following the opcode to the contents of the program counter if the branch condition is true. If the branch condition is not true, the program counter goes to the next instruction. In order to branch either forward or backward, the offset is a signed, twos complement byte that gives a branching range of -128 to +127 bytes from the address of the next location after the branch instruction.

The programmer need not calculate the offset when using the Motorola assembler since it calculates the proper offset and checks to see that it is within the span of the branch. Table 4-6 lists the instructions that can use the relative addressing mode.

Table 4-6. Relative Addressing Instructions

Instruction	Mnemonic
Branch if Carry Clear	BCC
Branch if Carry Set	BCS
Branch if Equal	BEQ
Branch if Half-Carry Clear	BHCC
Branch if Half-Carry Set	BHCS
Branch if Higher	BHI
Branch if Higher or Same	BHS
Branch if Interrupt Line is High	BIH
Branch if Interrupt Line Is Low	BIL
Branch if Lower	BLO
Branch if Lower or Same	BLS
Branch if Interrupt Mask is Clear	BMC
Branch if Minus	BMI
Branch if Interrupt Mask Is Set	BMS
Branch if Not Equal	BNE
Branch if Plus	BPL
Branch Always	BRA
Branch if Bit n Is Clear	BRCLR
Branch if Bit n Is Set	BRSET
Branch Never	BRN
Branch to Subroutine	BSR

4.4 Instruction Set

This MCU uses all the instructions available in the M146805 CMOS Family plus the unsigned multiply (MUL) instruction. The MUL instruction allows unsigned multiplication of the contents of the accumulator and the index register. The high-order product is then stored in the index register, and the low-order product is stored in the accumulator.

The MCU instructions can be divided into five basic types:

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

4.4.1 Register/Memory Instructions

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. Most register/memory instructions can be used in the following addressing modes:

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

Table 4-7 lists the register/memory instructions.

Table 4-7. Register/Memory Instructions

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

4.4.2 Read-Modify-Write Instructions

These instructions read a memory location or a register, modify its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not write a replacement value. Read-modify-write instructions can be used in the following addressing modes:

- Inherent
- Direct
- Indexed, no offset
- Indexed, 8-bit offset

Table 4-8 lists the read-modify-write instructions.

Table 4-8. Read-Modify-Write Instructions

Instruction	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (Twos Complement)	NEG
Rotate Left through Carry	ROL
Rotate Right through Carry	ROR
Logical Shift Left	LSL
Logical Shift Right	LSR
Arithmetic Shift Right	ASR
Test for Negative or Zero	TST

4.4.3 Jump/Branch Instructions

Jump instructions allow the CPU to interrupt the normal sequence of the program counter. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Jump instructions can be used in the following addressing modes:

- Direct
- Extended
- Indexed, no offset
- Indexed, 8-bit offset
- Indexed, 16-bit offset

Branch instructions allow the CPU to interrupt the normal sequence of the program counter when a test condition is met. If the test condition is not met, the

branch is not performed. All branch instructions are used in the relative addressing mode.

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

Table 4-9 lists the jump and branch instructions.

Table 4-9. Jump and Branch Instructions

Instruction	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Bit n of M = 0	BRCLR
Branch if Bit n of M = 1	BRSET
Branch if Higher	BHI
Branch if Lower or Same	BLS
Branch if Carry Clear	BCC
Branch if Higher or Same	BHS
Branch if Carry Set	BCS
Branch if Lower	BLO
Branch if Not Equal	BNE
Branch if Equal	BEQ
Branch if Half-Carry Clear	BHCC
Branch if Half-Carry Set	BHCS
Branch if Plus	BPL
Branch if Minus	BMI
Branch if Interrupt Mask Clear	BMC
Branch if Interrupt Mask Set	BMS
Branch if Interrupt Line Low	BIL
Branch if Interrupt Line High	BIH
Branch to Subroutine	BSR
Jump Unconditional	JMP
Jump to Subroutine	JSR

Table 4-12. Instruction Set (Sheet 1 of 4)

Source Form(s)	Operation	Description	Addressing Mode for Operand	Machine Coding (hexadecimal)		Cycles	Condition Code				
				Opcode	Operand		H	I	N	Z	C
ADC opr	Add with carry	$A \leftarrow (A) + (M) + C$	IMM	A9	ii	2	↑	-	↑	↑	↑
			DIR	B9	dd	3					
			EXT	C9	hh ll	4					
			IX2	D9	ee ff	5					
			IX1	E9	ff	4					
			IX	F9		3					
ADD opr	Add without carry	$A \leftarrow (A) + (M)$	IMM	AB	ii	2	↑	-	↑	↑	↑
			DIR	BB	dd	3					
			EXT	CB	hh ll	4					
			IX2	DB	ee ff	5					
			IX1	EB	ff	4					
			IX	FB		3					
AND opr	Logical AND	$A \leftarrow (A) \bullet (M)$	IMM	A4	ii	2	-	-	↑	↑	-
			DIR	B4	dd	3					
			EXT	C4	hh ll	4					
			IX2	D4	ee ff	5					
			IX1	E4	ff	4					
			IX	F4		3					
ASL opr ASLA ASLX ASL opr ASL opr	Arithmetic shift left		DIR	38	dd	5	-	-	↑	↑	↑
			INH	48		3					
			INH	58		3					
			IX1	68	ff	6					
			IX	78		5					
ASR opr ASRA ASRX ASR opr ASR opr	Arithmetic shift right		DIR	37	dd	5	-	-	↑	↑	↑
			INH	47		3					
			INH	57		3					
			IX1	67	ff	6					
			IX	77		5					
BCC rel	Branch if carry bit clear	? C = 0	REL	24	rr	3	-	-	-	-	
BCLR n opr	Clear bit n	$M_n \leftarrow 0$	DIR (b0)	11	dd	5	-	-	-	-	-
			DIR (b1)	13	dd	5					
			DIR (b2)	15	dd	5					
			DIR (b3)	17	dd	5					
			DIR (b4)	19	dd	5					
			DIR (b5)	1B	dd	5					
			DIR (b6)	1D	dd	5					
			DIR (b7)	1F	dd	5					
BCS rel	Branch if carry bit set	? C = 1	REL	25	rr	3	-	-	-	-	
BEQ rel	Branch if equal	? Z = 1	REL	27	rr	3	-	-	-	-	
BHCC rel	Branch if half carry bit clear	? H = 0	REL	28	rr	3	-	-	-	-	
BHCS rel	Branch if half carry bit set	? H = 1	REL	29	rr	3	-	-	-	-	
BHI rel	Branch if higher	? C + Z = 0	REL	22	rr	3	-	-	-	-	
BHS rel	Branch if higher or same	? C = 0	REL	24	rr	3	-	-	-	-	
BIH rel	Branch if \overline{IRQ} pin high	? $\overline{IRQ} = 1$	REL	2F	rr	3	-	-	-	-	
BIL rel	Branch if \overline{IRQ} pin low	? $\overline{IRQ} = 0$	REL	2E	rr	3	-	-	-	-	
BIT rel	Bit test accumulator contents with memory contents	$(A) \bullet (M)$	IMM	A5	ii	2	-	-	↑	↑	-
			DIR	B5	dd	3					
			EXT	C5	hh ll	4					
			IX2	D5	ee ff	5					
			IX1	E5	ff	4					
			IX	F5		3					
BLO rel	Branch if lower	? C = 1	REL	25	rr	3	-	-	-	-	
BLS rel	Branch if lower or same	? C + Z = 1	REL	23	rr	3	-	-	-	-	
BMC rel	Branch if interrupt mask clear	? I = 0	REL	2C	rr	3	-	-	-	-	
BMI rel	Branch if minus	? N = 1	REL	2B	rr	3	-	-	-	-	
BMS rel	Branch if interrupt mask set	? I = 0	REL	2D	rr	3	-	-	-	-	
BNE rel	Branch if not equal	? Z = 0	REL	26	rr	3	-	-	-	-	
BPL rel	Branch if plus	? N = 0	REL	2A	rr	3	-	-	-	-	

Freescale Semiconductor, Inc.

Table 4-12. Instruction Set (Sheet 2 of 4)

Source Form(s)	Operation	Description	Addressing Mode for Operand	Machine Coding (hexadecimal)		Cycles	Condition Code				
				Opcode	Operand		H	I	N	Z	C
BRA rel	Branch always	? 1 = 1	REL	20	rr	3	-	-	-	-	-
BRCLR n opr rel	Branch if bit n clear	? Mn = 0	DIR (b0)	01	dd rr	5	-	-	-	-	↓
			DIR (b1)	03	dd rr	5	-	-	-	-	-
			DIR (b2)	05	dd rr	5	-	-	-	-	-
			DIR (b3)	07	dd rr	5	-	-	-	-	-
			DIR (b4)	09	dd rr	5	-	-	-	-	-
			DIR (b5)	0B	dd rr	5	-	-	-	-	-
			DIR (b6)	0D	dd rr	5	-	-	-	-	-
DIR (b7)	0F	dd rr	5	-	-	-	-	-			
BRN rel	Branch never	? 1 = 0	REL	21	rr	3	-	-	-	-	
BRSET n opr rel	Branch if bit n set	? Mn = 1	DIR (b0)	00	dd rr	5	-	-	-	-	↓
			DIR (b1)	02	dd rr	5	-	-	-	-	-
			DIR (b2)	04	dd rr	5	-	-	-	-	-
			DIR (b3)	06	dd rr	5	-	-	-	-	-
			DIR (b4)	08	dd rr	5	-	-	-	-	-
			DIR (b5)	0A	dd rr	5	-	-	-	-	-
			DIR (b6)	0C	dd rr	5	-	-	-	-	-
DIR (b7)	0E	dd rr	5	-	-	-	-	-			
BSET n opr	Set bit n	Mn ← 1	DIR (b0)	10	dd	5	-	-	-	-	-
			DIR (b1)	12	dd	5	-	-	-	-	-
			DIR (b2)	14	dd	5	-	-	-	-	-
			DIR (b3)	16	dd	5	-	-	-	-	-
			DIR (b4)	18	dd	5	-	-	-	-	-
			DIR (b5)	1A	dd	5	-	-	-	-	-
			DIR (b6)	1C	dd	5	-	-	-	-	-
DIR (b7)	1E	dd	5	-	-	-	-	-			
BSR rel	Branch to subroutine	PC ← (PC) + 2; push (PCL) SP ← (SP) - 1; push (PCH) SP ← (SP) - 1 PC ← (PC) + rel	REL	AD	rr	6	-	-	-	-	
CLC	Clear carry bit	C ← 0	INH	98		2	-	-	-	-	
CLI	Clear interrupt mask	I ← 0	INH	9A		2	-	0	-	-	
CLR opr CLRA CLR opr CLR opr CLR opr	Clear register	M ← \$00 A ← \$00 X ← \$00 M ← \$00 M ← \$00	DIR	3F	dd	5	-	-	0	1	-
			INH	4F		3	-	-	-	-	-
			INH	5F		3	-	-	-	-	-
			IX1	6F	ff	6	-	-	-	-	-
			IX	7F		5	-	-	-	-	-
CMP opr	Compare accumulator contents with memory contents	(A) - (M)	IMM	A1	ii	2	-	-	↓	↓	↓
			DIR	B1	dd	3	-	-	-	-	-
			EXT	C1	hh ll	4	-	-	-	-	-
			IX2	D1	ee ff	5	-	-	-	-	-
			IX1	E1	ff	4	-	-	-	-	-
IX	F1		3	-	-	-	-	-			
COM opr COMA COMX COM opr COM opr	Complement register contents (ones complement)	M ← M = \$FF - (M) A ← A = \$FF - (A) X ← X = \$FF - (X) M ← M = \$FF - (M) M ← M = \$FF - (M)	DIR	33	dd	5	-	-	↓	↓	1
			INH	43		3	-	-	-	-	-
			INH	53		3	-	-	-	-	-
			IX1	63	ff	6	-	-	-	-	-
			IX	73		5	-	-	-	-	-
CPX opr	Compare index register contents with memory contents	(X) - (M)	IMM	A3	ii	2	-	-	↓	↓	↓
			DIR	B3	dd	3	-	-	-	-	-
			EXT	C3	hh ll	4	-	-	-	-	-
			IX2	D3	ee ff	5	-	-	-	-	-
			IX1	E3	ff	4	-	-	-	-	-
IX	F3		3	-	-	-	-	-			
DEC opr DECA DECX DEC opr DEC opr	Decrement register contents	M ← (M) - 1 A ← (A) - 1 X ← (X) - 1 M ← (M) - 1 M ← (M) - 1	DIR	3A	dd	5	-	-	↓	↓	-
			INH	4A		3	-	-	-	-	-
			INH	5A		3	-	-	-	-	-
			IX1	6A	ff	6	-	-	-	-	-
			IX	7A		5	-	-	-	-	-

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

Source Form(s)	Operation	Description	Addressing Mode for Operand	Machine Coding (hexadecimal)		Cycles	Condition Code				
				Opcode	Operand		H	I	N	Z	C
EOR opr	Exclusive OR accumulator contents with memory contents	$A \leftarrow (A) \oplus (M)$	IMM	A8	ii	2	-	-	↕	↕	-
			DIR	B8	dd	3					
			EXT	C8	hh ll	4					
			IX2	D8	ee ff	5					
			IX1	E8	ff	4					
			IX	F8		3					
INC opr INCA INCX INC opr INC opr	Increment memory or register contents	$M \leftarrow (M) + 1$ $A \leftarrow (A) + 1$ $X \leftarrow (X) + 1$ $M \leftarrow (M) + 1$ $M \leftarrow (M) + 1$	DIR	3C	dd	5	-	-	↕	↕	-
			INH	4C		3					
			INH	5C		3					
			IX1	6C	ff	6					
			IX	7C		5					
JMP opr	Unconditional jump	$PC \leftarrow \text{jump address}$	DIR	BC	dd	2	-	-	-	-	-
			EXT	CC	hh ll	3					
			IX2	DC	ee ff	4					
			IX1	EC	ff	3					
			IX	FC		2					
JSR opr	Jump to subroutine	$PC \leftarrow (PC) + n$ ($n = 1, 2, \text{ or } 3$) Push (PCL); $SP \leftarrow (SP) - 1$ Push (PCH); $SP \leftarrow (SP) - 1$ $PC \leftarrow \text{conditional address}$	DIR	BD	dd	5	-	-	-	-	-
			EXT	CD	hh ll	6					
			IX2	DD	ee ff	7					
			IX1	ED	ff	6					
			IX	FD		5					
LDA opr	Load accumulator with memory contents	$A \leftarrow (M)$	IMM	A6	ii	2	-	-	↕	↕	-
			DIR	B6	dd	3					
			EXT	C6	hh ll	4					
			IX2	D6	ee ff	5					
			IX1	E6	ff	4					
			IX	F6		3					
LDX opr	Load index register with memory contents	$X \leftarrow (M)$	IMM	AE	ii	2	-	-	↕	↕	-
			DIR	BE	dd	3					
			EXT	CE	hh ll	4					
			IX2	DE	ee ff	5					
			IX1	EE	ff	4					
			IX	FE		3					
LSL opr LSLA LSLX LSL opr LSL opr	Logical shift left		DIR	38	dd	5	-	-	↕	↕	↕
			INH	48		3					
			INH	58		3					
			IX1	68	ff	6					
			IX	78		5					
LSR opr LSRA LSRX LSR opr LSR opr	Logical shift right		DIR	34	dd	5	-	-	0	↕	↕
			INH	44		3					
			INH	54		3					
			IX1	64	ff	6					
			IX	74		5					
MUL	Unsigned multiply	$X : A \leftarrow (X) \times (A)$	INH	42		11	0	-	-	-	0
NEG opr NEGA NEGX NEG opr NEG opr	Negate memory or register contents (two's complement)	$M \leftarrow \neg(M) = \$00 - (M)$ $A \leftarrow \neg(A) = \$00 - (A)$ $X \leftarrow \neg(X) = \$00 - (X)$ $M \leftarrow \neg(M) = \$00 - (M)$ $M \leftarrow \neg(M) = \$00 - (M)$	DIR	30	dd	5	-	-	↕	↕	↕
			INH	40		3					
			INH	50		3					
			IX1	60	ff	6					
			IX	70		5					
NOP	No operation		INH	9D		2	-	-	-	-	-
ORA opr	Inclusive OR accumulator contents with memory contents	$A \leftarrow (A) + (M)$	IMM	AA	ii	2	-	-	↕	↕	-
			DIR	BA	dd	3					
			EXT	CA	hh ll	4					
			IX2	DA	ee ff	5					
			IX1	EA	ff	4					
			IX	FA		3					
ROL opr ROLA ROLX ROL opr ROL opr	Rotate left through carry		DIR	39	dd	5	-	-	↕	↕	↕
			INH	49		3					
			INH	59		3					
			IX1	69	ff	6					
			IX	79		5					

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

Source Form(s)	Operation	Description	Addressing Mode for Operand	Machine Coding (hexadecimal)		Cycles	Condition Code				
				Opcode	Operand		H	I	N	Z	C
ROR opr RORA RORX ROR opr ROR opr	Rotate right through carry		DIR INH INH IX1 IX	36 46 56 66 76	dd ff	5 3 3 6 5	-	-	↕	↕	↕
RSP	Reset stack pointer	$SP \leftarrow \$00FF$	INH	9C		2	From Stack				
RTI	Return from interrupt	$SP \leftarrow (SP) + 1$; pull (CCR) $SP \leftarrow (SP) + 1$; pull (A) $SP \leftarrow (SP) + 1$; pull (X) $SP \leftarrow (SP) + 1$; pull (PCH) $SP \leftarrow (SP) + 1$; pull (PCL)	INH	80		9	↕	↕	↕	↕	↕
RTS	Return from subroutine	$SP \leftarrow (SP) + 1$; pull (PCH) $SP \leftarrow (SP) + 1$; pull (PCL)	INH	81		6	-	-	-	-	-
SBC opr	Subtract memory contents and carry bit from accumulator contents	$A \leftarrow (A) - (M) - C$	IMM DIR EXT IX2 IX1 IX	A2 B2 C2 D2 E2 F2	ii dd hh ll ee ff ff	2 3 4 5 4 3	-	-	↕	↕	↕
SEC	Set carry bit	$C \leftarrow 1$	INH	99		2	-	-	-	-	1
SEI	Set interrupt mask	$I \leftarrow 1$	INH	9B		2	-	1	-	-	-
STA opr	Store accumulator contents in memory	$M \leftarrow (A)$	DIR EXT IX2 IX1 IX	B7 C7 D7 E7 F7	dd hh ll ee ff ff	4 5 6 5 4	-	-	↕	↕	-
STOP	Enable IRQ; stop oscillator		INH	8E		2	-	0	-	-	-
STX opr	Store index register contents in memory	$M \leftarrow (X)$	DIR EXT IX2 IX1 IX	BF CF DF EF FF	dd hh ll ee ff ff	4 5 6 5 4	-	-	↕	↕	-
SUB opr	Subtract memory contents from accumulator contents	$A \leftarrow (A) - (M)$	IMM DIR EXT IX2 IX1 IX	A0 B0 C0 D0 E0 F0	ii dd hh ll ee ff ff	2 3 4 5 4 3	-	-	↕	↕	↕
SWI	Software interrupt	$PC \leftarrow (PC) + 1$; push (PCL) $SP \leftarrow (SP) - 1$; push (PCH) $SP \leftarrow (SP) - 1$; push (X) $SP \leftarrow (SP) - 1$; push (A) $SP \leftarrow (SP) - 1$; push (CCR) $SP \leftarrow (SP) - 1$; $I \leftarrow 1$ $PCH \leftarrow$ Interrupt vector hi byte $PCL \leftarrow$ Int. vector low byte	INH	83		10	-	1	-	-	-
TAX	Transfer accumulator contents to index register	$X \leftarrow (A)$	INH	97		2	-	-	-	-	-
TST opr TSTA TSTX TST opr TST opr	Test memory, accumulator, or index register contents for negative or zero	$(M) - \$00$	DIR INH INH IX1 IX	3D 4D 5D 6D 7D	dd ff	4 3 3 5 4	-	-	↕	↕	0
TXA	Transfer index register contents to accumulator	$A \leftarrow (X)$	INH	9F		2	-	-	-	-	-
WAIT	Enable interrupts; halt CPU		INH	8F		2	-	0	-	-	-

4.4.7 Opcode Map

Table 4-13 is an opcode map for the MC68HC05P8 instructions.

Table 4-13. Opcode Map

Bit-Manipulation			Branch			Read-Modify-Write			Control			Register/Memory							
DIR	DIR	REL	DIR	DIR	REL	DIR	INH	INH	IX1	IX	INH	INH	IMM	DIR	EXT	IX2	IX1	IX	
HI	0	0000	1	0001	0010	3	0011	5	0101	6	7	8	A	B	C	D	E	F	LO
0	5	BSET0	5	NEG	5	NEG	5	NEG	5	RTI	9	1000	1010	SUB	SUB	SUB	SUB	SUB	0
0000	3	DIR 2	2	DIR 1	1	INH	1	INH	1	INH	1	INH	2	IMM 2	EXT 3	IX2 2	IX1 1	IX	0000
1	5	BCLR0	5	BRN	3	REL	3	RTS	6	1	INH	1	CMP	CMP	CMP	CMP	CMP	CMP	1
0001	3	DIR 2	2	REL	3	REL	3	MUL	11	1	INH	1	2	IMM 2	EXT 3	IX2 2	IX1 1	IX	0001
2	5	BSET1	5	BHI	3	REL	3	1	INH	1	INH	1	SBC	SBC	SBC	SBC	SBC	SBC	2
0010	3	DIR 2	2	REL	3	REL	3	COMA	3	5	COM	5	2	IMM 2	EXT 3	IX2 2	IX1 1	IX	0010
3	5	BCLR1	5	BLS	3	REL	3	1	INH	1	INH	1	CPX	CPX	CPX	CPX	CPX	CPX	3
0011	3	DIR 2	2	REL	2	REL	2	1	INH	1	INH	1	2	IMM 2	EXT 3	IX2 2	IX1 1	IX	0011
4	5	BSET2	5	BCC	3	REL	3	LSRX	3	6	LSR	6	AND	AND	AND	AND	AND	AND	4
0100	3	DIR 2	2	REL	2	REL	2	1	INH	1	INH	1	2	IMM 2	EXT 3	IX2 2	IX1 1	IX	0100
5	5	BCLR2	5	BCS	3	REL	3	1	INH	1	INH	1	BIT	BIT	BIT	BIT	BIT	BIT	5
0101	3	DIR 2	2	REL	2	REL	2	1	INH	1	INH	1	2	IMM 2	EXT 3	IX2 2	IX1 1	IX	0101
6	5	BSET3	5	BNE	3	REL	3	ROR	3	6	ROR	6	LDA	LDA	LDA	LDA	LDA	LDA	6
0110	3	DIR 2	2	REL	2	REL	2	1	INH	1	INH	1	2	IMM 2	EXT 3	IX2 2	IX1 1	IX	0110
7	5	BCLR3	5	BEQ	3	REL	3	ASRX	3	6	ASR	6	STA	STA	STA	STA	STA	STA	7
0111	3	DIR 2	2	REL	2	REL	2	1	INH	1	INH	1	2	IMM 2	EXT 3	IX2 2	IX1 1	IX	0111
8	5	BSET4	5	BHCC	3	REL	3	LSLX	3	6	LSL	6	EOR	EOR	EOR	EOR	EOR	EOR	8
1000	3	DIR 2	2	REL	2	REL	2	1	INH	1	INH	1	2	IMM 2	EXT 3	IX2 2	IX1 1	IX	1000
9	5	BCLR4	5	BHCS	3	REL	3	ROL	3	6	ROL	6	ADC	ADC	ADC	ADC	ADC	ADC	9
1001	3	DIR 2	2	REL	2	REL	2	1	INH	1	INH	1	2	IMM 2	EXT 3	IX2 2	IX1 1	IX	1001
A	5	BSET5	5	BPL	3	REL	3	DECX	3	6	DEC	6	ORA	ORA	ORA	ORA	ORA	ORA	A
1010	3	DIR 2	2	REL	2	REL	2	1	INH	1	INH	1	2	IMM 2	EXT 3	IX2 2	IX1 1	IX	1010
B	5	BCLR5	5	BMI	3	REL	3	1	INH	1	INH	1	2	IMM 2	EXT 3	IX2 2	IX1 1	IX	1011
1011	3	DIR 2	2	REL	2	REL	2	1	INH	1	INH	1	2	IMM 2	EXT 3	IX2 2	IX1 1	IX	1011
C	5	BSET6	5	BMC	3	REL	3	INCA	3	6	INC	6	JMP	JMP	JMP	JMP	JMP	JMP	C
1100	3	DIR 2	2	REL	2	REL	2	1	INH	1	INH	1	2	IMM 2	EXT 3	IX2 2	IX1 1	IX	1100
D	5	BCLR6	5	BMS	3	REL	3	TSTA	3	6	TST	6	BSR	BSR	BSR	BSR	BSR	BSR	D
1101	3	DIR 2	2	REL	2	REL	2	1	INH	1	INH	1	2	IMM 2	EXT 3	IX2 2	IX1 1	IX	1101
E	5	BSET7	5	BIL	3	REL	3	1	INH	1	INH	1	2	IMM 2	EXT 3	IX2 2	IX1 1	IX	1110
1110	3	DIR 2	2	REL	2	REL	2	1	INH	1	INH	1	2	IMM 2	EXT 3	IX2 2	IX1 1	IX	1110
F	5	BCLR7	5	BIH	3	REL	3	CLRX	3	6	CLR	6	STX	STX	STX	STX	STX	STX	F
1111	3	DIR 2	2	REL	2	REL	2	1	INH	1	INH	1	2	IMM 2	EXT 3	IX2 2	IX1 1	IX	1111

LEGEND

F	High Byte of Opcode in Hexadecimal
1111	High Byte of Opcode in Binary
0	Low Byte of Opcode in Hexadecimal
SUB	Low Byte of Opcode in Binary
1	Number of Bytes/Addressing Mode
IX	Number of Cycles
0000	Opcode Mnemonic
1	Low Byte of Opcode in Binary

ABBREVIATIONS FOR ADDRESSING MODES

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

4.5 Low-Power Modes

The following paragraphs describe the STOP and WAIT modes. (See also 6.2 Data-Retention Mode.)

4.5.1 STOP Mode

The STOP instruction places the MCU in its lowest power-consumption mode. In STOP mode, the internal oscillator turns off, halting all internal processing including timer operation and computer operating properly (COP) timeout operation. (See Figure 4-8.)

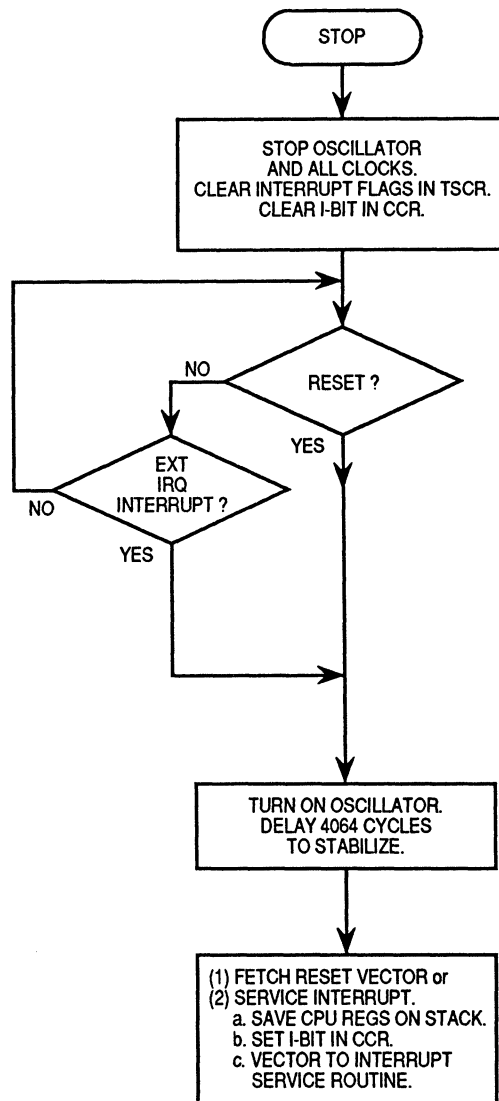


Figure 4-8. STOP Function Flowchart

During STOP mode, the timer overflow enable (TOFE) and real-time interrupt enable (RTIE) bits in the timer status and control register (TSCR) are cleared to remove any pending timer interrupt request and to disable any further timer interrupts. The timer prescaler is cleared. The interrupt mask (I-bit) in the condition code register is cleared to enable external interrupts. All other registers and memory locations remain unchanged. All I/O lines remain unchanged. The MCU can be brought out of STOP mode only by an external interrupt or a reset. An external interrupt automatically loads the program counter with the contents of \$1FFA and \$1FFB, the location of the vector address of the interrupt service routine. A reset automatically loads the program counter with the contents of \$1FFE and \$1FFF, the location of the vector address of the reset service routine. (See also **7.5 Timer During STOP Mode** and **8.5 A/D Converter During STOP Mode**.)

4.5.2 WAIT Mode

The WAIT instruction places the MCU in an intermediate power-consumption mode. All CPU action stops, but the timer remains active. A timer interrupt can cause the MCU to exit WAIT mode. (See Figure 4-9.)

The computer operating properly (COP) watchdog is not disabled in WAIT mode. The user should exit from WAIT and reset the COP timer before timeout to prevent a watchdog reset.

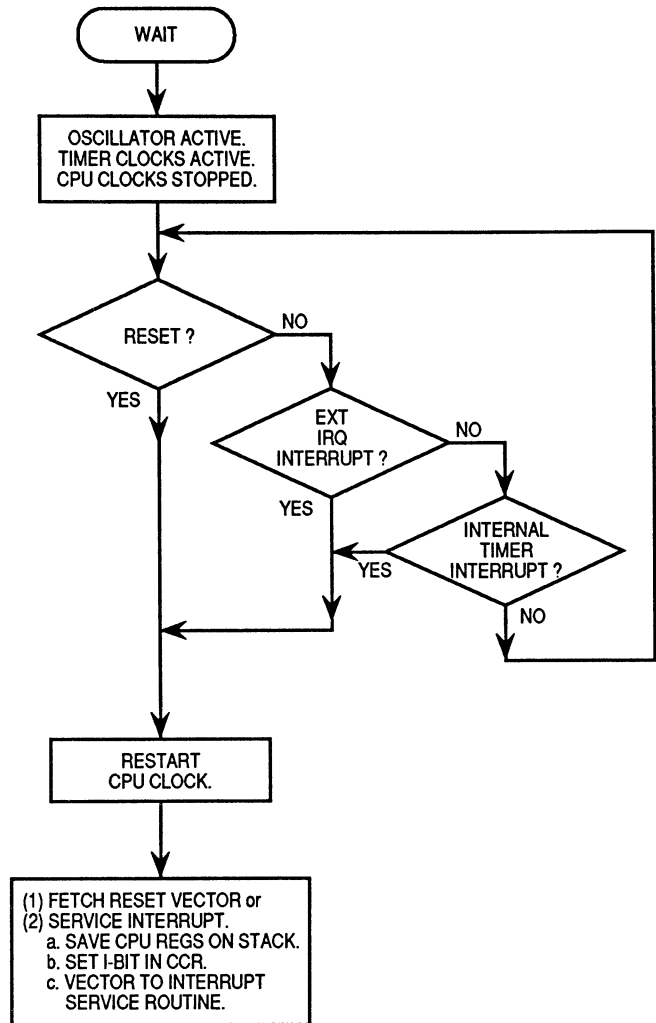


Figure 4-9. WAIT Function Flowchart

During WAIT mode, the interrupt mask (I-bit) in the condition code register is cleared to enable interrupts. All other registers, memory locations, and I/O lines remain in their previous states. (See also 7.4 Timer During WAIT Mode and 8.4 A/D Converter During WAIT Mode.)

SECTION 5 RESETS AND INTERRUPTS

This section describes the four ways that the CPU can be reset and the three kinds of interrupts.

5.1 Resets

A reset immediately stops execution of the current instruction. A reset forces the internal address bus to a known starting address and forces certain control and status bits to known conditions. The CPU can be reset in the following four ways:

- Initial power-up (power-on reset)
- An external, logical zero signal on the reset pin ($\overline{\text{RESET}}$)
- Timeout of the computer operating properly (COP) watchdog timer
- Illegal address fetch

NOTE

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

The following internal actions occur as a result of any reset:

- All data direction register bits are cleared to logical zero, so that the corresponding I/O pins become high-impedance inputs.
- The stack pointer is loaded with \$FF.
- The interrupt mask (I-bit) is set, inhibiting interrupts, and the $\overline{\text{IRQ}}$ request latch is cleared.
- The timer divide-by-four prescaler is cleared.
- The timer is cleared.
- RT1 and RT0 are set to logical one, selecting the lowest real-time interrupt rate.
- The timer interrupt enable bits (TOFE and RTIE) and the timer interrupt flags (TOF and RTIF) are cleared to disable timer interrupts.

- The A/D status and control register (ADSCR) is cleared, disabling the A/D converter.
- The programming register (PROG) is cleared.
- The STOP latch is cleared to enable MCU clocks.
- The WAIT latch is cleared to wake the CPU from the WAIT mode.
- On exit from reset, the program counter is loaded with the user-defined reset vector address; the high byte of the program counter is loaded with the contents of location \$1FFE, and the low byte of the program counter is loaded from location \$1FFF.

5.1.1 Power-On Reset (POR)

A reset is generated on power-up when a positive transition occurs on V_{DD} . The power-on reset is strictly for power turn-on conditions and cannot be used to detect a drop in the power supply voltage.

To allow the on-chip oscillator to stabilize, there is a $4064 t_{cyc}$ (internal clock cycle) delay after the oscillator becomes active. If the \overline{RESET} pin is at logical zero at the end of $4064 t_{cyc}$, the CPU remains in the reset condition until \overline{RESET} goes to logical one.

5.1.2 External \overline{RESET} Input

The CPU is reset when a logical zero is applied to the \overline{RESET} pin for a period of one and one-half internal clock cycles (t_{cyc}). The \overline{RESET} input consists of a Schmitt trigger that senses the logic level at the \overline{RESET} pin.

\overline{RESET} is an input-only pin and does not become active (go to logical zero) when a power-on reset or computer operating properly watchdog reset is generated.

5.1.3 Computer Operating Properly (COP) Reset

The MCU contains a watchdog timer as a factory-set mask option that automatically times out if not cleared within a specific time by a program sequence. The COP watchdog timer system is used to detect software errors. If the COP watchdog timer is allowed to time out, a reset is generated. The COP is implemented with an 18-stage ripple counter that provides a choice of four timeout periods. The minimum COP timeout period ranges from 57.4 ms to 458.5 ms at an internal operating frequency (f_{op}) of 2.0 MHz. (See **SECTION 7 TIMER**.) A COP timeout resets and reinitializes the CPU in the same fashion as a power-on reset or external reset. A COP reset is prevented by writing a logical zero to bit 0 (COPR) of the COP control register at location \$1FF0.

Writing a logical zero to COPR resets the last three stages of the counter and begins the timeout period again.

The COP register is a write-only register that is used to prevent a COP watchdog timeout. Reading this location returns the contents of a ROM location. Figure 5-1 shows the COP register.

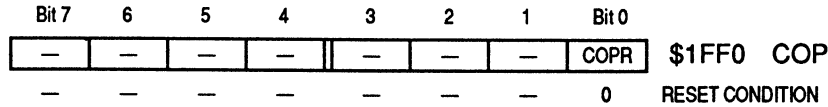


Figure 5-1. COP Control Register

COPR — COP Timer Reset

Periodically writing a logical zero to COPR prevents the COP watchdog timer from resetting the CPU.

5.1.4 Illegal Address Reset

When an opcode fetch occurs at an address which is not in the EEPROM (\$30–\$4F), the RAM (\$90–\$FF), or the ROM (\$1680–\$1FFF), the CPU is automatically reset.

5.2 Interrupts

An interrupt temporarily stops normal processing so that some exceptional event can be processed. Unlike a reset, an interrupt does not stop the current instruction. An interrupt is considered pending until the current instruction is complete. There are three kinds of CPU interrupts:

- External interrupt — If the interrupt mask (I-bit) is a logical zero, and the external interrupt pin (\overline{IRQ}) goes to logical zero, then the CPU recognizes an external interrupt.
- Timer interrupt — When the interrupt mask is a logical zero, the CPU can recognize interrupts from the timer. A timer interrupt is requested if one of the two timer interrupt flags (TOF, RTIF) goes to logical one while its corresponding timer interrupt enable bit (TOFE, RTIE) is a logical one. The timer interrupt flags and the timer interrupt enable bits are in the timer status and control register (TSCR). (See **SECTION 7 TIMER.**)
- Software interrupt — The software interrupt is an executable instruction. It is executed regardless of the state of the interrupt mask.

The following internal actions occur as a result of any interrupt:

- CPU register contents are stored on the stack in the order PCL, PCH, X, A, CCR.
- The interrupt mask (I-bit) is automatically set to prevent additional interrupts.
- An interrupt vector is fetched that causes processing to continue at the starting address of the interrupt routine.
- The RTI (return from interrupt) instruction causes the register contents to be recovered from the stack in the order CCR, A, X, PCH, PCL. Normal processing resumes.

Figure 5-2 shows the stacking and recovery sequence.

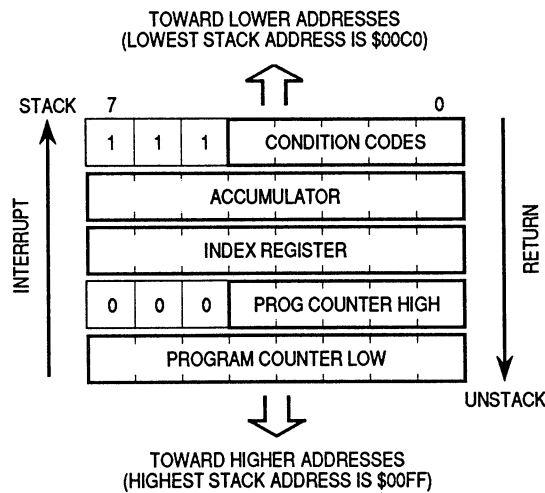


Figure 5-2. Interrupt Stacking Order

As each instruction is completed, the CPU checks for the presence of enabled external interrupt requests and enabled timer interrupt requests. For an external interrupt request to be recognized, the interrupt mask (I-bit) in the CCR must be a logical zero. If the interrupt mask is set or if no qualified interrupt request is pending, the CPU fetches and executes the next program instruction.

For a timer interrupt request to be recognized, the interrupt mask must be a logical zero, and the corresponding timer interrupt enable bit (RTIE or TOFE) in the timer status and control register (TSCR) must be a logical one. If the interrupt mask is set or if no qualified interrupt request is pending, the CPU fetches and executes the next program instruction.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first.

A software interrupt (SWI) is executed as an instruction, regardless of the state of the interrupt mask. Figure 5-3 shows how interrupts relate to normal instruction execution. CPU control logic determines the sequence of operations.

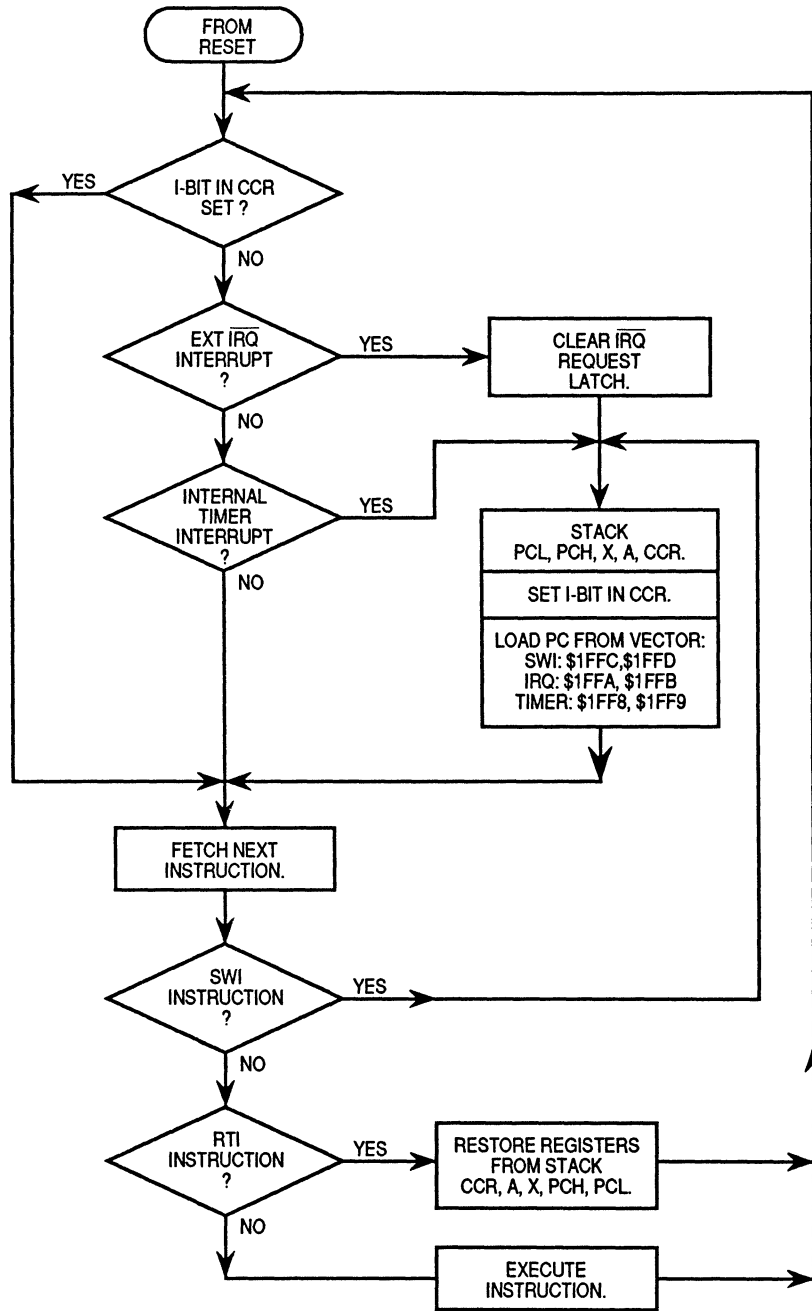


Figure 5-3. Reset and Interrupt Flowchart

5.2.1 External Interrupt

The CPU recognizes an external interrupt when the external interrupt pin (\overline{IRQ}) goes to a logical zero while the interrupt mask (I-bit) is a logical zero. A small synchronization delay occurs, and a logical one is latched internally to signify that an external interrupt is requested. When the CPU completes its current instruction, the interrupt latch is tested. If the interrupt latch contains a logical one, and the interrupt mask in the condition code register is a logical zero, the CPU then begins the interrupt sequence. The current state of the CPU is pushed onto the stack, and the interrupt mask is set to inhibit further interrupts until the present one is serviced. The address of the interrupt service routine is contained in memory locations \$1FFA and \$1FFB.

Both an edge-sensitive and level-sensitive interrupt trigger and an edge-sensitive-only interrupt trigger are available as factory-set mask options. Figure 5-4 shows the internal logic of this mask option. The interrupt latch is cleared while the interrupt vector is being fetched. During the interrupt service routine, a new external interrupt request can be initiated and latched. As soon as the interrupt mask is cleared (usually during the return from interrupt), the latched request is recognized and serviced.

The level-sensitive trigger option allows multiple external interrupt sources to be wire-ORed to the \overline{IRQ} pin. As long as any source is holding the \overline{IRQ} pin at logical zero, an external interrupt request is considered to be pending.

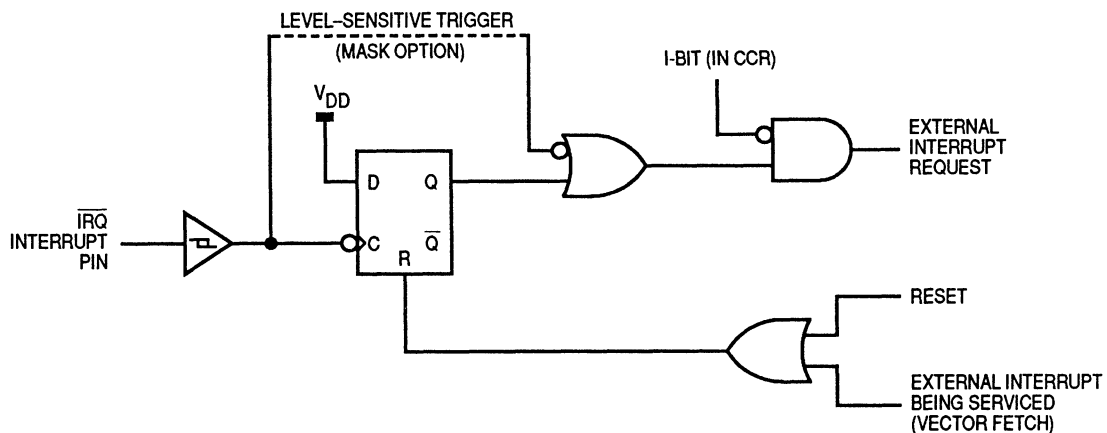


Figure 5-4. \overline{IRQ} Mask Option Logic

5.2.2 Software Interrupt (SWI)

The SWI is an executable instruction. The SWI instruction is executed regardless of the state of the interrupt mask (I-bit) in the CCR. The address of the SWI interrupt service routine is in memory locations \$1FFC and \$1FFD.

5.2.3 Timer Interrupt

Two interrupts can be generated by the timer when the interrupt mask (I-bit) is a logical zero. When one of the two timer interrupt flags in the timer status and control (TSCR) register is at logical one, and the corresponding timer interrupt enable flag is at logical one, the CPU recognizes a timer interrupt. (See **SECTION 7 TIMER** for more information.) Both of the timer interrupts use the same interrupt vector at \$1FF8 and \$1FF9.

SECTION 6 MEMORY

This section describes the organization of the on-chip memory. The MC68HC05P8 MCU can address 8K bytes of memory space.

6.1 Memory Map

The program counter normally advances one address at a time through the on-chip memory, reading the instructions and data necessary to execute the program. The ROM and EEPROM portions of memory hold the program instructions, user-defined vectors, and service routines. The RAM portion of memory holds variable data. I/O registers and status/control registers are memory-mapped so that the CPU can access their locations in the same way it accesses any other memory location.

6.1.1 ROM

On-chip ROM includes 2048 bytes of factory-programmed memory at addresses \$1680–\$1E7F for storage of application program instructions and fixed data. The last 16 memory addresses (\$1FF0–\$1FFF) are ROM addresses that contain user-defined vectors for servicing interrupts and resets. When ordering the MCU, the user specifies the instructions and data to be programmed into the user ROM.

The 368 bytes between \$1E80 and \$1FEF are reserved ROM addresses that contain the instructions for a series of self-check tests.

6.1.2 RAM

The MCU has 112 bytes of fully static read-write memory for storage of variable and temporary data during program execution. The top 64 RAM addresses (\$C0–\$FF) serve as the stack. The CPU uses the stack to save CPU register contents before processing an interrupt or subroutine call. The stack pointer decrements during pushes and increments during pulls.

The first 32 bytes of the memory space contain port data registers, port data direction registers, timer status/control and counter registers, and A/D status/control and data registers.

Figure 6-1 is a memory map of the MCU, and Figure 6-2 is a more detailed memory map of the 32-byte I/O register and status and control register area.

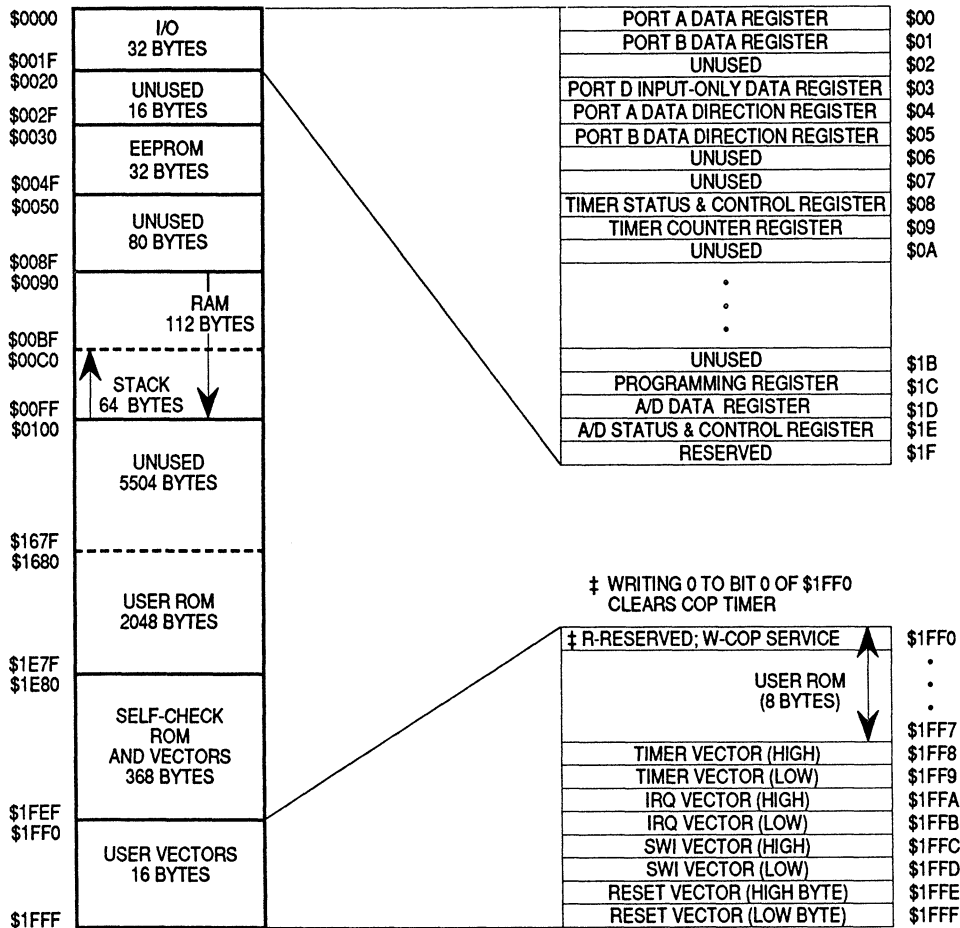


Figure 6-1. MC68HC05P8 Memory Map

NOTE

Using the stack area for data storage or as a temporary work area requires care to prevent data from being overwritten due to stacking from an interrupt or subroutine call.

	Bit 7	6	5	4	3	2	1	Bit 0	
\$0000	I/O	I/O	I/O	I/O	I/O	I/O	I/O	I/O	PORTA PORT A PIN NUMBERS (REF.) PORT A PIN NAMES (REF.)
	19 PA7	20 PA6	21 PA5	22 PA4	23 PA3	24 PA2	25 PA1	26 PA0	
\$0001	I/O	I/O	I/O	I/O	I/O	I/O	I/O	I/O	PORTB PORT B PIN NUMBERS (REF.) PORT B PIN NAMES (REF.)
	3 PB7	4 PB6	5 PB5	6 PB4	7 PB3	8 PB2	9 PB1	10 PB0	
\$0002									Unused
\$0003	-	-	-	-	I only	I only	I only	I only	PORTD PORT D PIN NUMBERS (REF.) PORT D PIN NAMES (REF.)
	-	-	-	-	15 PD3/AN3	16 PD2/AN2	17 PD1/AN1	18 PD0/AN0	
\$0004	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0	DDRA
\$0005	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0	DDRB
\$0006									Unused
\$0007									Unused
\$0008	TOF	RTIF	TOFE	RTIE	0	0	RT1	RT0	TSCR
\$0009	TCR7	TCR6	TCR5	TCR4	TCR3	TCR2	TCR1	TCR0	TCR
\$000A									Unused
\$000B									Unused
\$000C									Unused
\$000D									Unused
\$000E									Unused
\$000F									Unused
\$0010									Unused
\$0011									Unused
\$0012									Unused
\$0013									Unused
\$0014									Unused
\$0015									Unused
\$0016									Unused
\$0017									Unused
\$0018									Unused
\$0019									Unused
\$001A									Unused
\$001B									Unused
\$001C	LVPI	CPEN	0	ER1	ER0	LATCH	EERC	EEPGM	PROG
\$001D	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0	ADDR
\$001E	COCO	ADRC	ADON	0	0	0	CH1	CH0	ADSCR
\$001F									RESERVED
\$1FF0	-	-	-	-	-	-	-	COPR	COP READS ACCESS A ROM LOCATION; WRITES ACCESS THE COP WATCHDOG RESET LOGIC.

Figure 6-2. I/O and Control Register Summary

6.1.3 EEPROM

The 32 addresses \$30–\$4F are EEPROM (electrically erasable programmable read-only memory) locations for storage of application program instructions and fixed data. The EEPROM is erased and programmed under software control of the programming register (PROG) at location \$1C. PROG controls the on-chip charge pump that generates the erasing/programming voltage.

6.1.3.1 Programming Register (PROG)

The programming register contains all the control bits for programming and erasing the EEPROM. The low-voltage programming inhibit circuitry inhibits the use of the programming register when the supply voltage falls below a set level.

Bit 7	6	5	4	3	2	1	Bit 0	
LVPI	CPEN	0	ER1	ER0	LATCH	EERC	EEPGM	\$1C PROG
0	0	0	0	0	0	0	0	RESET CONDITION

Figure 6-3. Programming Register (PROG)

LVPI — Low-Voltage Programming Inhibit

This bit is automatically set and cleared by the LVPI circuit and is not writable. The function is active in STOP mode.

1 = V_{DD} is below V_{LVPI} . All other bits of the register are cleared, disabling the charge pump and preventing EEPROM programming.

0 = V_{DD} is above V_{LVPR} . All other bits of the register can be written to program or erase the EEPROM.

CPEN — Charge Pump Enable

This bit should be set with the LATCH bit, and is automatically cleared by the LVPI circuit when the LVPI bit is set. This bit should be cleared when the charge pump is not in use.

1 = Charge pump is enabled.

0 = Internal programming or erasure voltage is not available on-chip.

Bit 5 — Not used; always reads logical zero.

ER1, ER0 — Erase Mode Select

These readable and writable bits, cleared by a reset, are used to select one of four erasing modes, as shown in the following table. ER1 and

ER0 are cleared automatically when the LVPI bit is set. (See 6.1.3.2 EEPROM Erasure.)

Table 6-1. Erasing Modes

ER1	ER0	Erasing Mode
0	0	No Erasing
0	1	Byte Erase
1	0	Block Erase
1	1	Bulk Erase

LATCH — EEPROM Address and Data Bus Latch

This readable and writable bit is cleared by a reset, and is cleared automatically when the LVPI bit is set.

- 1 = EEPROM address and data bus are configured for programming. Writing to the array latches the data bus and address bus. Reading the array is inhibited if a write to the EEPROM space has taken place.
- 0 = EEPROM address bus and data bus are configured for normal operation.

EERC — EEPROM RC Oscillator Select

This readable and writable bit is cleared by a reset, and cleared automatically when the LVPI bit is set. EERC should be set when the internal clock frequency is below 1.5 MHz.

- 1 = EEPROM programming is driven by the on-chip RC oscillator instead of the internal clock. After setting EERC, wait a time t_{RCON} for the RC oscillator to stabilize. (See 10.7 Control Timing ($V_{DD} = 5.0$ Vdc.)
- 0 = EEPROM programming is driven by the internal clock.

EEPGM — EEPROM Programming Power Enable

Readable anytime, this bit can only be written if the LATCH bit is set and a write to the EEPROM has taken place. EEGPM is automatically cleared if the LVPI bit is set. EEGPM must be written to enable or disable the programming or erasing function. This is accomplished by turning the charge pump on and off. Pulsing of the programming voltage can be controlled internally using EEGPM.

- 1 = Charge pump is switched on.
- 0 = Charge pump is switched off.

6.1.3.2 EEPROM Erasure

The erased state of an EEPROM bit is logical one. Any EEPROM byte to be programmed should be erased first. There are four EEPROM erasing modes:

- No erase mode — EEPROM erasing is inhibited
- Byte erase mode — Erases one byte at a time
- Block erase mode — Erases one of four 8-byte blocks
- Bulk erase mode — Erases all 32 bytes of EEPROM

In byte erase mode, only the selected byte is erased. In block erase mode, erasing a byte erases the entire 8-byte block in which the byte resides. The four 8-byte blocks in the EEPROM space are at addresses \$30–\$37, \$38–\$3F, \$40–\$47, and \$48–\$4F. In bulk erase mode, erasing any byte erases the entire EEPROM. The erase mode selection bits are ER1 and ER0 in the programming register (PROG). (See **6.1.3.1 Programming Register (PROG)**). For the values of the time intervals named in the following procedures, see **10.7 Control Timing ($V_{DD} = 5.0$ Vdc)** and **10.8 Control Timing ($V_{DD} = 3.3$ Vdc)**.

To erase a byte of EEPROM, take the following steps:

1. Set the LATCH, CPEN, and ER0 bits to logical one.
2. Clear the ER1 bit.
3. Write to the address to be erased.
4. Set the EEPGM bit to logical one for a time $t_{E\text{BYT}}$ to apply the programming voltage.
5. Clear the EEPGM bit and wait a time t_{FPV} to allow the erasing voltage to fall.
6. Clear the LATCH and CPEN bits to free up the buses.
7. Clear all programming control bits.

To erase a block of EEPROM, take the following steps:

1. Set the LATCH, CPEN, and ER1 bits to logical one.
2. Clear the ER0 bit.
3. Write to any address in the block to be erased.
4. Set the EEPGM bit to logical one for a time $t_{E\text{BLOCK}}$ to apply the programming voltage.
5. Clear the EEPGM bit and wait a time t_{FPV} to allow the erasing voltage to fall.
6. Clear the LATCH and CPEN bits to free up the buses.
7. Clear all programming control bits.

To erase all of the device EEPROM, take the following steps:

1. Set the LATCH, CPEN, ER0, and ER1 bits to logical one.
2. Write to any EEPROM address.
3. Set the EEPGM bit to logical one for a time t_{EBULK} to apply the programming voltage.
4. Clear the EEPGM bit and wait a time t_{FPV} to allow the erasing voltage to fall.
5. Clear the LATCH and CPEN bits to free up the buses.
6. Clear all programming control bits.

6.1.3.3 EEPROM Programming

To program a byte of EEPROM, first erase the EEPROM, then take the following steps:

1. Set the LATCH and CPEN bits to logical one.
2. Clear the ER1 and ER0 bits.
3. Write the new data to the address to be programmed.
4. Set the EEPGM bit to logical one for a time t_{EPGM} to apply the programming voltage.
5. Clear the EEPGM bit and wait a time t_{FPV} to allow the programming voltage to fall.
6. Clear the LATCH and CPEN bits to free up the buses.
7. Clear all programming control bits.

6.1.3.4 Minimizing EEPROM Erase/Program Cycles

Each EEPROM byte should be set to all logical ones (erased) before it is programmed. To avoid erasing existing EEPROM bytes that are already in the erased state (\$FF), the following test can be applied to each EEPROM byte before programming:

Let EB be the existing EEPROM byte, and let PB be the data byte to be programmed.

If $PB \cdot \overline{EB} = 0$, then program the byte without erasing it first.

If $PB \cdot \overline{EB} \neq 0$, then erase the existing byte before programming it.

By testing the existing EEPROM byte before erasing and programming it, the number of erase/program cycles can be kept to a minimum.

6.1.3.5 Low-Voltage Programming Inhibit (LVPI) Circuit

The LVPI circuit prevents EEPROM programming and erasure whenever V_{DD} falls below V_{LVPI} , and enables EEPROM programming and erasure once V_{DD} returns to a level above V_{LVPR} . When V_{DD} falls below V_{LVPI} , the circuit sets the LVPI bit of the programming register, which in turn clears the other control bits (6–0) to disable the charge pump and prevent programming. During such low-voltage periods and after a reset, LVPI remains set until V_{DD} reaches V_{LVPR} , at which time LVPI is cleared, and the remaining control bits of the programming register can again be written.

The V_{DD} rise and fall slew rates (S_{VDDR} and S_{VDDF}) must be within the specifications described in **10.7 Control Timing ($V_{DD} = 5.0$ Vdc)** for proper operation. If the specification is not met, then the circuit will operate properly following a delay of $V_{DD} \div \text{slew rate}$.

The LVPI function is a mask option. If the mask option is not elected, then this function is disabled, and bit 7 of the programming register is set to logical zero. The LVPI function is active in both STOP and WAIT modes.

6.2 Data-Retention Mode

In data-retention mode, the MCU retains RAM contents and CPU register contents at V_{DD} voltages as low as 2.0 Vdc. The $\overline{\text{RESET}}$ line must be driven to logical zero before the V_{DD} voltage is lowered, and $\overline{\text{RESET}}$ must remain low continuously during data-retention mode. The data-retention feature allows the MCU to be left in a low power-consumption mode during which data is held, but the CPU cannot execute instructions. To exit the data-retention mode, V_{DD} must be returned to its normal operating voltage before $\overline{\text{RESET}}$ is returned to logical one.

SECTION 7 TIMER

This section describes the operation of the timer. The timer shown in Figure 7-1 provides a means to generate internal interrupts at selected rates. Timer features include the following:

- Timer overflow
- Four selectable interrupt rates
- Computer operating properly (COP) watchdog timer

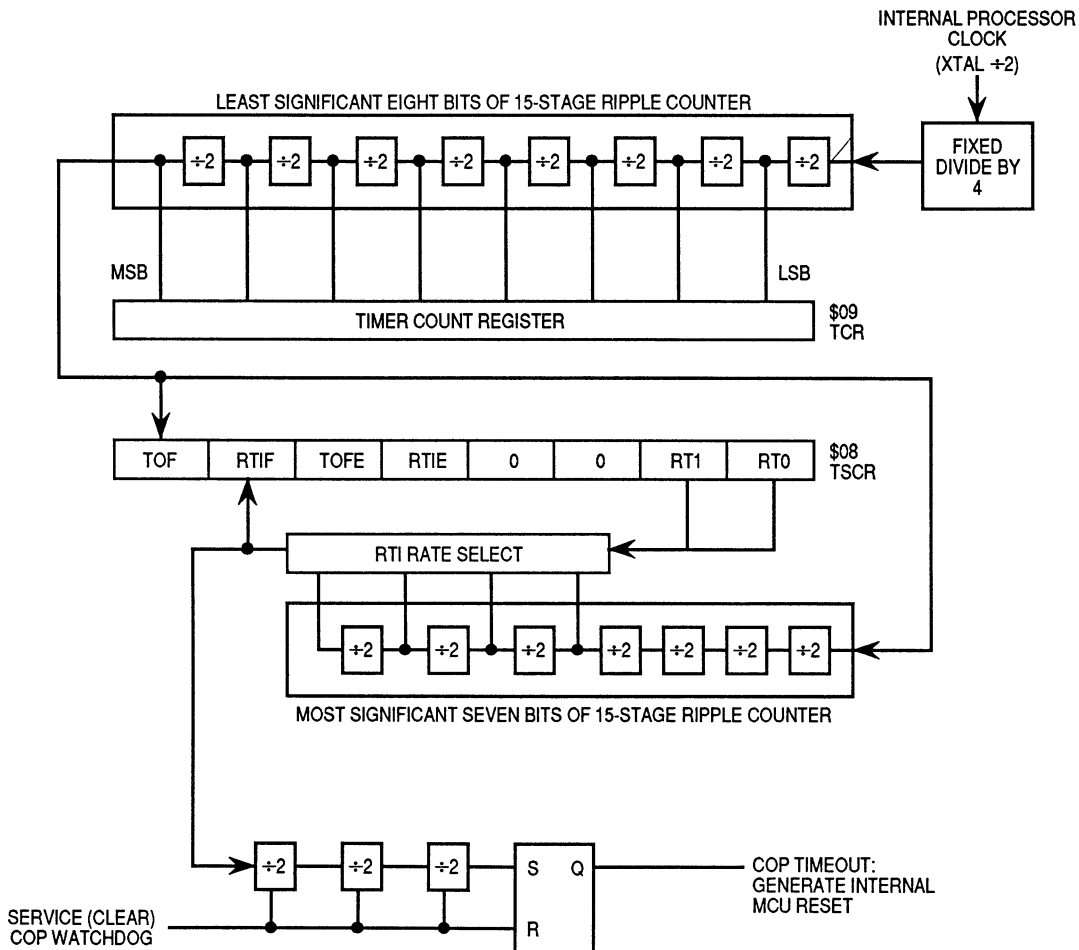


Figure 7-1. Timer Block Diagram

A 15-stage ripple counter, preceded by a prescaler that divides the internal clock signal by four, provides the timing reference for the timer functions. As Figure 7-1 shows, the value of the first eight timer stages can be read at any time by accessing the timer counter register (TCR) at address \$09. A timer overflow function at the eighth stage allows a timer interrupt every 1024 internal clock cycles. The next four stages lead to the real-time interrupt (RTI) circuit. The RT1 and RT0 bits in the timer status and control register (TSCR) at address \$08 allow a timer interrupt every 16,384, 32,768, 65,536, or 131,072 clock cycles. The last three stages drive the mask-optional COP system.

7.1 Timer Status and Control Register (TSCR)

TSCR controls and monitors how the timer generates and services timer interrupts. (See Figure 7-2.)

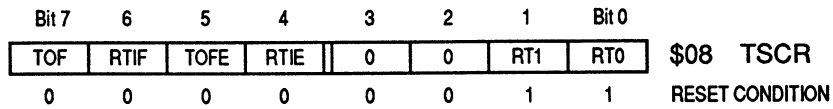


Figure 7-2. Timer Status and Control Register (TSCR)

TOF — Timer Overflow Flag

TOF is set when the first eight stages of the counter roll over from \$FF to \$00. Rollover also generates a timer interrupt request if the timer overflow enable bit (TOFE) is set. TOF is cleared by writing a logical zero to it. Writing a logical one to TOF has no effect.

RTIF — Real-Time Interrupt Flag

RTIF is set when the chosen RTI circuit output becomes active. RTIF also generates a timer interrupt request if the real-time interrupt enable bit (RTIE) is set. RTIF is cleared by a reset or by writing a logical zero to it. Writing a logical one to RTIF has no effect.

TOFE — Timer Overflow Enable

- 1 = TOF interrupt enabled
- 0 = TOF interrupt disabled

RTIE — Real-Time Interrupt Enable

- 1 = RTIF interrupt enabled
- 0 = RTIF interrupt disabled

Bits 3–2 — Not used; always read logical zero

RT1 and RT0 — Real-Time Interrupt Rate Select

The RT1 and RT0 bits allow changing the real-time interrupt rate by selecting one of the four outputs of the real-time interrupt circuit. Table 7-1 shows the available timer interrupt rates with a 2 MHz internal clock. A reset sets both bits, selecting the lowest periodic rate. Usually, the RTI rate is set one time in the reset initialization software. Unpredictable results may be obtained by altering this rate when the timeout period is imminent or uncertain. If the RTI circuit output is changed during a cycle in which the counter is switching, an RTIF could be missed, or an extra one could be generated. Because the COP timer is derived from this rate, changing the output also affects the COP system. The COP timer reset bit (COPR) should be cleared before changing RTI outputs. (See Table 7-1.)

Table 7-1. RTI and COP Reset Rates ($f_{op} = 2 \text{ MHz}$)

RT1	RT0	RTI Rate ($f_{op} = 2 \text{ MHz}$)	COP Timeout Period ($f_{op} = 2 \text{ MHz}$)
0	0	8.2 ms	57.3 ms
0	1	16.4 ms	114.7 ms
1	0	32.8 ms	229.4 ms
1	1	65.5 ms	458.8 ms

NOTE: f_{op} = internal operating frequency = $f_{osc} + 2$

7.2 COP Timer

Implemented as a mask option, the COP uses the output of the RTI circuit and divides it by eight. (See Figure 7-1.) Table 7-1 lists the minimum COP reset rates. If the COP circuit times out, a reset is generated and the normal reset vector is fetched. The user may prevent a COP timeout by writing a logical zero to bit 0 of address \$1FF0. When the COP is cleared, only the final three counter stages are cleared, which gives a tolerance of +0 to -1 RTI period for the COP timeout period.

7.3 Timer Counter Register (TCR)

TCR is an 8-bit read-only register that contains the current value of the first eight counter stages. The ripple counter is clocked at a rate of f_{op} divided by four and can be employed for various functions, including a software input capture. Extended time periods can be attained by using the TOF feature to increment a temporary RAM storage location, thereby simulating a 16-bit (or larger) counter.

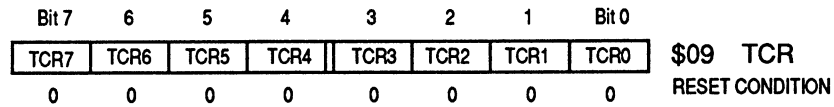


Figure 7-3. Timer Counter Register (TCR)

Power-on clears the entire counter chain and begins clocking the counter. After 4064 cycles, the power-on reset circuit is released, which again clears the counter chain and allows the device to come out of reset. At this point, if \overline{RESET} is not asserted, the timer starts counting up from zero and normal device operation begins. If \overline{RESET} is asserted at any time during operation, the counter chain is cleared.

7.4 Timer During WAIT Mode

The CPU clock halts during the WAIT mode, but the timer and computer operating properly (COP) watchdog timer remain active. If interrupts are not masked, a timer interrupt causes the CPU to exit WAIT mode.

7.5 Timer During STOP Mode

A STOP instruction clears the timer and the RTIF, RTIE, TOF, and TOFE bits in the TSCR. When STOP is exited by an external interrupt or a \overline{RESET} , the internal oscillator resumes operation. Following the 4064-cycle internal processor oscillator stabilization delay, the timer is cleared and normal operation continues.

SECTION 8

ANALOG-TO-DIGITAL CONVERTER

This section describes the A/D converter. The A/D converter system consists of a four-channel, multiplexed input and a successive-approximation A/D converter.

8.1 Conversion Process

The A/D conversion process is ratiometric, using two reference voltages, V_{RH} and V_{RL} . V_{RH} and V_{RL} may be any value between V_{DD} and V_{SS} as long as V_{RH} is greater than or equal to V_{RL} . Operation with $(V_{RH} - V_{RL})$ less than 2.5 V is not recommended. Conversion accuracy is tested and guaranteed for $V_{RH} = V_{DD}$ and $V_{RL} = V_{SS}$.

The reference voltages are applied to a precision internal digital-to-analog (D/A) converter. A multiplexer selects one of four analog input channels (AN3, AN2, AN1, or AN0) for sampling. A comparator successively compares the D/A converter output to the sampled analog input. Control logic changes the D/A converter input one bit at a time, starting with the MSB, until the D/A converter output matches the sampled analog input. The conversion is monotonic and has no missing codes.

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

The four pins of port D are input signals to the multiplexer. Each channel of conversion takes 32 internal clock cycles, and the clock frequency must be equal to or greater than 1 MHz. If the internal clock frequency is less than 1 MHz, the A/D internal RC oscillator (nominally 1.5 MHz) must be used for the A/D conversion clock. This selection is made by setting the ADRC bit in the A/D status and control register to logical one.

8.2 A/D Status and Control Register (ADSCR)

ADSCR contains a status flag and four writable control bits. A reset clears all five bits. (See Figure 8-1.)

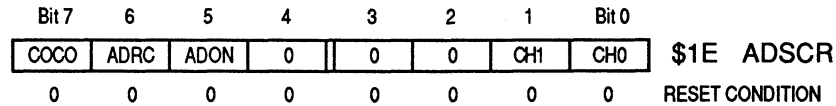


Figure 8-1. A/D Status and Control Register (ADSCR)

COCO — Conversion Complete

COCO is automatically set when an analog-to-digital conversion is complete, and a new result can be read from the A/D data register. COCO is automatically cleared when a new conversion begins or when ADSCR or the A/D data register (ADDR) is accessed. Writing to or reading ADSCR or ADDR starts a new conversion sequence. While COCO is a logical zero, the requested A/D result is not yet available in the A/D data register.

ADRC — A/D RC Oscillator Control

When the RC oscillator is turned on, it requires a time (t_{ADRC}) to stabilize, and results can be inaccurate during this time.

1 = Internal RC oscillator drives A/D converter

0 = Internal clock drives A/D converter

When the internal RC oscillator is being used as the A/D converter clock, two limitations apply:

- The conversion complete flag (COCO) must be used to determine when a conversion sequence has been completed because of the asynchronism between the RC oscillator and the internal clock.
- The conversion process runs at the nominal 1.5 MHz rate, but the conversion results must be transferred to the A/D data register synchronously with the internal clock; therefore, the conversion process is limited to a maximum of one channel per internal clock cycle.

ADON — A/D On

When the A/D is turned on, it requires a time (t_{ADON}) for the current sources to stabilize, and results can be inaccurate during this time.

1 = A/D converter enabled

0 = A/D converter disabled

Bits 4–3 — Not used; always read logical zero

Bit 2 — For factory use; normally reads zero

CH1, CH0 — Channel Select

These bits select one of the four A/D inputs (AN3, AN2, AN1, or AN0) for conversion. (See Table 8-1.)

Table 8-1. A/D Input Selection

CH1	CH0	Input Selected
0	0	AN0, Port D Bit 0
0	1	AN1, Port D Bit 1
1	0	AN2, Port D Bit 2
1	1	AN3, Port D Bit 3

NOTE

Using one or more of the port D pins as A/D converter inputs does not affect the ability to use the remaining port D pins as digital inputs.

Performing a digital read of port D with levels other than V_{DD} or V_{SS} on the inputs causes greater than normal power dissipation during the read and may give erroneous results.

8.3 A/D Data Register (ADDR)

ADDR is a read-only register that contains the result of the most recent A/D conversion. This register is updated each time the conversion complete flag (COCO) is set in the A/D status and control register.

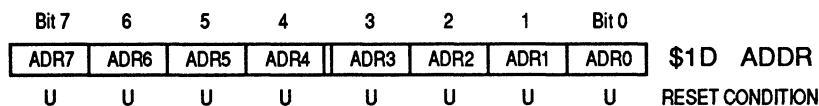


Figure 8-2. A/D Data Register (ADDR)

8.4 A/D Converter During WAIT Mode

If a conversion is in process when the MCU enters the WAIT mode, the A/D converter continues to operate normally. If the A/D converter is not being used, the ADON and ADRC bits in the A/D status and control register should be cleared to decrease power consumption during WAIT mode.

8.5 A/D Converter During STOP Mode

In STOP mode, the comparator and charge pump are disabled. Any conversion in progress or pending is aborted. When the internal clock begins running again as the MCU leaves STOP mode, built-in timing delays give the A/D circuits enough time to stabilize, and so no delays need to be written.

SECTION 9 SELF-CHECK MODE

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

9.1 Self-Check Circuit

The self-check function determines if the MCU is operating properly. The self-check circuit is shown in Figure 9-1. If $2 \times V_{DD}$ (V_{TST}) is applied to the \overline{IRQ} pin, and logical ones are applied to PB3–PB0, the MCU enters the self-check mode when reset. Port B pins PB3–PB0 are monitored for the self-check results. The following six tests are performed automatically in self-check mode:

- I/O — Functional test of ports A, B, and D
- RAM — Counter test for each RAM byte
- Timer — Test of timer counter register and status flags TOF and RTIF
- ROM — Checksum of entire ROM pattern
- Interrupts — Test of external and timer interrupts
- A/D converter — Conversion test of internal channels 4 and 6

9.2 Self-Check Results

Table 9-1 gives the codes displayed by the light-emitting diodes to indicate the self-check results.

Table 9-1. Self-Check Results

PB3	PB2	PB1	PB0	Remarks
1	0	0	1	Bad I/O
1	0	1	0	Bad RAM
1	0	1	1	Bad Timer
1	1	0	0	Bad ROM
1	1	0	1	Bad Interrupts or \overline{IRQ} Request
1	1	1	0	Bad A/D Converter
Flashing				Good Device
All Others				Bad Device

NOTE: Zero indicates LED is on; 1 indicates LED is off.

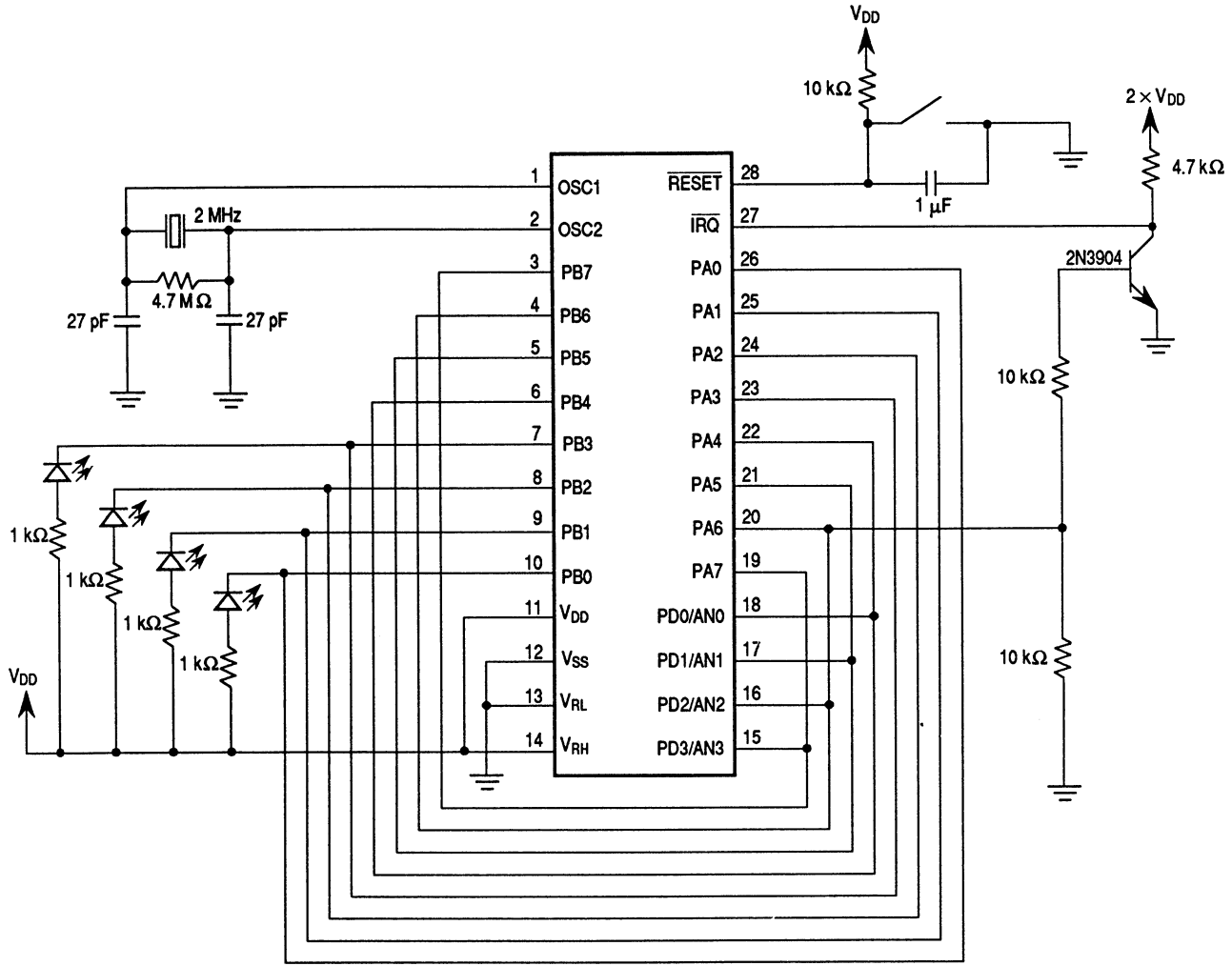


Figure 9-1. Self-Check Circuit

SECTION 10 ELECTRICAL SPECIFICATIONS

This section contains MCU electrical specifications and timing information.

10.1 Maximum Ratings

The MCU contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, precautions should be taken to avoid application of voltages higher than those shown in Table 10-1. For proper operation, it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in} \text{ or } V_{out}) \leq V_{DD}$. Reliability of operation is enhanced if unused inputs are connected to an appropriate logical voltage level (e.g., either V_{SS} or V_{DD}).

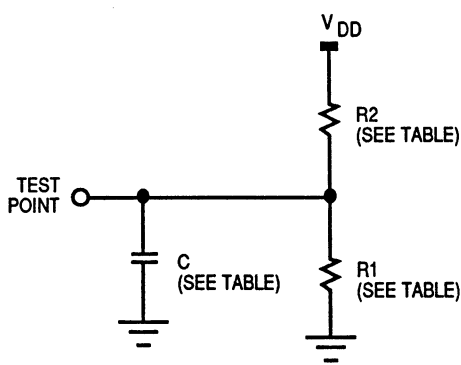
Table 10-1. Maximum Ratings

Rating	Symbol	Value	Unit
Supply voltage	V_{DD}	-0.3 to +7.0	V
Input voltage	V_{in}	$V_{SS} - 0.3$ to $V_{DD} + 0.3$	V
Self-check mode (IRQ pin only)	V_{in}	$V_{SS} - 0.3$ to $2 \times V_{DD} + 0.3$	V
Current drain per pin (excluding V_{DD} and V_{SS})	I	25	mA
Operating temperature range	T_A		$^{\circ}\text{C}$
MC68HC05P8 (Standard)		0 to +70	
MC68HC05P8 (Extended)		-40 to +85	
MC68HC05P8 (Automotive)		-40 to +125	
Storage temperature range	T_{stg}	-65 to +150	$^{\circ}\text{C}$

10.2 Thermal Characteristics

Table 10-2. Thermal Characteristics

Characteristic	Symbol	Value	Unit
Thermal resistance	$R_{\theta JA}$		$^{\circ}\text{C/W}$
Plastic		60	
SOIC		60	



V_{DD} = 4.5 V

Pins	R1	R2	C
PA7-PA0	3.26 kΩ	2.38 kΩ	50 pF
PB5-PB0	3.26 kΩ	2.38 kΩ	50pF

V_{DD} = 3.0 V

Pins	R1	R2	C
PA7-PA0	10.91 kΩ	6.32 kΩ	50 pF
PB7-PB5	10.91 kΩ	6.32 kΩ	50 pF

Figure 10-1. Test Load

10.3 Power Considerations

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

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

where:

T_A = Ambient temperature in °C

R_{θJA} = Package thermal resistance, junction-to-ambient in °C/W

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

P_{INT} = I_{CC} × V_{CC}, watts — chip internal power

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

For most applications P_{I/O} ≪ P_{INT} and can be neglected.

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

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

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

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

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

10.4 DC Electrical Characteristics ($V_{DD} = 5.0$ Vdc)
Table 10-3. DC Electrical Characteristics ($V_{DD} = 5.0$ Vdc)

 ($V_{DD} = 5.0$ Vdc \pm 10%, $V_{SS} = 0$ Vdc, $T_A = T_L$ to T_H , unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Output voltage ($I_{Load} \leq 10.0 \mu A$)	V_{OL} V_{OH}	– $V_{DD} - 0.1$	– –	0.1 –	V V
Output high voltage ($I_{Load} = -0.8$ mA) PA7–PA0, PB7–PB0	V_{OH}	$V_{DD} - 0.8$	–	–	V
Output low voltage ($I_{Load} = 1.6$ mA) PA7–PA0, PB7–PB0	V_{OL}	–	–	0.4	V
Input high voltage PA7–PA0, PB7–PB0, PD3–PD0, \overline{IRQ} , \overline{RESET} , OSC1	V_{IH}	$0.7 \times V_{DD}$	–	V_{DD}	V
Input low voltage PA7–PA0, PB7–PB0, PD3–PD0, \overline{IRQ} , \overline{RESET} , OSC1	V_{IL}	V_{SS}	–	$0.3 \times V_{DD}$	V
Low-voltage programming inhibit	VLVPI	3.5	–	–	V
Low-voltage programming recover	VLVPR	–	–	4.5	V
Low-voltage programming inhibit/recover hysteresis	HLVPI	0.1	–	1.0	V
Data-retention mode supply voltage (0 to 70°C)	V _{RM}	2.0			V
Supply current with LVPI enabled	I_{DD}				
RUN		–	3.5	5.0	mA
WAIT		–	1.6	2.75	mA
STOP					
25°C		–	150	TBD	μA
0 to 70°C (Standard)		–	–	–	–
–40 to + 85°C (Extended)		–	200	TBD	μA
–40 to 125°C (Automotive)		–	250	TBD	μA
Supply current with LVPI disabled	I_{DD}				
RUN		–	3.5	5.0	mA
WAIT		–	1.6	2.75	mA
STOP					
25°C		–	2.0	50	μA
0 to 70°C (Standard)		–	–	140	μA
–40 to + 85°C (Extended)		–	–	180	μA
–40 to 125°C (Automotive)		–	–	250	μA
I/O ports hi-Z leakage current PA7–PA0, PB7–PB0	I_{IL}	–	–	± 10	μA
Input current	I_{in}				
RESET, IRQ, OSC1		–	–	± 1	μA
PD3–PD0		–	–	400	nA
Capacitance					
Ports (as input or output)	C_{out} C_{in}	– –	– –	12 8	pF pF

NOTES:

1. Typical values at midpoint of voltage range, 25°C only.
2. WAIT I_{DD} : only timer system active.
3. RUN (operating) I_{DD} and WAIT I_{DD} measured using external square wave clock source ($f_{osc} = 4.2$ MHz), all inputs 0.2 V from rail; no dc loads, less than 50 pF on all outputs, $C_L = 20$ pF on OSC2.
4. WAIT I_{DD} and STOP I_{DD} : all ports configured as inputs, $V_{IL} = 0.2$ V, $V_{IH} = V_{DD} - 0.2$ V.
5. STOP I_{DD} measured with OSC1 = V_{SS} .
6. WAIT I_{DD} is affected linearly by the OSC2 capacitance.

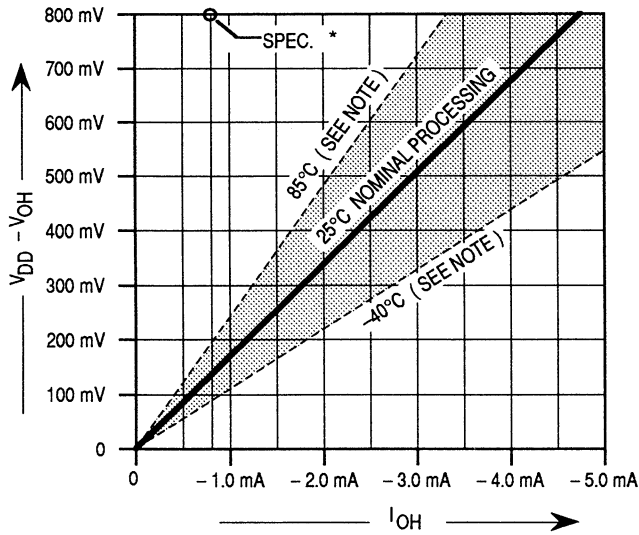
10.5 DC Electrical Characteristics (V_{DD} = 3.3 Vdc)
Table 10-4. DC Electrical Characteristics (V_{DD} = 3.3 Vdc)

 (V_{DD} = 3.3 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = T_L to T_H, unless otherwise noted)

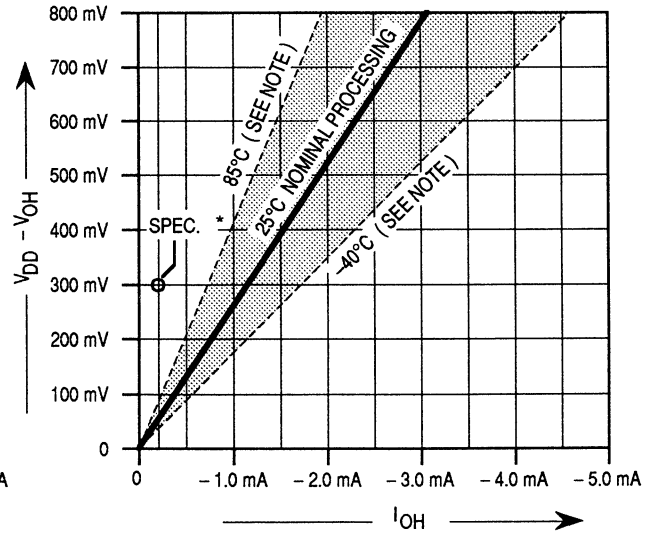
Characteristic	Symbol	Min	Typ	Max	Unit
Output voltage (I _{Load} ≤ 10.0 μA)	V _{OL} V _{OH}	– V _{DD} – 0.1	– –	0.1 –	V V
Output high voltage (I _{Load} = –0.2 mA) PA7–PA0, PB7–PB0	V _{OH}	V _{DD} – 0.3	–	–	V
Output low voltage (I _{Load} = 0.4 mA) PA7–PA0, PB7–PB0	V _{OL}	–	–	0.3	V
Input high voltage PA7–PA0, PB7–PB0, PD3–PD0, $\overline{\text{IRQ}}$, $\overline{\text{RESET}}$, OSC1	V _{IH}	0.7 × V _{DD}	–	V _{DD}	V
Input low voltage PA7–PA0, PB7–PB0, PD3–PD0, $\overline{\text{IRQ}}$, $\overline{\text{RESET}}$, OSC1	V _{IL}	V _{SS}	–	0.3 × V _{DD}	V
Data-retention mode supply voltage (0°C to 70°C)	V _{RM}	2.0			V
Supply current with LVPI enabled	I _{DD}				
RUN		–	1.0	2.5	mA
WAIT		–	0.8	1.4	mA
STOP					
25°C		–	100	TBD	μA
0 to 70°C (Standard)		–	–	–	–
–40 to + 85°C (Extended)		–	150	TBD	μA
–40 to 125°C (Automotive)		–	200	TBD	μA
Supply current with LVPI disabled	I _{DD}				
RUN		–	1.0	2.5	mA
WAIT		–	0.5	1.4	mA
STOP					
25°C		–	1.0	30	μA
0 to 70°C (Standard)		–	–	80	μA
–40 to + 85°C (Extended)		–	–	120	μA
–40 to 125°C (Automotive)		–	–	175	μA
I/O ports hi-Z leakage current PA7–PA0, PB7–PB0, PD3–PD0	I _{IL}	–	–	±10	μA
Input current	I _{in}				
$\overline{\text{RESET}}$, $\overline{\text{IRQ}}$, OSC1		–	–	±1	μA
PD3–PD0		–	–	400	nA
Capacitance					
Ports (as input or output)	C _{out}	–	–	12	pF
$\overline{\text{RESET}}$, $\overline{\text{IRQ}}$, PD3–PD0	C _{in}	–	–	8	pF

NOTES:

1. Typical values at midpoint of voltage range, 25°C only.
2. RUN (operating) I_{DD} and WAIT I_{DD} measured using external square wave clock source (f_{osc} = 2.0 MHz), all inputs 0.2 V from rail; no dc loads, less than 50 pF on all outputs, C_L = 20 pF on OSC2.
3. WAIT I_{DD} and STOP I_{DD}: all ports configured as inputs, V_{IL} = 0.2 V, V_{IH} = V_{DD} – 0.2 V.
4. STOP I_{DD} measured with OSC1 = V_{SS}.
5. WAIT I_{DD}: only timer system active.
6. WAIT I_{DD} is affected linearly by the OSC2 capacitance.



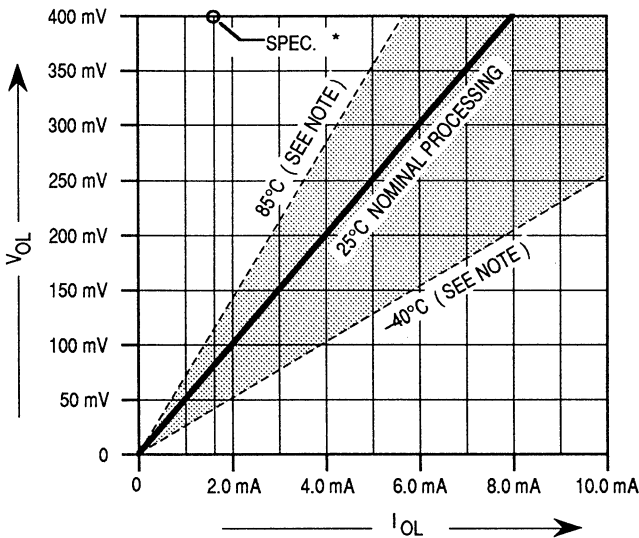
* At $V_{DD} = 5.0$ V, devices are specified and tested for $(V_{DD} - V_{OH}) \leq 800$ mV @ $I_{OH} = -0.8$ mA.



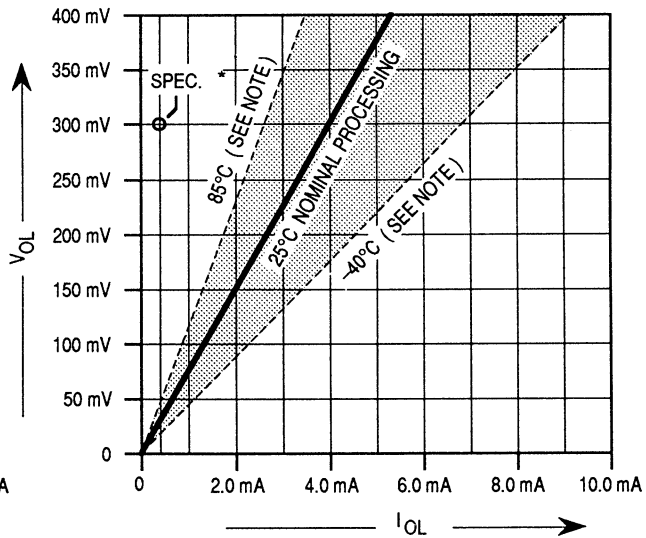
* At $V_{DD} = 3.3$ V, devices are specified and tested for $(V_{DD} - V_{OH}) \leq 300$ mV @ $I_{OH} = -0.2$ mA.

Shaded area indicates variation in driver characteristics due to changes in temperature and for normal processing tolerances. Within the limited range of values shown, V vs. I curves are approximately straight lines.

Figure 10-2. Typical High-Side Driver Characteristics



* At $V_{DD} = 5.0$ V, devices are specified and tested for $V_{OL} \leq 400$ mV @ $I_{OL} = 1.6$ mA.



* At $V_{DD} = 3.3$ V, devices are specified and tested for $V_{OL} \leq 300$ mV @ $I_{OL} = 0.4$ mA.

Shaded area indicates variation in driver characteristics due to changes in temperature and for normal processing tolerances. Within the limited range of values shown, V vs. I curves are approximately straight lines.

Figure 10-3. Typical Low-Side Driver Characteristics

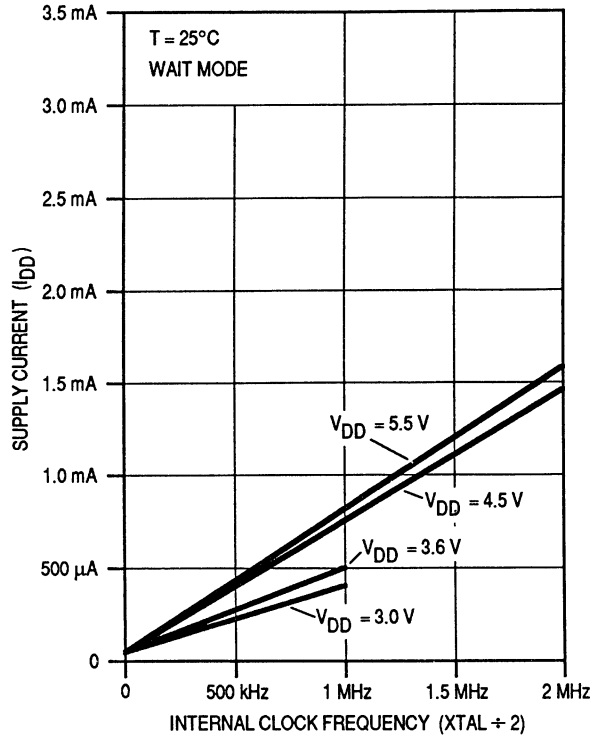
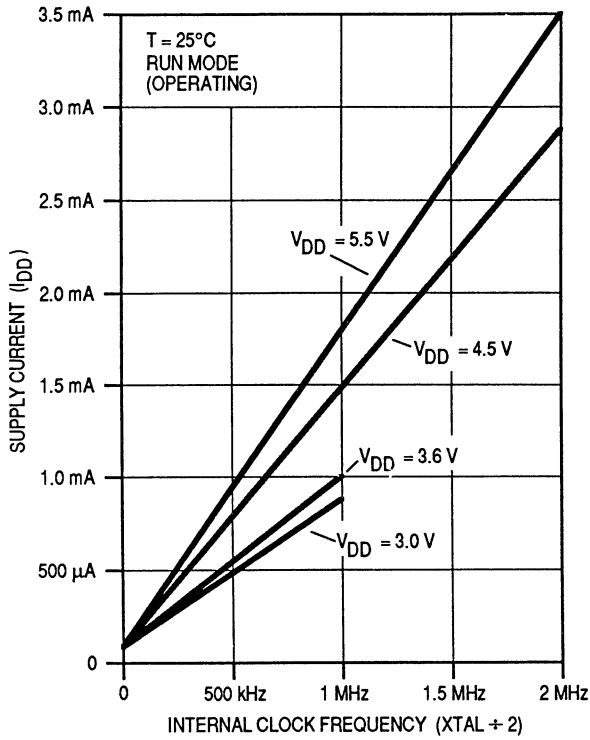


Figure 10-4. Typical Supply Current vs Clock Frequency

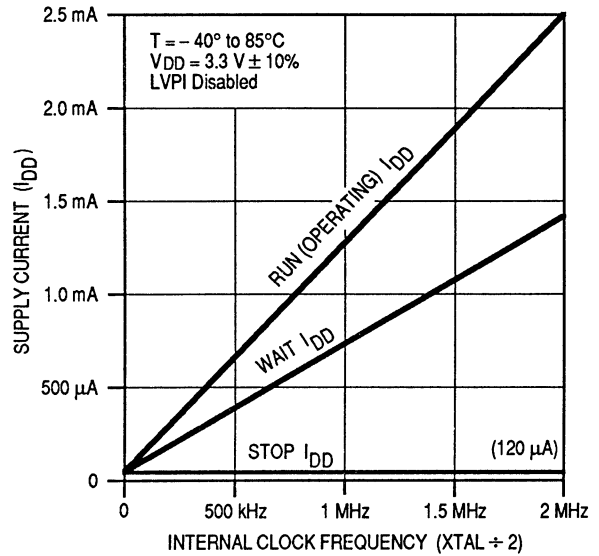
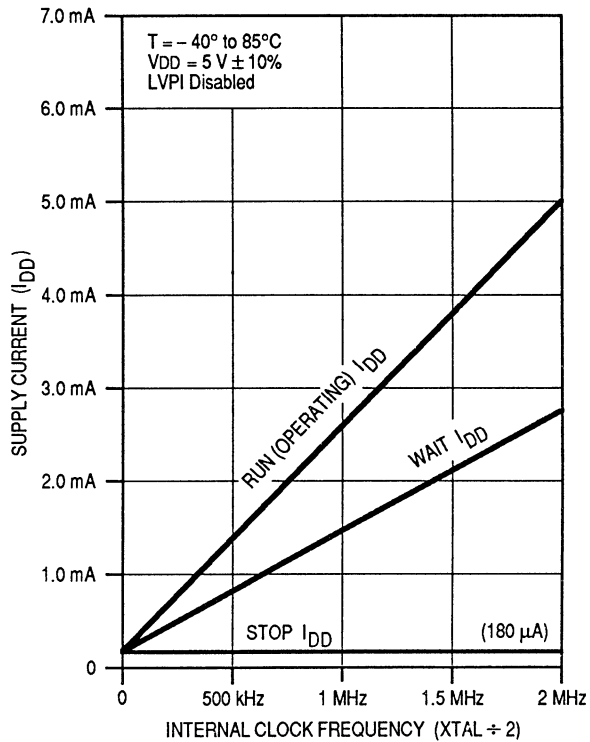


Figure 10-5. Maximum Supply Current vs Clock Frequency

10.6 A/D Converter Characteristics
Table 10-5. A/D Converter Characteristics

Characteristic	Parameter	Min	Max	Unit
Resolution	Number of bits resolved by the converter	8	8	Bit
Absolute accuracy	Difference between input voltage and full-scale equivalent of binary output code for all errors	–	$\pm 1-1/2$	LSB
Conversion range	Analog input voltage range	V _{RL}	V _{RH}	V
V _{RH}	Upper analog reference voltage	V _{RL}	V _{DD} + 0.1	V
V _{RL}	Lower analog reference voltage	V _{SS} – 0.1	V _{RH}	V
Conversion time	Total time to perform a single A/D conversion a. External clock (XTAL, EXTAL) b. Internal RC oscillator (ADRC = 1)	– –	32 1	μ s t _{cyc}
Zero input reading	Conversion result when V _{in} = V _{RL}	00	01	Hex
Full-scale reading	Conversion result when V _{in} = V _{RH}	FF	FF	Hex
Sample acquisition time (See NOTE 1.)	Analog input acquisition sampling a. External clock (XTAL, EXTAL) b. Internal RC oscillator (ADRC = 1)	t _{AD} 12 –	12 12	t _{cyc} μ s
Sample hold capacitance	Input capacitance on PD3/AN3–PD0/AN0	–	8	pF
Input leakage (See NOTE 2.)	Input leakage on PD3/AN3–PD0/AN0 Input leakage on V _{RL} and V _{RH}	– –	400 1	nA μ A

NOTES:

1. Source impedances greater than 10 k Ω will adversely affect internal RC charging time during input sampling.
2. The external system error caused by input leakage current is approximately equal to the product of R source and input current.

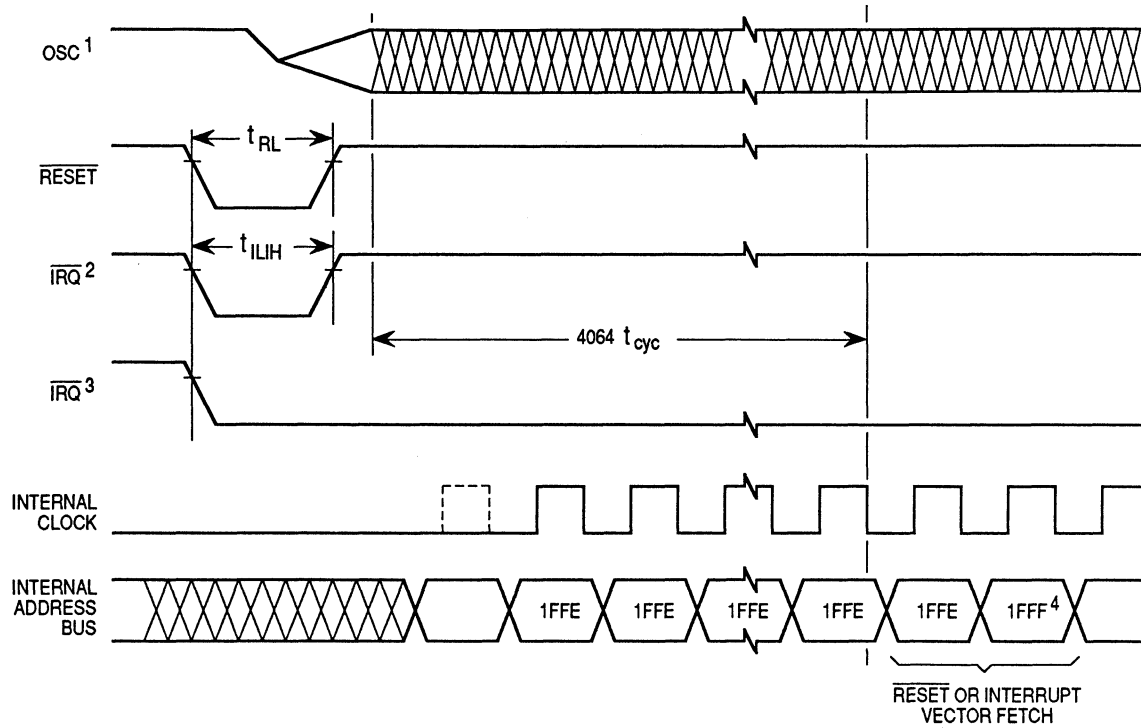
10.7 Control Timing ($V_{DD} = 5.0$ Vdc)
Table 10-6. Control Timing ($V_{DD} = 5.0$ Vdc)

 ($V_{DD} = 5.0$ Vdc \pm 10%, $V_{SS} = 0$ Vdc, $T_A = T_L$ to T_H)

Characteristic	Symbol	Min	Max	Unit
Oscillator frequency	f_{osc}			
Crystal option		–	4.2	MHz
External clock option		dc	4.2	MHz
Internal operating frequency	f_{op}			
Crystal ($f_{osc} + 2$)		–	2.1	MHz
External clock ($f_{osc} + 2$)		dc	2.1	MHz
Cycle time	t_{cyc}	480	–	ns
RESET pulse width	t_{RL}	1.5	–	t_{cyc}
Timer resolution (See NOTE 1.)	t_{RESL}	4.0	–	t_{cyc}
Interrupt pulse width low (edge-triggered)	t_{ILIH}	125	–	ns
Interrupt pulse period	t_{ILIL}	(See NOTE 2.)	–	t_{cyc}
OSC1 pulse width	t_{OH}, t_{OL}	90	–	ns
EEPROM byte programming time	t_{PGM}	–	15.0	ms
EEPROM byte erase time	t_{EBYT}	–	15.0	ms
EEPROM block erase time	t_{EBLOCK}	–	30.0	ms
EEPROM bulk erase time	t_{EBULK}	–	100.0	ms
EEPROM programming voltage fall time				μ s
Normal operation	t_{FPV}	–	10.0	
After LVPI is set	t_{FPVL}	–	10.0	
RC oscillator stabilization time	t_{RCON}	–	5	μ s
V_{DD} slew rate				V/ μ s
Rising	SVDDR	–	0.05	
Falling	SVDDF	–	0.1	
A/D on current stabilization time	t_{ADON}	–	100	μ s

NOTES:

1. Since a 2-bit prescaler in the timer must count four internal cycles (t_{cyc}), this is the limiting minimum factor in determining the timer resolution.
2. The minimum period t_{ILIL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus 19 t_{cyc} .



NOTES:

1. Represents the internal clocking of the OSC1 pin.
2. \overline{IRQ} pin edge-sensitive mask option.
3. \overline{IRQ} pin level- and edge-sensitive mask option.
4. RESET vector address shown for timing example.

Figure 10-6. STOP Recovery Timing

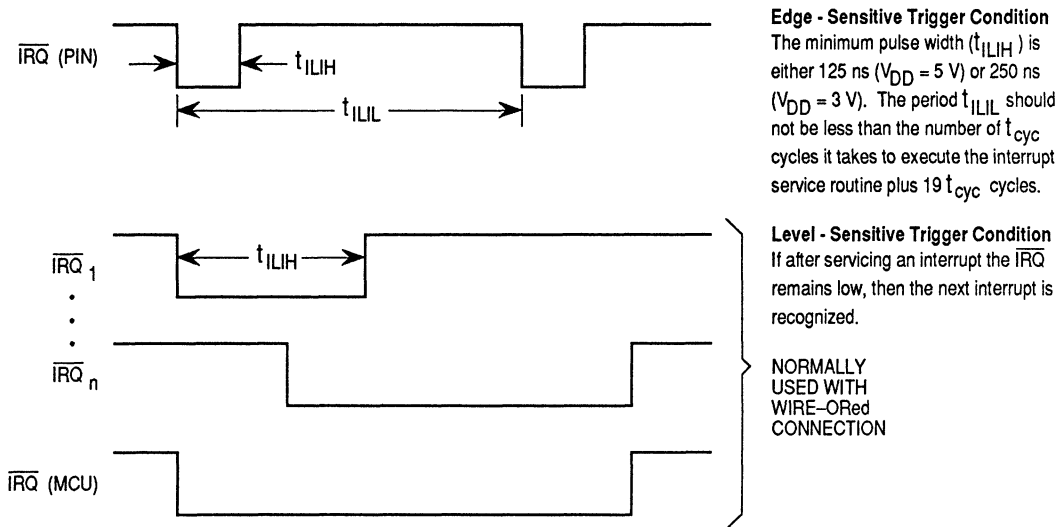


Figure 10-7. External Interrupt Timing

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

Table 10-7. Control Timing ($V_{DD} = 3.3 \text{ Vdc}$)

($V_{DD} = 3.3 \text{ Vdc} \pm 10\%$, $V_{SS} = 0 \text{ Vdc}$, $T_A = T_L \text{ to } T_H$)

Characteristic	Symbol	Min	Max	Unit
Oscillator frequency	f_{osc}			
Crystal option		–	2.0	MHz
External clock option		dc	2.0	MHz
Internal operating frequency	f_{op}			
Crystal ($f_{osc} + 2$)		–	1.0	MHz
External Clock ($f_{osc} + 2$)		dc	1.0	MHz
Cycle time	t_{cyc}	1	–	μs
RESET pulse width	t_{RL}	1.5	–	t_{cyc}
Timer resolution (See NOTE 1.)	t_{RESL}	4.0	–	t_{cyc}
Interrupt pulse width low (edge-triggered)	t_{ILIH}	250	–	ns
Interrupt pulse period	t_{ILIL}	(See NOTE 2.)	–	t_{cyc}
OSC1 pulse width	t_{OH}, t_{OL}	200	–	ns

NOTES:

1. Since a 2-bit prescaler in the timer must count four internal cycles (t_{cyc}), this is the limiting minimum factor in determining the timer resolution.
2. The minimum period t_{ILIL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus $19 t_{cyc}$.

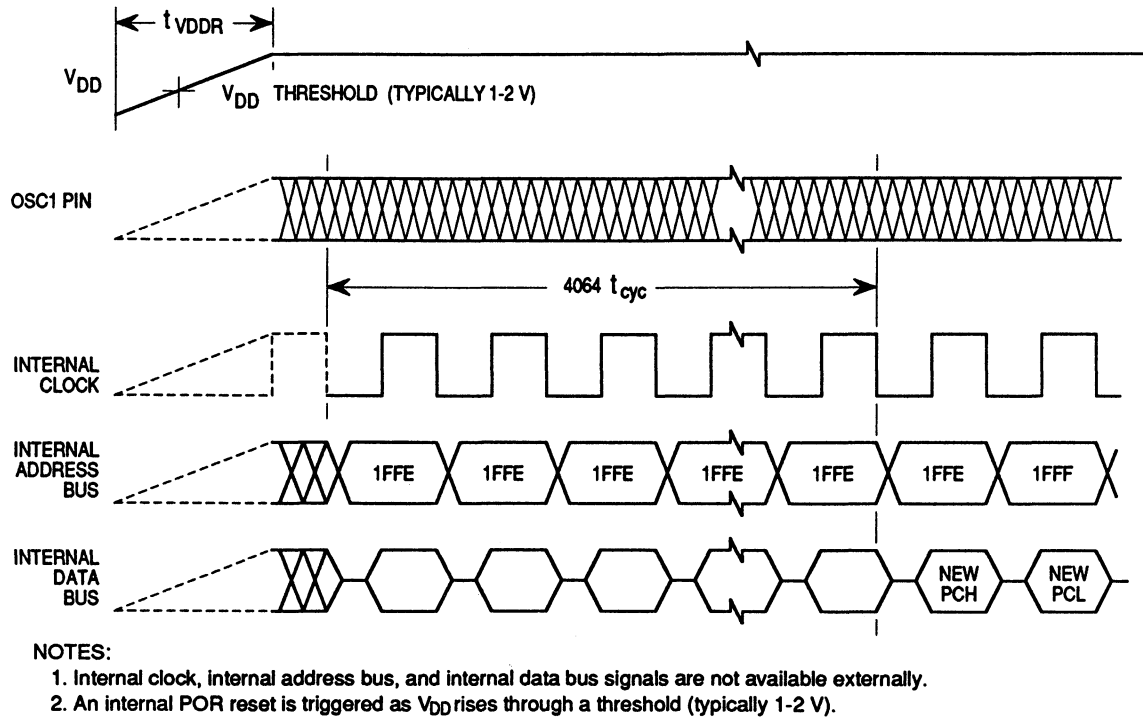


Figure 10-8. Power-On Reset Timing

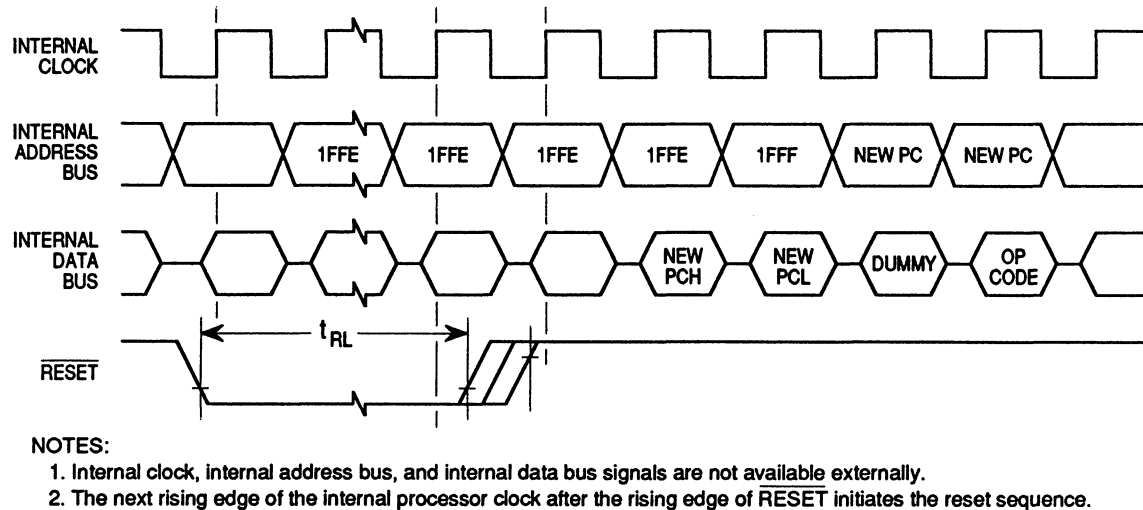


Figure 10-9. External Reset Timing

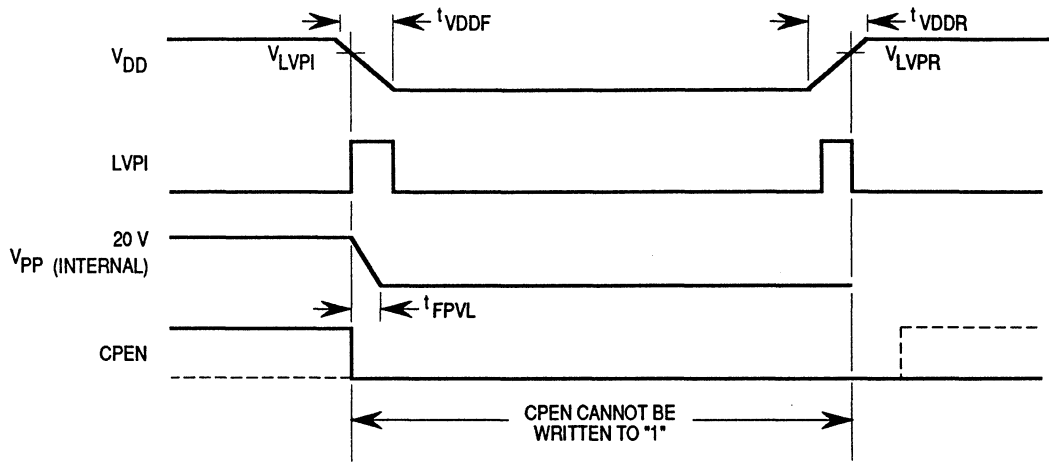


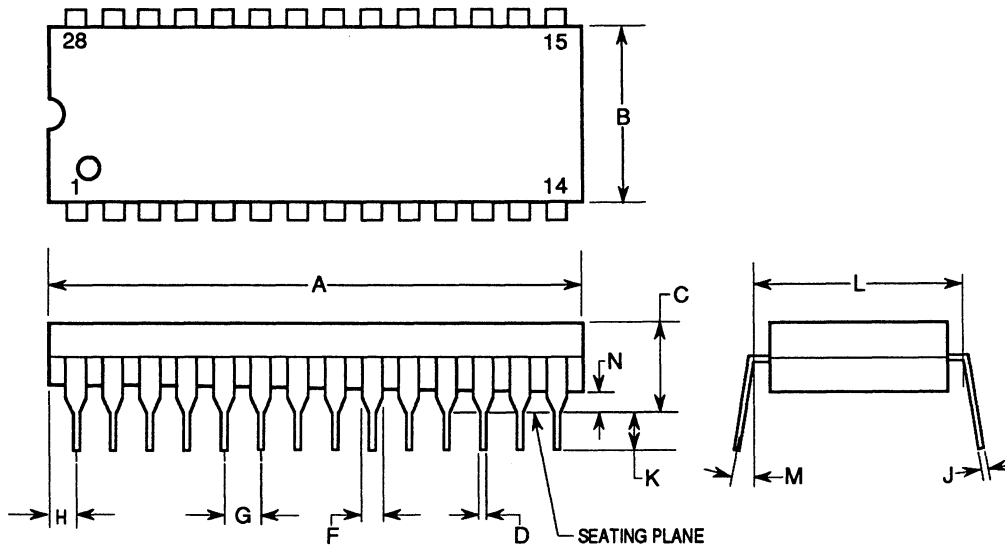
Figure 10-10. LVPI Timing

SECTION 11 MECHANICAL SPECIFICATIONS

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

11.1 DIP (P Suffix)

P SUFFIX
PLASTIC PACKAGE
CASE 710-02



NOTES:

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

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

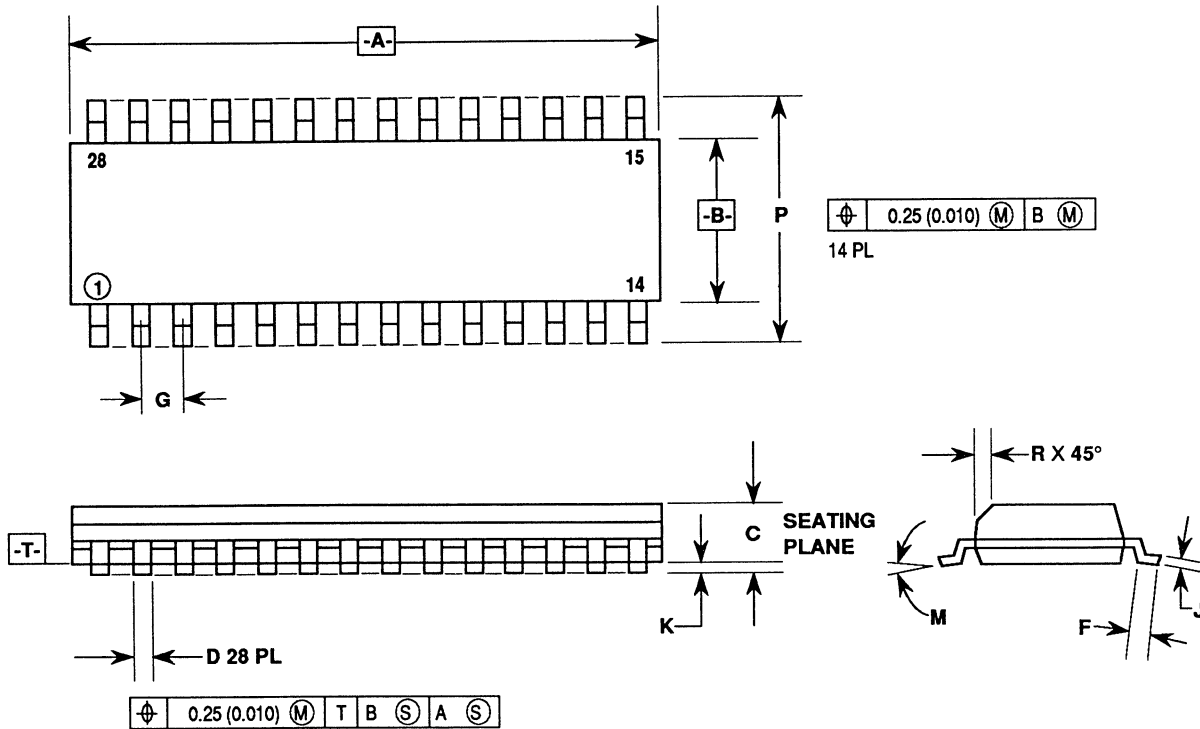
Figure 11-1. Case 710-02 Dimensions

11.2 SOIC (DW Suffix)

DW SUFFIX

SMALL OUTLINE INTEGRATED CIRCUIT

CASE 751F-02



NOTES:

1. DIMENSIONS A AND B ARE DATUMS AND T IS A DATUM SURFACE.
2. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
3. CONTROLLING DIMENSION: MILLIMETER.
4. DIMENSION A AND B DO NOT INCLUDE MOLD PROTRUSION.
5. MAXIMUM MOLD PROTRUSION 0.15 (0.006) PER SIDE.

DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	17.80	18.05	0.701	0.710
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

Figure 11-2. Case 751F-02 Dimensions

SECTION 12

ORDERING INFORMATION

This section describes the information needed to order the MCU.

12.1 ROM Pattern Media

Ordering information can be delivered to Motorola in the following media:

- MS™-DOS¹ or PC-DOS flexible disk (360K)
- EPROM(s) 2764, MCM68764, MCM68766

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, a sales person, or a Motorola representative.

12.1.1 Flexible Disks

A flexible disk containing the customer's program (using positive logic for address and data), may be submitted for pattern generation. The disk should be clearly labeled with the customer's name, data, project or product name, and the name of the file containing the pattern.

In addition to the program pattern, a file containing the program source code listing can be included. This data is kept confidential and used to expedite the process in case of any difficulty with the pattern file.

MS-DOS is the Microsoft Disk Operating System. PC-DOS is the IBM®² Personal Computer (PC) Disk Operating System. Submitted disks must be standard density (360K) double-sided 5-1/4 in. disks. The disks must contain object file code in Motorola's S-record format. The S-record format is a character-based object file format generated by M6805 cross assemblers and linkers on IBM PC-style machines.

¹MS-DOS is a trademark of Microsoft, Inc.

²IBM is a registered trademark of International Business Machines Corporation.

12.1.2 EPROMs

A type 2764, 68764, or 68766 EPROM containing the customer's program (using positive logic for address and data) may be submitted for pattern generation. User ROM is programmed at EPROM addresses \$30 through \$4F (page zero) and \$1680 through \$1E7F with vectors at addresses \$1FF0 to \$1FFF. All unused bytes, including those in the user's space, must be set to logical zero. For shipment to Motorola, EPROMs should be packed securely in a conductive IC carrier. Styrofoam packaging is not acceptable for shipment.

12.2 ROM Pattern Verification

12.2.1 Verification Media

All original pattern media are filed for contractual purposes and are not returned. A computer listing of the ROM code is generated and returned along with a listing verification form. The listing should be thoroughly checked, and the verification form should be completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for the creation of the customer mask. To aid in the verification process, Motorola programs the *customer-supplied* blank EPROMs or DOS disks from the data file used to create the custom mask.

12.2.2 ROM Verification Units (RVUs)

Ten RVUs containing the customer's ROM pattern are sent for program verification. These units are made using the custom mask, but are for the purpose of ROM verification only. For expediency, the RVUs are unmarked, packaged in ceramic, and tested with 5 V at room temperature. These RVUs are free of charge with the minimum order quantity, but are not production parts. RVUs are not backed or guaranteed by Motorola Quality Assurance.

12.3 MC Order Numbers

Table 12-1 provides ordering information for available package types.

Table 12-1. MC Order Numbers

Package Type	MC Order Number
Plastic (P suffix)	MC68HC05P8P
SOIC (DW suffix)	MC68HC05P8DW



Freescale Semiconductor, Inc.

MC68HC05P8 MCU ORDERING FORM

Date _____ Customer PO Number _____

Customer Company _____

Address _____

City _____ State _____ Zip _____

Country _____

Phone _____ Extension _____

Customer Contact Person _____

Customer Part Number (If Applicable – 12 Characters Maximum) _____

Application _____

COP Watchdog Timer:

- Enable
- Disable

STOP Instruction:

- Enable
- Disable

External Interrupt Trigger:

- Edge-Sensitive
- Edge- and Level-Sensitive

Low-Voltage Programming Inhibit Circuit:

- Enable
- Disable

Temperature Range:

- 0°C to 70°C (Standard)
- 40°C to +85°C (Extended)
- 40°C to +125°C (Automotive)

Special Electrical Provisions: _____
(Customer specifications required)

Pattern Media:

- MS-DOS Disk File
- 2764 EPROM
- MCM68764 EPROM
- PC-DOS Disk File
- MCM68766 EPROM
- Other _____

(Requires prior factory approval)

Device Marking:

- Motorola Standard
Motorola Logo
Motorola Part Number
Mask and Datecode
- Standard with Customer Part Number
Motorola Logo
Motorola Part Number
Customer Part Number
Mask and Datecode
- Other _____

Device marking other than the two standard forms requires prior factory approval.

(SIGNATURE)

Device to be tested to Motorola data sheet specifications. Customer part number, if used as part of marking, is for reference purposes only.

(SIGNATURE)

Device to be tested to customer specifications. (Customer specifications required)

ONLY ONE SIGNATURE IS REQUIRED TO PROCESS THIS ORDERING FORM.



Home Page:

www.freescale.com

email:

support@freescale.com

USA/Europe or Locations Not Listed:

Freescale Semiconductor
 Technical Information Center, CH370
 1300 N. Alma School Road
 Chandler, Arizona 85224
 (800) 521-6274
 480-768-2130

support@freescale.com

Europe, Middle East, and Africa:

Freescale Halbleiter Deutschland GmbH
 Technical Information Center
 Schatzbogen 7
 81829 Muenchen, Germany
 +44 1296 380 456 (English)
 +46 8 52200080 (English)
 +49 89 92103 559 (German)
 +33 1 69 35 48 48 (French)
support@freescale.com

Japan:

Freescale Semiconductor Japan Ltd.
 Headquarters
 ARCO Tower 15F
 1-8-1, Shimo-Meguro, Meguro-ku
 Tokyo 153-0064, Japan
 0120 191014
 +81 2666 8080
support.japan@freescale.com

Asia/Pacific:

Freescale Semiconductor Hong Kong Ltd.
 Technical Information Center
 2 Dai King Street
 Tai Po Industrial Estate,
 Tai Po, N.T., Hong Kong
 +800 2666 8080
support.asia@freescale.com

For Literature Requests Only:

Freescale Semiconductor
 Literature Distribution Center
 P.O. Box 5405
 Denver, Colorado 80217
 (800) 441-2447
 303-675-2140
 Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

RoHS-compliant and/or Pb- free versions of Freescale products have the functionality and electrical characteristics of their non-RoHS-compliant and/or non-Pb- free counterparts. For further information, see <http://www.freescale.com> or contact your Freescale sales representative.

For information on Freescale's Environmental Products program, go to <http://www.freescale.com/epp>.

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document. Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.



**Home Page:**

www.freescale.com

email:

support@freescale.com

USA/Europe or Locations Not Listed:

Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
(800) 521-6274
480-768-2130

support@freescale.com

Europe, Middle East, and Africa:

Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:

Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku
Tokyo 153-0064, Japan
0120 191014
+81 2666 8080
support.japan@freescale.com

Asia/Pacific:

Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate,
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:

Freescale Semiconductor
Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
(800) 441-2447
303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document. Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.