HC05

MC68HC05F5

TECHNICAL
DATA

MC68HC05F5

HCMOS MICROCONTROLLER UNIT

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

Paragraph **Title** **Page**

SECTION 3
CENTRAL PROCESSOR UNIT

SECTION 4
INTERRUPTS

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Concluded)

LIST OF FIGURES

LIST OF FIGURES
(Concluded)

LIST OF TABLES

SECTION 1
GENERAL DESCRIPTION

The MC68HC05F5 is a member of the low-cost, high-performance M68HC05 Family of 8-bit microcontroller units (MCUs). The M68HC05 Family is based on the customer-specified integrated circuit (CSIC) design strategy. All MCUs in the family use the popular M68HC05 central processor unit (CPU) and are available with a variety of subsystems, memory sizes and types, and package types.

On-chip memory of the MC68HC05F5 includes 5632 bytes of user ROM and 224 bytes of user RAM.

1.1 Features

Features of the MC68HC05F5 MCU include the following:

- Popular M68HC05 Central Processor Unit
- Memory-Mapped Input/Output (I/O) Registers
- 5632 Bytes of User ROM Including 8 Locations for User Vectors
- 224 Bytes of User RAM
- 30 Bidirectional I/O Pins
- Two Independently Maskable External Interrupt Pins ($\overline{IRQ}$ and IRQ2)
- Mask-Optional $\overline{IRQ}$ Pin Sensitivity (Edge-Triggered or Edge- and Level-Triggered)
- Fully Static Operation with no Minimum Clock Speed
- On-Chip Oscillator with Mask-Optional Connections for Crystal/Resonator or Resistor-Capacitor (RC) Network
- Mask-Optional Computer Operating Properly (COP) Watchdog
- 15-Bit Multifunction Timer with Real-Time Interrupt Circuit
- Dual-Tone Multi-Frequency (DTMF) Receiver
- Power-Saving Stop, Wait, and Data-Retention Modes
- Self-Check ROM
- Mask-Optional STOP Instruction
- 8 × 8 Unsigned Multiply Instruction
- 40-Pin Plastic Dual In-Line Package (PDIP)
- 44-Pin Plastic Leaded Chip Carrier (PLCC)

1.2 Mask Options

The following MC68HC05F5 mask options are available:

- Enabled or disabled COP watchdog
- Edge-triggered or edge- and level-triggered external $\overline{IRQ}$ pin
- On-chip oscillator connections for crystal/resonator or for resistor-capacitor (RC) circuit
- Enabled or disabled STOP instruction

1.3 MCU Structure

Figure 1-1 shows the structure of the MC68HC05F5 MCU.

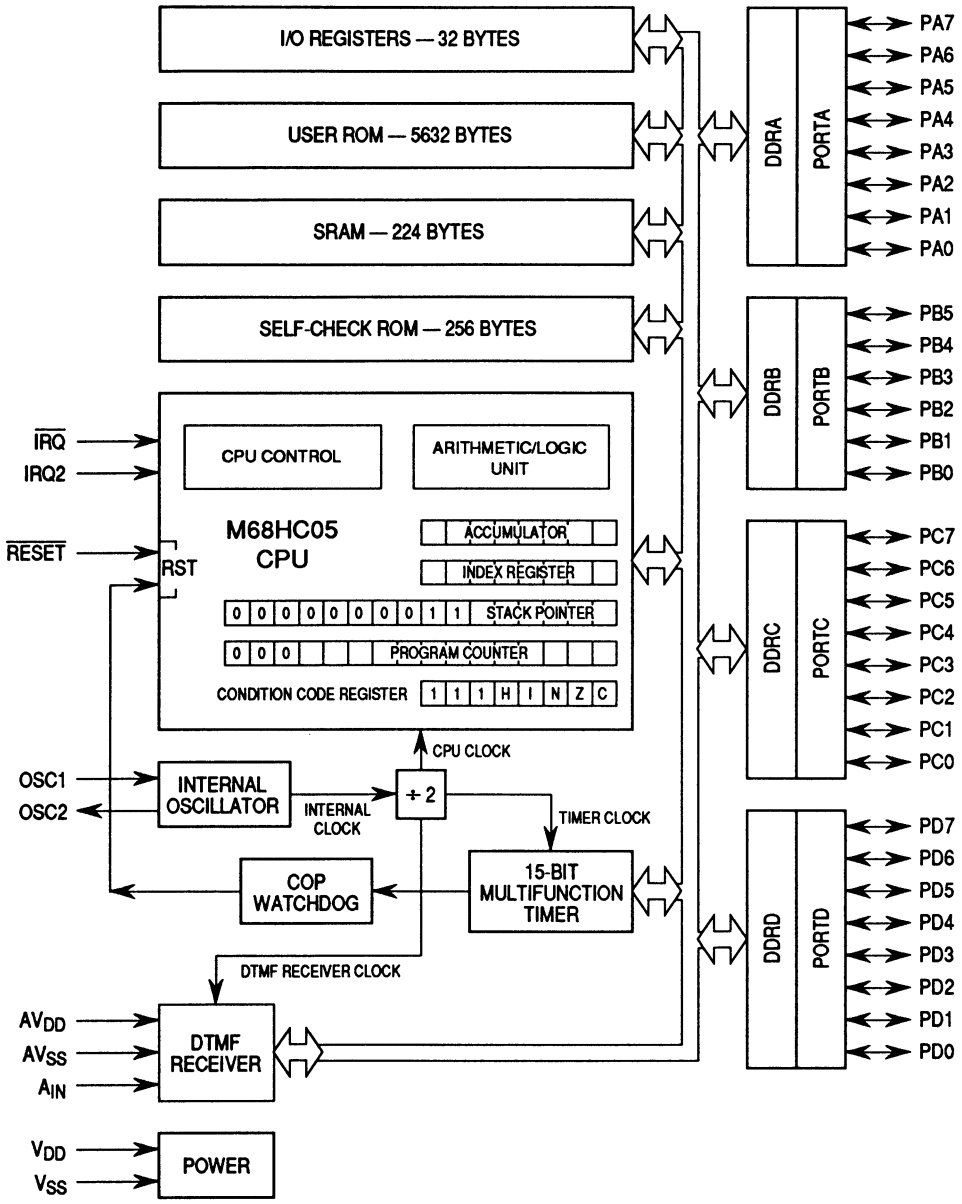


Figure 1-1. MC68HC05F5 Block Diagram

1.4 Pin Assignments

Figure 1-2 shows the PDIP and PLCC pin assignments.

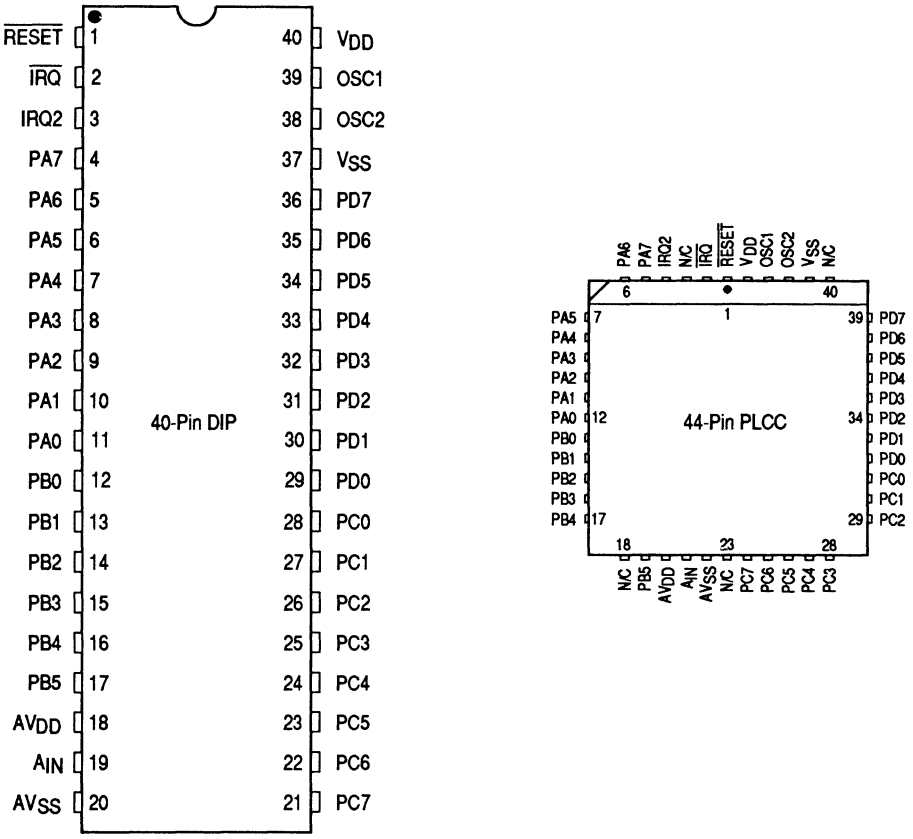


Figure 1-2. Pin Assignments

1.4.1 V_{DD} and V_{SS}

V_{DD} and V_{SS} are the power supply and ground pins. The MCU operates from a single 5 V power supply.

Very fast signal transitions occur on the MCU pins, placing very high short-duration current demands on the power supply. To prevent noise problems, take special care to provide good power supply bypassing at the MCU. Place the C1 bypass capacitor as close to the MCU as possible, as Figure 1-3 shows. C2 is an optional bulk current bypass capacitor for use in applications that require the port pins to source high current levels.

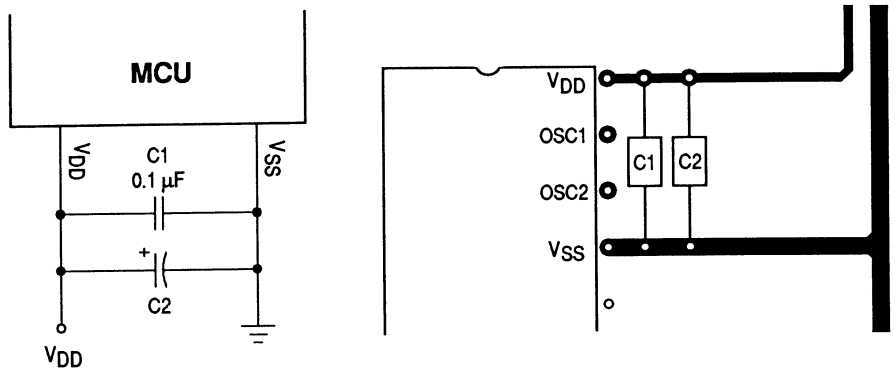


Figure 1-3. Bypassing Layout Recommendation

1.4.2 OSC1 and OSC2

The OSC1 and OSC2 pins are the control connections for the on-chip oscillator. Depending on the mask option selected, the oscillator can be driven by any of the following:

- Crystal (crystal/ceramic resonator mask option)
- Ceramic resonator (crystal/ceramic resonator mask option)
- Resistor-capacitor network (RC oscillator mask option)
- External clock signal (RC oscillator mask option)

The MCU divides the frequency of the oscillator, f_{OSC}, or external clock source by two to produce the internal operating frequency, f_{OP}.

1.4.2.1 Crystal

The crystal/ceramic resonator mask option allows a crystal connected to the OSC1 and OSC2 pins to drive the on-chip oscillator. Figure 1-4 shows a typical crystal oscillator circuit for an AT-cut parallel resonant crystal. Follow the crystal supplier's recommendations, as the crystal parameters determine the external component values required to provide reliable start-up and maximum stability. The load capacitance values used in the oscillator circuit design should include all stray layout capacitances. To minimize output distortion, mount the crystal and capacitors as close as possible to the pins.

NOTE

Use an AT-cut crystal and not an AT-strip crystal. The MCU may overdrive an AT-strip crystal.

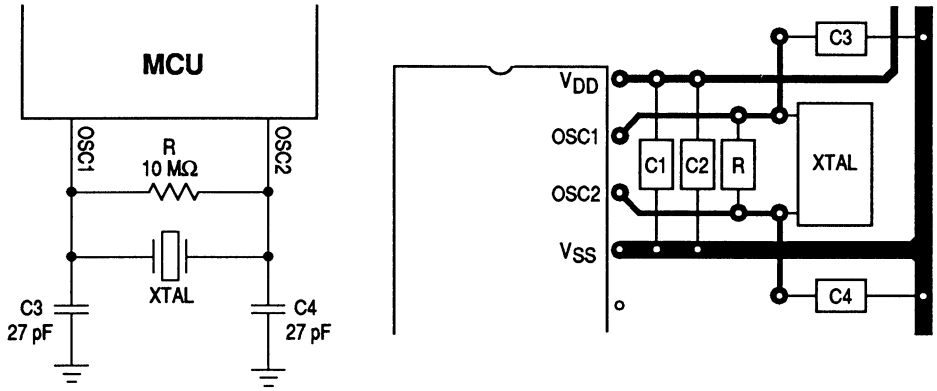


Figure 1-4. Crystal Connections

1.4.2.2 Ceramic Resonator

To reduce cost, use a ceramic resonator in place of the crystal. Figure 1-5 shows a ceramic resonator circuit. Follow the resonator manufacturer's recommendations for the values of any external components required. Loading capacitance values used in the oscillator circuit design should include all stray layout capacitances. To minimize output distortion, mount the resonator as close as possible to the pins.

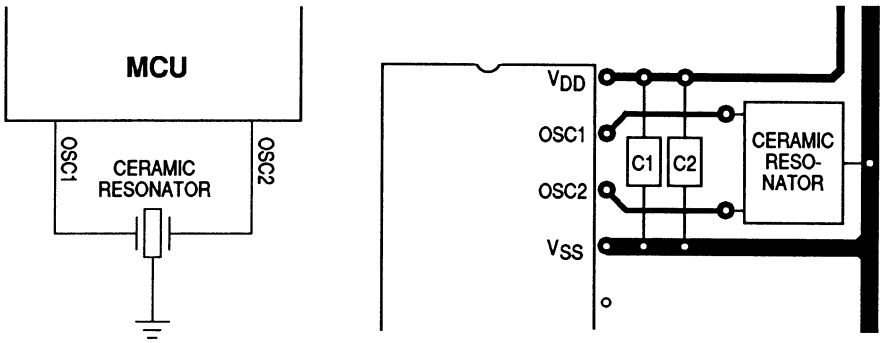


Figure 1-5. Ceramic Resonator Connections

NOTE

Because the frequency stability of ceramic resonators is not as high as that of crystal oscillators, using a ceramic resonator may degrade the performance of the DTMF receiver.

1.4.2.3 RC Oscillator

For maximum cost reduction, the RC oscillator mask option allows the configuration shown in Figure 1-6 to drive the on-chip oscillator. The relationship between f_{OSC} and the external components is $f_{OSC} \approx 1 \div 2.28RC$. The OSC2 signal is a square wave, and the signal on OSC1 is a triangular wave. The optimum frequency for the RC oscillator configuration is 2 MHz. Mount the RC components as close as possible to the pins for start-up stabilization and to minimize output distortion.

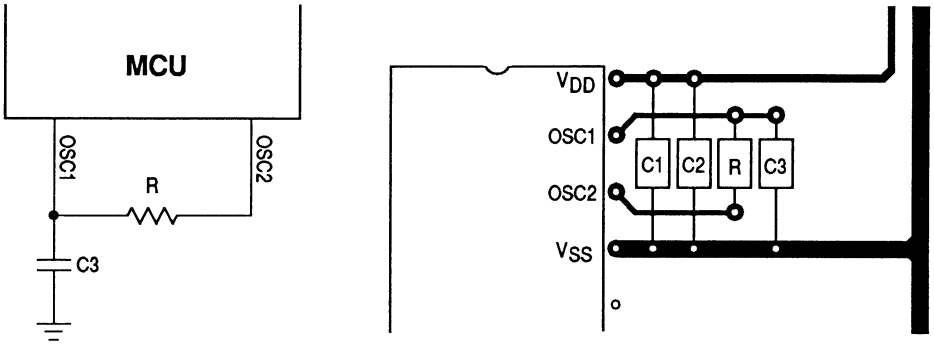


Figure 1-6. RC Oscillator Connections

1.4.2.4 External Clock

The RC oscillator mask option allows an external clock from a CMOS-compatible device to drive the OSC1 input with OSC2 unconnected, as Figure 1-7 shows.

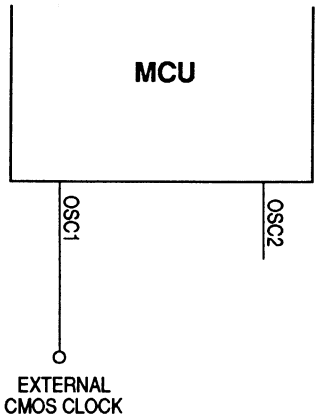


Figure 1-7. External Clock Connections

1.4.3 $\overline{\text{RESET}}$

A logic zero on the $\overline{\text{RESET}}$ pin forces the MCU to a known start-up state. (See **5.1.2 External Reset** for more information.)

1.4.4 $\overline{\text{IRQ}}$ (External Interrupt)

The $\overline{\text{IRQ}}$ pin has the following functions:

- Applying asynchronous external interrupt signals (See **4.1.2 External Interrupts**.)
- Applying V_{TST} to put the MCU in self-check mode (See **SECTION 10 SELF-CHECK ROM**.)

1.4.5 IRQ2 (External Interrupt 2)

IRQ2 is a second maskable external interrupt pin that is edge-sensitive only. Polarity of the triggering edge is programmable in the IRQ status and control register. (See **4.1.2.2 IRQ2 Pin**.)

1.4.6 PA7–PA0

PA7–PA0 are general-purpose bidirectional I/O port pins. (See **7.2 Port A**.)

1.4.7 PB5–PB0

PB5–PB0 are general-purpose bidirectional I/O port pins. (See **7.3 Port B**.)

1.4.8 PC7–PC0

PC7–PC0 are general-purpose bidirectional I/O port pins. (See **7.4 Port C**.)

1.4.9 PD7–PD0

PD7–PD0 are general-purpose bidirectional I/O port pins. (See **7.5 Port D**.)

1.4.10 AV_{DD} and AV_{SS}

AV_{DD} and AV_{SS} are the power supply and ground pins for the analog section of the DTMF receiver. The proximity of the AV_{DD} and AV_{SS} pins allows for effective supply decoupling. Figure 1-8 shows the recommended decoupling configuration.

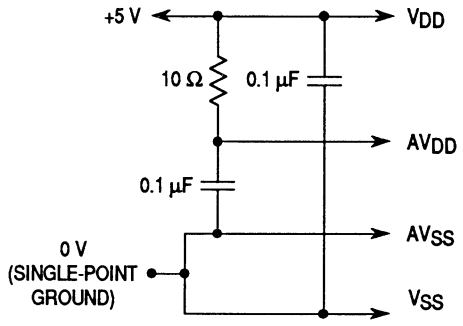


Figure 1-8. Power Supply Decoupling

1.4.11 A_{IN} (Analog Input)

A_{IN} is the analog input pin of the DTMF receiver.

SECTION 2
MEMORY

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

2.1 Memory Map

The CPU can address 8 Kbytes of memory space. The program counter typically advances one address at a time through the memory, reading the program instructions and data. The ROM portion of memory holds the program instructions, fixed data, user-defined vectors, and interrupt service routines. The RAM portion of memory holds variable data. I/O registers are memory-mapped so that the CPU can access their locations in the same way that it accesses all other memory locations.

Figure 2-1 is a memory map of the MCU. Figure 2-2 is a more detailed memory map of the 32-byte I/O register section.

2.2 Input/Output Section

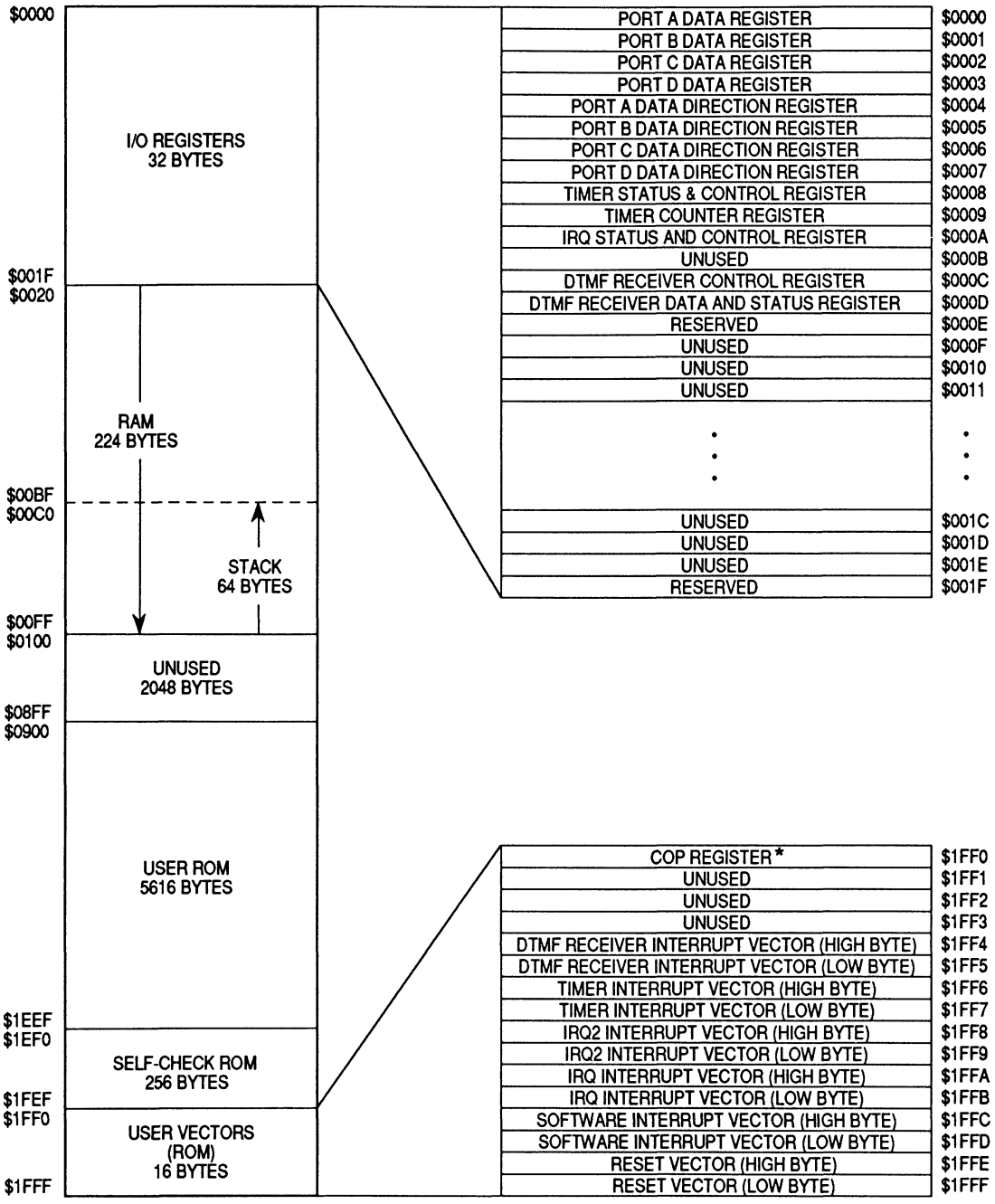
The first 32 addresses of the memory space, $0000–$001F, are the I/O section. These are the addresses of the I/O control registers, status registers, and data registers.

2.3 RAM

The 224 addresses from $0020–$00FF are RAM locations. The CPU uses the top 64 RAM addresses, $00C0–$00FF, as the stack. Before processing an interrupt, the CPU uses five bytes of the stack to save the contents of the CPU registers. During a subroutine call, the CPU uses two bytes of the stack to store the return address. The stack pointer decrements during pushes and increments during pulls.

NOTE

Be careful when using nested subroutines or multiple interrupt levels. The CPU may overwrite data in the RAM during a subroutine or during the interrupt stacking operation.



* Writing zero to bit 0 of $1FF0 clears the COP watchdog. Reading $1FF0 returns ROM data.

Figure 2-1. Memory Map

	Bit 7	6	5	4	3	2	1	Bit 0	
$0000	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0	PORTA
$0001	0	0	PB5	PB4	PB3	PB2	PB1	PB0	PORTB
$0002	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0	PORTC
$0003	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0	PORTD
$0004	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0	DDRA
$0005	0	0	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0	DDRB
$0006	DDRC7	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0	DDRC
$0007	DDRD7	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0	DDRD
$0008	TOF	RTIF	TOIE	RTIE	0	0	RT1	RT0	TSCR
$0009	Bit 7	6	5	4	3	2	1	Bit 0	TCNTR
$000A	IRQM	IRQ2M	EDGE	PIN	REQ	REQ2	ACK	ACK2	ISCR
$000B	—	—	—	—	—	—	—	—	UNUSED
$000C	DRIE	0	0	0	0	CLK	G1	G0	DCR
$000D	DIRQ	ETD	PAUSE	DV	D8	D4	D2	D1	DDSR
$000E	—	—	—	—	—	—	—	—	RESERVED
$000F	—	—	—	—	—	—	—	—	UNUSED
$0010	—	—	—	—	—	—	—	—	UNUSED
$0011	—	—	—	—	—	—	—	—	UNUSED
•									
•									
•									
$001C	—	—	—	—	—	—	—	—	UNUSED
$001D	—	—	—	—	—	—	—	—	UNUSED
$001E	—	—	—	—	—	—	—	—	UNUSED
$001F	—	—	—	—	—	—	—	—	RESERVED
$1FF0	—	—	—	—	—	—	—	COPC	COPR

Figure 2-2. I/O Registers

2.4 ROM

The memory map includes the following three ROM sections:

- $0900–$1EEF — 5616 bytes of user ROM
- $1EF0–$1FEF — 256 bytes reserved for self-check ROM
- $1FF0–$1FFF — 16 bytes reserved for interrupt and reset vectors.

SECTION 3
CENTRAL PROCESSOR UNIT

This section describes the CPU registers.

3.1 CPU Registers

Figure 3-1 shows the five CPU registers. CPU registers are not part of the memory map.

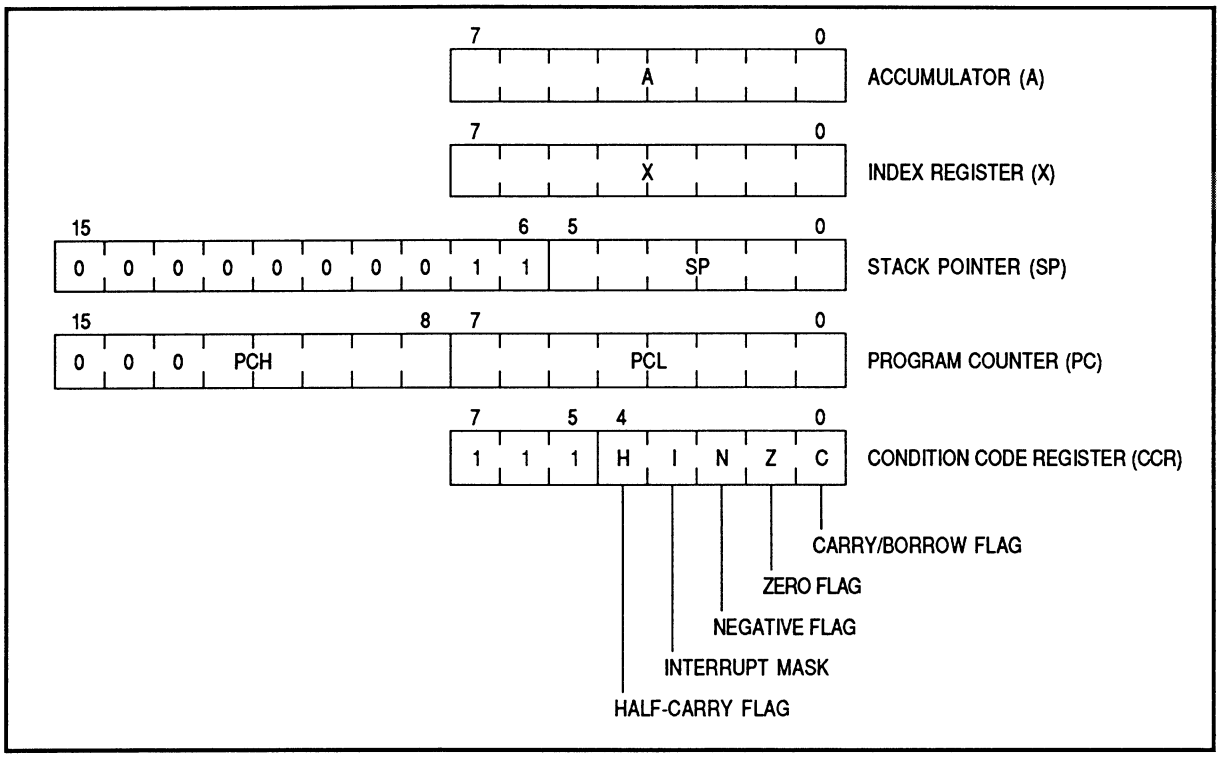


Figure 3-1. Programming Model

3.1.1 Accumulator

The accumulator, as shown in Figure 3-2, is a general-purpose 8-bit register. The CPU uses the accumulator to hold operands and the results of arithmetic and nonarithmetic operations.

	Bit 7	6	5	4	3	2	1	Bit 0
RESET:	UNAFFECTED BY RESET							

Figure 3-2. Accumulator

3.1.2 Index Register

In the indexed addressing modes, the CPU uses the byte in the index register (Figure 3-3) to determine the conditional address of the operand. (See **11.1.5 Indexed, No Offset**; **11.1.6 Indexed, 8-Bit Offset**; and **11.1.7 Indexed, 16-Bit Offset**.)

	Bit 7	6	5	4	3	2	1	Bit 0
RESET:	UNAFFECTED BY RESET							

Figure 3-3. Index Register

The 8-bit index register can also serve as a temporary data storage location.

3.1.3 Stack Pointer

The stack pointer shown in Figure 3-4 is a 16-bit register that contains the address of the next location on the stack. During a reset or after the reset stack pointer (RSP) instruction, the stack pointer is preset to $00FF. The address in the stack pointer decrements as data is pushed onto the stack and increments as data is pulled from the stack.

	Bit 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	Bit 0
	0	0	0	0	0	0	0	0	1	1						
RESET:	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Figure 3-4. Stack Pointer

The ten most significant bits of the stack pointer are permanently fixed at 0000000011, so the stack pointer produces addresses from $00C0 to $00FF. If subroutines and interrupts use more than 64 stack locations, the stack pointer wraps around to address $00FF and begins writing over the previously stored data. A subroutine uses two stack locations; an interrupt uses five locations.

3.1.4 Program Counter

The program counter shown in Figure 3-5 is a 16-bit register that contains the address of the next instruction or operand to be fetched. The three most significant bits of the program counter are ignored internally and appear as 000.

Normally, the address in the program counter automatically 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.

	Bit15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	Bit 0
	0	0	0													
RESET:	0	0	0	LOADED WITH VECTOR FROM $1FFE AND $1FFF												

Figure 3-5. Program Counter

3.1.5 Condition Code Register

The condition code register shown in Figure 3-6 is an 8-bit register whose three most significant bits are permanently fixed at 111. The condition code register contains the interrupt mask and four flags that indicate the results of the instruction just executed. The following paragraphs describe the functions of the condition code register.

	Bit 7	6	5	4	3	2	1	Bit 0
	1	1	1	H	I	N	Z	C
RESET:	1	1	1	U	1	U	U	U

U = UNAFFECTED

Figure 3-6. Condition Code Register

3.1.5.1 Half-Carry Flag

The CPU sets the half-carry flag when a carry occurs between bits 3 and 4 of the accumulator during an ADD or ADC operation. The half-carry flag is required for binary-coded decimal (BCD) arithmetic operations.

3.1.5.2 Interrupt Mask

Setting the interrupt mask disables interrupts. If an interrupt request occurs while the interrupt mask is logic zero, the CPU saves the CPU registers on the stack, sets the interrupt mask, and then fetches the interrupt vector. If an interrupt request occurs while the interrupt mask is set, the interrupt request is latched. Normally, the CPU processes the latched interrupt as soon as the interrupt mask is cleared again.

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 be cleared only by a software instruction.

3.1.5.3 Negative Flag

The CPU sets the negative flag when an arithmetic operation, logical operation, or data manipulation produces a negative result. Bit 7 of the negative result is automatically set, so the negative flag can be used to check an often-tested bit by assigning it 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 flag according to the state of the tested bit.

3.1.5.4 Zero Flag

The CPU sets the zero flag when an arithmetic operation, logical operation, or data manipulation produces a result of $00.

3.1.5.5 Carry/Borrow Flag

The CPU sets the carry/borrow flag when an addition operation produces a carry out of bit 7 of the accumulator or when a subtraction operation requires a borrow. Some logical operations and data manipulation instructions also clear or set the carry/borrow flag.

3.2 Arithmetic/Logic Unit (ALU)

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

The binary arithmetic circuits decode instructions and set up the ALU for the selected operation. Most binary arithmetic is based on the addition algorithm, carrying out subtraction as negative addition. Multiplication is not performed as a discrete operation 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.

SECTION 4
INTERRUPTS

This section describes how interrupts temporarily change the normal processing sequence.

4.1 Interrupt Sources

The following sources can generate interrupts:

- SWI instruction
- $\overline{\text{IRQ}}$ pin
- IRQ2 pin
- Timer
- Dual-tone multiple-frequency receiver

An interrupt temporarily stops normal program execution to process a particular event. An interrupt does not stop the operation of the instruction being executed, but takes effect when the current instruction completes its execution. Interrupt processing automatically saves the CPU registers on the stack and loads the program counter with a user-defined interrupt vector address.

4.1.1 Software Interrupt

The software interrupt (SWI) instruction causes a nonmaskable interrupt.

4.1.2 External Interrupts

The following sources can generate external interrupts:

- $\overline{\text{IRQ}}$ pin
- IRQ2 pin

Setting the I bit in the condition code register disables both the IRQ pin and the IRQ2 pin as external interrupt sources.

4.1.2.1 $\overline{\text{IRQ}}$ Pin

An interrupt signal on the $\overline{\text{IRQ}}$ pin latches an external interrupt request. When the CPU completes its current instruction, it tests the IRQ latch. If the IRQ latch is set, the CPU then tests the I bit in the condition code register and the IRQM bit in the IRQ status and control register. If both the I bit and the IRQM bit are clear, the CPU then begins the interrupt sequence. The CPU clears the IRQ latch during interrupt processing, so that another interrupt signal on the IRQ pin can latch another interrupt request during the interrupt service routine. Software can also clear the IRQ latch by setting the ACK bit in the IRQ status and control register. As soon as the I bit is cleared during the return from interrupt, the CPU can recognize the new interrupt request. Figure 4-1 shows the $\overline{\text{IRQ}}$ pin interrupt logic.

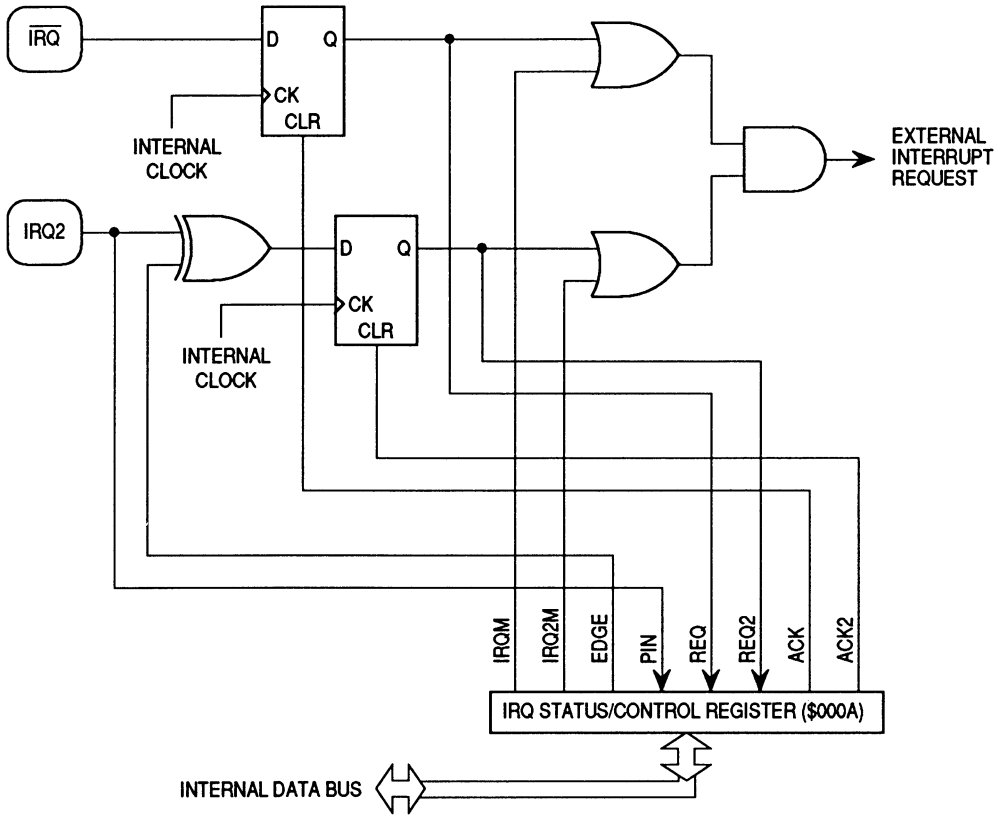


Figure 4-1. External Interrupt Function

Interrupt triggering sensitivity of the $\overline{\text{IRQ}}$ pin is a mask option. The $\overline{\text{IRQ}}$ pin can be negative edge-triggered or negative edge- and low level-triggered. The mask option for level-sensitive triggering allows multiple external interrupt sources to be wired-OR to the $\overline{\text{IRQ}}$ pin. An external interrupt request is latched as long as any source is holding the $\overline{\text{IRQ}}$ pin low.

4.1.2.2 IRQ2 Pin

An interrupt signal on the IRQ2 pin latches an external interrupt request. When the CPU completes its current instruction, it tests the IRQ2 latch. If the IRQ2 latch is set, the CPU then tests the I bit in the condition code register and the IRQ2M bit in the IRQ status and control register. If both the I bit and the IRQ2M bit are clear, the CPU then begins the interrupt sequence. The CPU clears the IRQ2 latch during interrupt processing, so that another interrupt signal on the IRQ2 pin can latch another interrupt request during the interrupt service routine. Software can also clear the IRQ2 latch by setting the ACK2 bit in the IRQ status and control register. As soon as the I bit is cleared during the return from interrupt, the CPU can recognize the new interrupt request.

The EDGE bit in the IRQ status and control register controls the interrupt triggering sensitivity of the IRQ2 pin. The IRQ2 pin can be negative edge-triggered or positive edge-triggered. Figure 4-1 shows the IRQ2 pin interrupt logic.

4.1.2.3 IRQ Status and Control Register (ISCR)

The IRQ status and control register, shown in Figure 4-2, controls and monitors external interrupt requests.

ISCR — IRQ Status and Control Register **$000A**

	Bit 7	6	5	4	3	2	1	Bit 0
	IRQM	IRQ2M	EDGE	PIN	REQ	REQ2	ACK	ACK2
RESET:	0	0	0	—	0	0	0	0

Figure 4-2. IRQ Status and Control Register (ISCR)

IRQM — $\overline{\text{IRQ}}$ Pin Interrupt Mask

This read/write bit disables $\overline{\text{IRQ}}$ pin interrupts.

1 = $\overline{\text{IRQ}}$ pin interrupts disabled
0 = $\overline{\text{IRQ}}$ pin interrupts enabled

IRQ2M — IRQ2 Pin Interrupt Mask

This read/write bit disables IRQ2 pin interrupts.

1 = IRQ2 pin interrupts disabled
0 = IRQ2 pin interrupts enabled

EDGE — IRQ2 Pin Active Edge Select

This read/write bit determines whether a positive edge or negative edge on the IRQ2 pin triggers an interrupt.

1 = IRQ2 pin interrupts negative edge triggered
0 = IRQ2 pin interrupts positive edge triggered

PIN — IRQ2 Pin State

This read-only bit reflects the logic state of the IRQ2 pin.

1 = IRQ2 pin at logic one
0 = IRQ2 pin at logic zero

REQ — $\overline{\text{IRQ}}$ Pin Interrupt Request

This read-only bit reflects the state of the $\overline{\text{IRQ}}$ pin interrupt request latch. At the beginning of $\overline{\text{IRQ}}$ pin interrupt processing, the CPU clears the $\overline{\text{IRQ}}$ pin interrupt request latch. Therefore, a second $\overline{\text{IRQ}}$ pin interrupt request can be latched while the first one is being processed. The CPU recognizes the second request as soon as the return from interrupt clears the I bit.

1 = $\overline{\text{IRQ}}$ pin interrupt request pending
0 = No $\overline{\text{IRQ}}$ pin interrupt request pending

REQ2 — IRQ2 Pin Interrupt Request

This read-only bit reflects the state of the IRQ2 pin interrupt request latch. At the beginning of IRQ2 pin interrupt processing, the CPU clears the IRQ2 pin interrupt request latch. Therefore, a second IRQ2 pin interrupt request can be latched while the first one is being processed. The CPU recognizes the second request as soon as the return from interrupt clears the I bit.

1 = IRQ2 pin interrupt request pending
0 = No IRQ2 pin interrupt request pending

ACK — $\overline{\text{IRQ}}$ Pin Interrupt Request Acknowledge

Setting this write-only bit clears the $\overline{\text{IRQ}}$ pin interrupt request latch. ACK always reads as zero.

1 = $\overline{\text{IRQ}}$ pin interrupt request latch cleared
0 = No meaning or effect

ACK2 — IRQ2 Pin Interrupt Request Acknowledge

Setting this write-only bit clears the IRQ2 pin interrupt request latch. ACK2 always reads as zero.

1 = IRQ2 pin interrupt request latch cleared
0 = No meaning or effect

4.1.3 Timer Interrupts

The timer can generate the following interrupts:

- Timer overflow interrupt
- Real-time interrupt

Setting the I bit in the condition code register disables timer interrupts.

4.1.3.1 Timer Overflow Interrupt

A timer overflow interrupt request occurs if the timer overflow flag, TOF, becomes set while the timer overflow interrupt enable bit, TOIE, is also set. TOF and TOIE are in the timer status and control register. (See **8.1 Timer Status and Control Register (TSCR)**.)

4.1.3.2 Real-Time Interrupt

A real-time interrupt request occurs if the real-time interrupt flag, RTIF, becomes set while the real-time interrupt enable bit, RTIE, is also set. RTIF and RTIE are in the timer status and control register. (See **8.1 Timer Status and Control Register (TSCR)**.)

4.1.4 DTMF Receiver Interrupt

A DTMF receiver interrupt request occurs when the DTMF receiver detects a valid tone pair while the DTMF receiver interrupt enable bit, DRIE, is set. DRIE is in the DTMF receiver control register. (See **9.2.2 DTMF Receiver Control Register (DCR)**.)

Setting the I bit in the condition code register disables DTMF receiver interrupts.

4.2 Interrupt Processing

The CPU does the following things to begin servicing an interrupt:

- Stores the CPU registers on the stack in the order shown in Figure 4-3
- Sets the I bit in the condition code register to prevent further interrupts
- Loads the program counter with the contents of the appropriate interrupt vector locations:
  - — $1FFC and $1FFD (software interrupt vector)
  - — $1FFA and $1FFB (IRQ interrupt vector)
  - — $1FF8 and $1FF9 (IRQ2 interrupt vector)
  - — $1FF6 and $1FF7 (timer interrupt vector)
  - — $1FF4 and $1FF5 (DTMF receiver interrupt vector)

The return from interrupt (RTI) instruction causes the CPU to recover the CPU registers from the stack as shown in Figure 4-3.

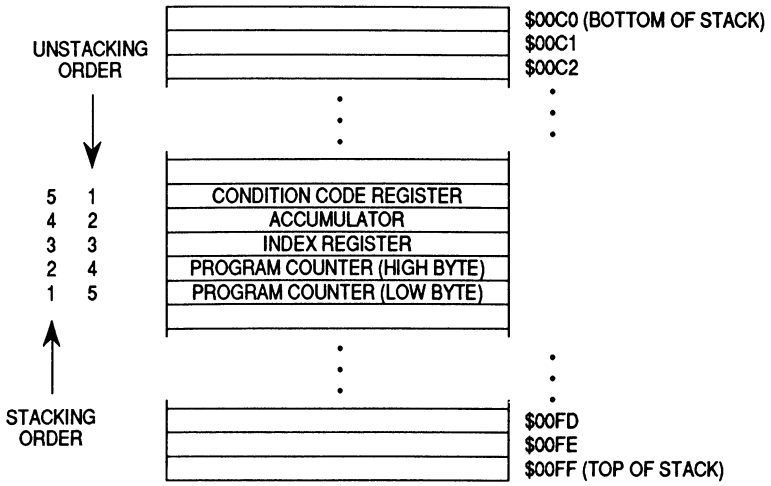


Figure 4-3. Interrupt Stacking Order

Table 4-1 summarizes the reset and interrupt sources and vector assignments.

Table 4-1. Reset/Interrupt Vector Addresses

Function	Source	Local Mask	Global Mask	Priority (1 = Highest)	Vector Address
Reset	Power-On RESET Pin COP Watchdog	None	None None None	1 1 1	$1FFE–$1FFF
Software Interrupt (SWI)	User Code	None	None	Same Priority As Instruction	$1FFC–$1FFD
External Interrupt	$\overline{\text{IRQ}}$ Pin IRQ2 Pin	IRQM Bit IRQ2M Bit	I Bit	2	$1FFA–$1FFB $1FF8–$1FF9
Timer Interrupts	TOF Bit RTIF Bit	TOFE Bit RTIE Bit	I Bit	3	$1FF6–$1FF7
DTMF Interrupt	DIRQ	DRIE Bit	I Bit	4	$1FF4–$1FF5

NOTE: The COP watchdog is a mask option.

Figure 4-4 shows the sequence of events caused by an interrupt.

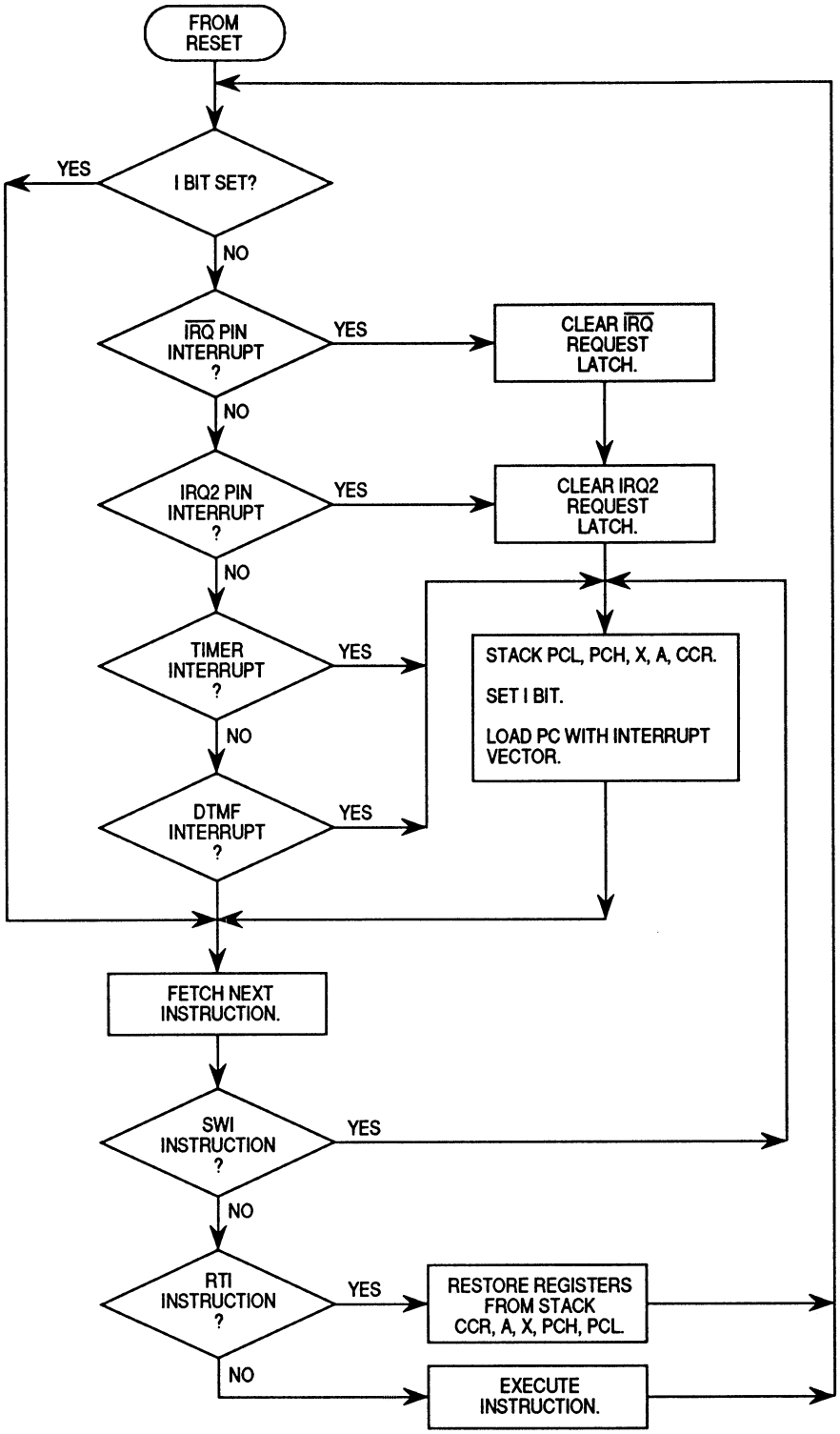


Figure 4-4. Interrupt Flowchart

SECTION 5
RESETS

This section describes how resets initialize the MCU.

5.1 Reset Sources

The following sources can generate resets:

- Power-on reset (POR) circuit
- $\overline{\text{RESET}}$ pin
- COP watchdog

A reset immediately stops the operation of the instruction being executed, initializes certain control bits, and loads the program counter with a user-defined reset vector address. Figure 5-1 is a block diagram of the reset sources.

5.1.1 Power-On Reset

A positive transition on the V_{DD} pin generates a power-on reset. The power-on reset is strictly for power-up conditions and cannot be used to detect drops in power supply voltage.

A 4064 t_{CYC} (internal clock cycle) delay after the oscillator becomes active allows the clock generator to stabilize. If the $\overline{\text{RESET}}$ pin is at logic zero at the end of 4064 t_{CYC}, the MCU remains in the reset condition until the signal on the $\overline{\text{RESET}}$ pin goes to logic one.

5.1.2 External Reset

A logic zero applied to the $\overline{\text{RESET}}$ pin for one and one-half t_{CYC} generates an external reset. A Schmitt trigger senses the logic level at the $\overline{\text{RESET}}$ pin.

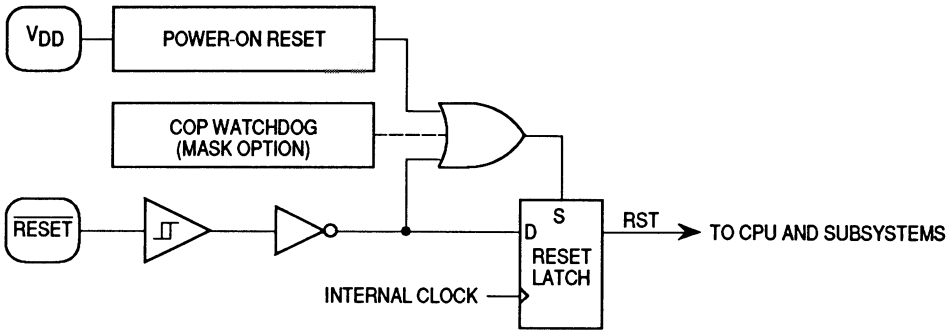


Figure 5-1. Reset Sources

5.1.3 Computer Operating Properly (COP) Reset

A timeout of the COP watchdog generates a COP reset. The COP watchdog is part of a software error detection system and must be cleared periodically to start a new timeout period. (See **8.3 COP Watchdog**.) To clear the COP watchdog and prevent a COP reset, write a logic zero to bit 0 (COPC) of the COP register at location $1FF0. The COP register, shown in Figure 5-2, is a write-only register that returns the contents of a ROM location when read.

The COP watchdog function is a mask option.

COPR — COP Register **$1FF0**

	Bit 7	6	5	4	3	2	1	Bit 0
	—	—	—	—	—	—	—	COPC
RESET:	—	—	—	—	—	—	—	0

Figure 5-2. COP Register (COPR)

COPC — COP Clear

COPC is a write-only bit. Periodically writing a logic zero to COPC prevents the COP watchdog from resetting the MCU.

5.2 Reset States

The following paragraphs describe how resets initialize the MCU.

5.2.1 CPU

A reset has the following effects on the CPU:

- Loads the stack pointer with $FF
- Sets the I bit in the condition code register, inhibiting interrupts
- Loads the program counter with the user-defined reset vector from locations $1FFE and $1FFF
- Clears the stop latch, enabling the CPU clock
- Clears the wait latch, waking the CPU from the wait mode

5.2.2 IRQ Interrupt Status and Control Register (ISCR)

A reset clears the IRQ status and control register, with the following results:

- Clears the IRQ and IRQ2 interrupt request latches
- Clears REQ and REQ2, the IRQ and IRQ2 interrupt request flags
- Clears IRQM and IRQ2M, the local $\overline{\text{IRQ}}$ pin and IRQ2 pin interrupt masks
- Clears the EDGE bit, making the IRQ2 pin positive edge-sensitive

5.2.3 I/O Port Registers

A reset has the following effects on I/O port registers:

- Clears ports A, B, C, and D data direction registers so that all I/O port pins are inputs
- Has no effect on port data registers

5.2.4 Multifunction Timer

A reset has the following effects on the timer status and control register and the timer counter register:

- Clears TOF and RTIF, the timer overflow interrupt and real-time interrupt flags
- Clears TOIE and RTIE, the timer overflow interrupt and real-time interrupt enable bits
- Sets RT1 and RT0, the real-time interrupt rate select bits, selecting the maximum COP timeout period
- Clears the entire counter chain, including the timer counter register

5.2.5 COP Watchdog

A reset clears the COP watchdog (if the COP watchdog is enabled by mask option).

5.2.6 DTMF Receiver

A reset clears the DTMF receiver data and status register and the DTMF receiver control register, with the following results:

- Clears DIRQ, the DTMF receiver interrupt flag
- Clears DRIE, the DTMF receiver interrupt enable bit
- Clears D8, D4, D2, and D1, the data code of the last DTMF character detected
- Clears the CLK bit, configuring the DTMF receiver for a bus speed of 2 MHz
- Clears the G1 and G0 bits, selecting the minimum detect and reject guard times

SECTION 6
LOW POWER MODES

This section describes the three low-power modes:

- Stop mode
- Wait mode
- Data-retention mode

6.1 Stop Mode

The STOP instruction puts the MCU in its lowest power-consumption mode and has the following effects on the MCU:

- Clears TOF and RTIF, the timer interrupt flags in the timer status and control register, removing any pending timer interrupts
- Clears TOIE and RTIE, the timer interrupt enable bits in the timer status and control register, disabling further timer interrupts
- Clears the multifunction timer counter
- Clears the I bit in the condition code register, enabling external interrupts
- Stops the internal oscillator, turning off the CPU clock, the timer clock, the COP watchdog, and the DTMF receiver

The STOP instruction does not affect any other registers or any I/O lines.

The following conditions bring the MCU out of stop mode:

- An external interrupt signal on the $\overline{\text{IRQ}}$ pin — A high-to-low transition on the $\overline{\text{IRQ}}$ pin loads the program counter with the contents of locations $1FFA and $1FFB.

NOTE

If the IRQM bit in the IRQ status and control register is set when the STOP instruction executes, the $\overline{\text{IRQ}}$ pin cannot bring the MCU out of stop mode.

- An external interrupt signal on the IRQ2 pin — An active signal on the IRQ2 pin loads the program counter with the contents of locations $1FF8 and $1FF9.

NOTE

If the IRQ2M bit in the IRQ status and control register is set when the STOP instruction executes, the IRQ2 pin cannot bring the MCU out of stop mode.

- External reset — A logic zero on the $\overline{\text{RESET}}$ pin resets the MCU and loads the program counter with the contents of locations $1FFE and $1FFF.

When the MCU exits stop mode, processing resumes after a stabilization delay of 4064 oscillator cycles.

Figure 6-1 shows the sequence of events caused by the STOP instruction.

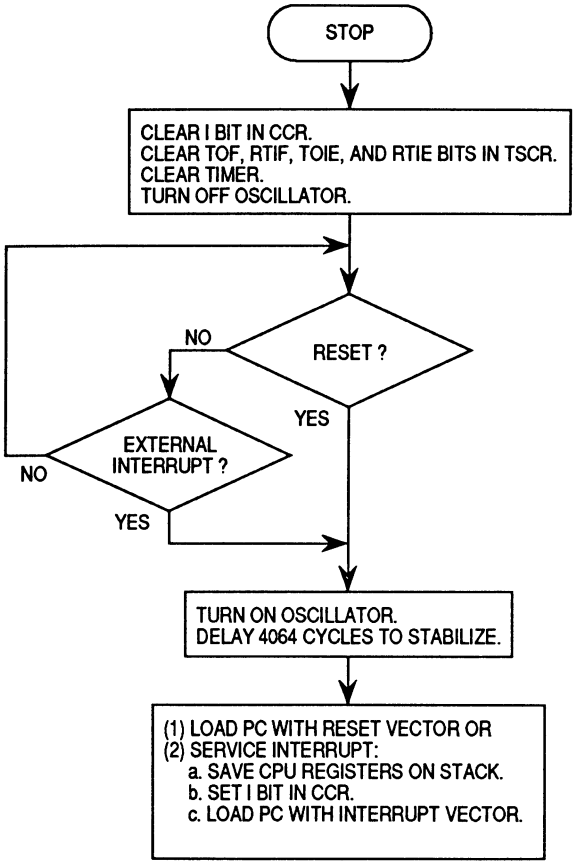


Figure 6-1. STOP Instruction Flowchart

6.2 Wait Mode

The WAIT instruction puts the MCU in an intermediate power-consumption mode and has the following effects on the MCU:

- Clears the I bit in the condition code register, enabling interrupts
- Stops the CPU clock, but allows the internal oscillator and timer clock to continue to run

The WAIT instruction does not affect any other registers or any I/O lines.

The following conditions restart the CPU clock and bring the MCU out of wait mode:

- An external interrupt signal on the $\overline{\text{IRQ}}$ pin — A high-to-low transition on the $\overline{\text{IRQ}}$ pin loads the program counter with the contents of locations $1FFA and $1FFB.

NOTE

If the IRQM bit in the IRQ status and control register is set when the WAIT instruction executes, the IRQ pin cannot bring the MCU out of wait mode.

- An external interrupt signal on the IRQ2 pin — An interrupt signal on the IRQ2 pin loads the program counter with the contents of locations $1FF8 and $1FF9.

NOTE

If the IRQ2M bit in the IRQ status and control register is set when the WAIT instruction executes, the IRQ2 pin cannot bring the MCU out of wait mode.

- A timer interrupt — A timer overflow or a real-time interrupt request loads the program counter with the contents of locations $1FF6 and $1FF7.
- A DTMF receiver interrupt — A qualified DTMF signal loads the program counter with the contents of locations $1FF4 and $1FF5.
- A COP watchdog reset — A timeout of the mask-optional COP watchdog resets the MCU and loads the program counter with the contents of locations $1FFE and $1FFF. Software can enable real-time interrupts so that the MCU can periodically exit wait mode to reset the COP watchdog.
- External reset — A logic zero on the $\overline{\text{RESET}}$ pin resets the MCU and loads the program counter with the contents of locations $1FFE and $1FFF.

Figure 6-2 shows the sequence of events caused by the WAIT instruction.

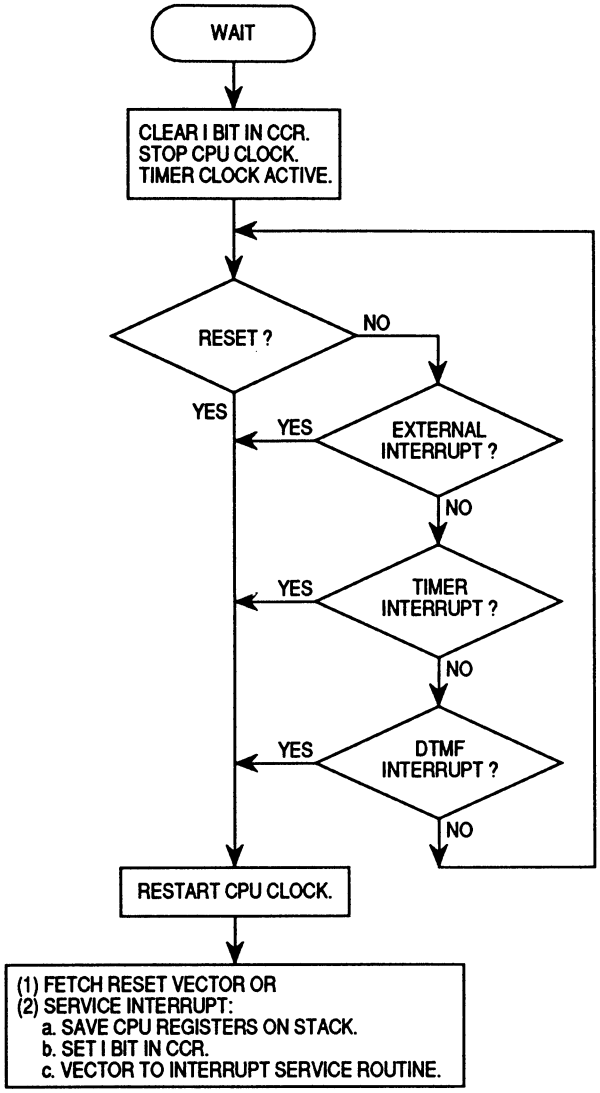


Figure 6-2. WAIT Instruction Flowchart

The following figure shows the effect of the STOP and WAIT instructions on the CPU clock and the timer clock.

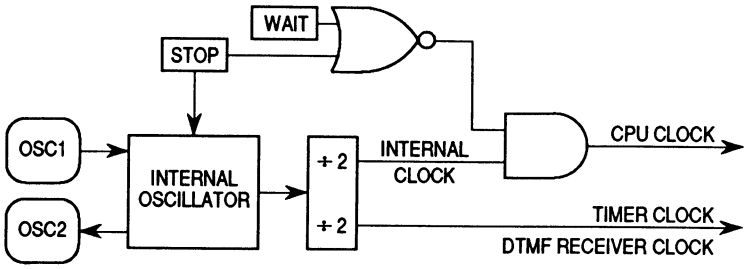


Figure 6-3. STOP/WAIT Clock Logic

6.4 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 data-retention feature allows the MCU to remain in a low power-consumption state during which it retains data, but the CPU cannot execute instructions. To put the MCU in data-retention mode:

1. Drive the $\overline{\text{RESET}}$ pin to logic zero.
2. Lower the V_{DD} voltage. The $\overline{\text{RESET}}$ pin must remain low continuously during data-retention mode.

To take the MCU out of data-retention mode:

1. Return V_{DD} to normal operating voltage.
2. Return the $\overline{\text{RESET}}$ pin to logic one.

SECTION 7
PARALLEL I/O

This section describes the four parallel I/O ports.

7.1 I/O Port Function

The 30 bidirectional I/O pins form four parallel I/O ports. Each I/O pin is programmable as an input or an output in the four data direction registers. Reset configures all I/O pins as inputs.

NOTE

Connect any unused inputs and I/O pins to an appropriate logic level, either V_{DD} or V_{SS}. Although the I/O ports do not require termination for proper operation, termination reduces excess current consumption and the possibility of electrostatic damage.

7.2 Port A

Port A is an 8-bit general-purpose bidirectional I/O port.

7.2.1 Port A Data Register (PORTA)

The port A data register, shown in Figure 7-1, contains a data latch for each of the eight port A pins.

PORTA — Port A Data Register **$0000**

	Bit 7	6	5	4	3	2	1	Bit 0
	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
RESET:	UNAFFECTED BY RESET							

Figure 7-1. Port A Data Register (PORTA)

PA7–PA0 — Port A Data Bits

These read/write bits are software-programmable. Data direction of each bit is under the control of the corresponding bit in data direction register A.

7.2.2 Data Direction Register A (DDRA)

Data direction register A, shown in Figure 7-2, determines whether each port A pin is an input or an output. Writing a logic one to a DDRA bit enables the output buffer for the corresponding port A pin; a logic zero disables the output buffer. A reset initializes all DDRA bits to logic zero, configuring all port A pins as inputs.

DDRA — Data Direction Register A **$0004**

	Bit 7	6	5	4	3	2	1	Bit 0
	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
RESET:	0	0	0	0	0	0	0	0

Figure 7-2. Data Direction Register A (DDRA)

DDRA7–DDRA0 — Port A Data Direction Bits

These read/write bits control port A data direction.

1 = Corresponding port A pin configured as output
0 = Corresponding port A pin configured as input

NOTE

Avoid glitches on port A pins by writing to the port A data register before changing data direction register A bits from 0 to 1.

Figure 7-3 shows the port A I/O logic.

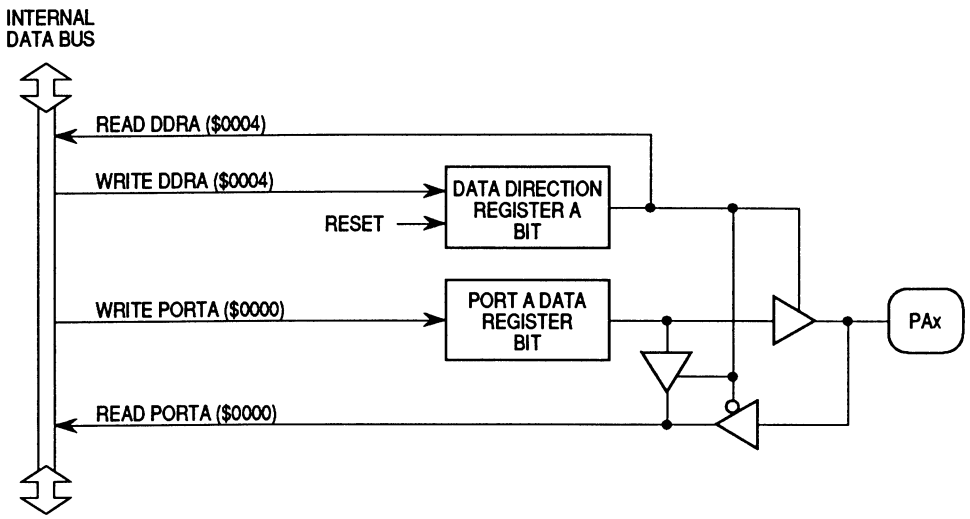


Figure 7-3. Port A I/O Circuit

When a port A pin is programmed as an output, reading the port bit reads the value of the data latch and not the voltage on the pin. When a port A pin is programmed as an input, reading the port bit reads the voltage level on the pin. The data latch can always be written, regardless of the state of its DDRA bit. Table 7-1 summarizes the operation of the port A pins.

Table 7-1. Port A Pin Functions

DDRA Bit	PORTA Bit	I/O Pin Mode	Accesses to DDRA	Accesses to PORTA	
			Read/Write	Read	Write
0	X	Input, hi-Z	DDRA7–0	Pin	NOTE 2
1	X	Output	DDRA7–0	PA7–0	PA7–0

NOTES:
1. X = don't care.
2. Writing affects data register, but does not affect input.

7.3 Port B

Port B is a 6-bit general-purpose bidirectional I/O port.

7.3.1 Port B Data Register (PORTB)

The port B data register, shown in Figure 7-4, contains a data latch for each of the six port B pins.

PORTB — Port B Data Register **$0001**

	Bit 7	6	5	4	3	2	1	Bit 0
	0	0	PB5	PB4	PB3	PB2	PB1	PB0
RESET:	UNAFFECTED BY RESET							

Figure 7-4. Port B Data Register (PORTB)

PB5–PB0 — Port B Data Bits

These read/write bits are software-programmable. Data direction of each bit is under the control of the corresponding bit in data direction register B.

Bits 7 and 6 — Not used

Bits 7 and 6 always read as logic zeros. Writes to these bits have no effect.

7.3.2 Data Direction Register B (DDRB)

Data direction register B, shown in Figure 7-5, determines whether each port B pin is an input or an output. Writing a logic one to a DDRB bit enables the output buffer for the corresponding port B pin; a logic zero disables the output buffer. A reset initializes all DDRB bits to logic zero, configuring all port B pins as inputs.

DDRB — Data Direction Register B **$0005**

	Bit 7	6	5	4	3	2	1	Bit 0
	0	0	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
RESET:	—	—	0	0	0	0	0	0

Figure 7-5. Data Direction Register B (DDRB)

DDRB5–DDRB0 — Port B Data Direction Bits
These read/write bits control port B data direction.
1 = Corresponding port B pin configured as output
0 = Corresponding port B pin configured as input

Bits 7 and 6 — Not used
Bits 7 and 6 always read as logic zeros.

NOTE

Avoid glitches on port B pins by writing to the port B data register before changing data direction register B bits from 0 to 1.

Figure 7-6 shows the port B I/O logic.

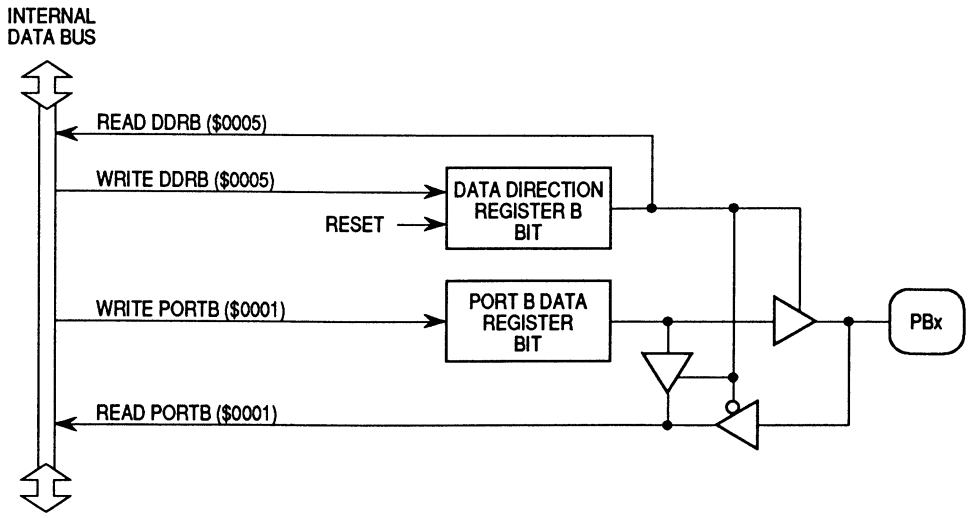


Figure 7-6. Port B I/O Circuit

When a port B pin is programmed as an output, reading the port bit reads the value of the data latch and not the voltage on the pin. When a port B pin is programmed as an input, reading the port bit reads the voltage level on the pin. The data latch can always be written, regardless of the state of its DDRB bit. Table 7-2 summarizes the operation of the port B pins.

Table 7-2. Port B Pin Functions

DDRB Bit	PORTB Bit	I/O Pin Mode	Accesses to DDRB	Accesses to PORTB	
			Read/Write	Read	Write
0	X	Input, hi-Z	DDRB5–0	Pin	NOTE 2
1	X	Output	DDRB5–0	PB5–0	PB5–0

NOTES:
1. X = don't care.
2. Writing affects data register, but does not affect input.

7.4 Port C

Port C is an 8-bit general-purpose bidirectional I/O port.

7.4.1 Port C Data Register (PORTC)

The port C data register, shown in Figure 7-7, contains a data latch for each of the eight port C pins.

PORTC — Port C Data Register **$0002**

	Bit 7	6	5	4	3	2	1	Bit 0
	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
RESET:	UNAFFECTED BY RESET							

Figure 7-7. Port C Data Register (PORTC)

PC7–PC0 — Port C Data Bits

These read/write bits are software-programmable. Data direction of each bit is under the control of the corresponding bit in data direction register C.

7.4.2 Data Direction Register C (DDRC)

Data direction register C, shown in Figure 7-8, determines whether each port C pin is an input or an output. Writing a logic one to a DDRC bit enables the output buffer for the corresponding port C pin; a logic zero disables the output buffer. A reset initializes all DDRC bits to logic zero, configuring all port C pins as inputs.

DDRC — Data Direction Register C **$0006**

	Bit 7	6	5	4	3	2	1	Bit 0
	DDRC7	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0
RESET:	0	0	0	0	0	0	0	0

Figure 7-8. Data Direction Register C (DDRC)

DDRC7–DDRC0 — Port C Data Direction Bits

These read/write bits control port C data direction.

1 = Corresponding port C pin configured as output

0 = Corresponding port C pin configured as input

NOTE

Avoid glitches on port C pins by writing to the port C data register before changing data direction register C bits from 0 to 1.

Figure 7-9 shows the port C I/O logic.

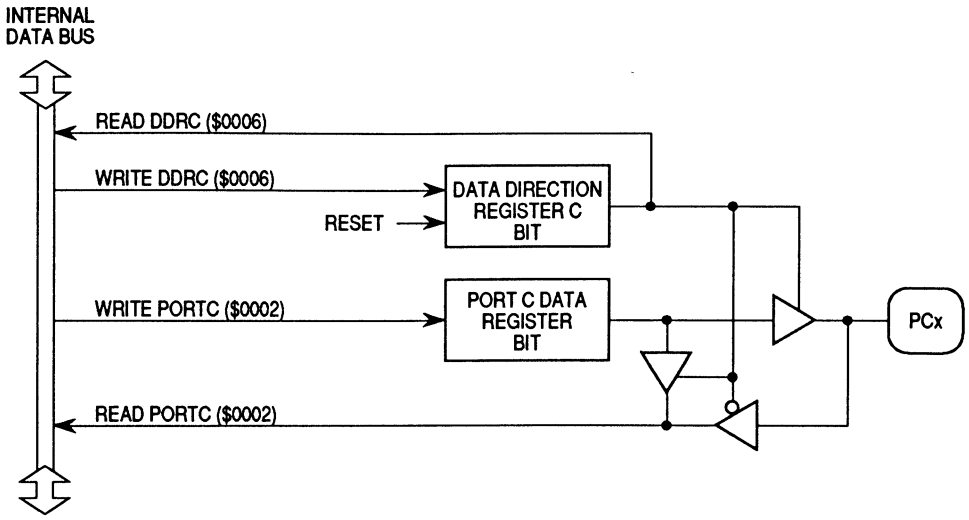


Figure 7-9. Port C I/O Circuit

When a port C pin is programmed as an output, reading the port bit reads the value of the data latch and not the voltage on the pin. When a port C pin is programmed as an input, reading the port bit reads the voltage level on the pin. The data latch can always be written, regardless of the state of its DDRC bit. Table 7-3 summarizes the operation of the port C pins.

Table 7-3. Port C Pin Functions

DDRC Bit	PORTC Bit	I/O Pin Mode	Accesses to DDRC	Accesses to PORTC	
			Read/Write	Read	Write
0	X	Input, hi-Z	DDRC7–0	Pin	NOTE 2
1	X	Output	DDRC7–0	PC7–0	PC7–0

NOTES:
1. X = don't care.
2. Writing affects data register, but does not affect input.

7.5 Port D

Port D is an 8-bit general-purpose bidirectional I/O port.

7.5.1 Port D Data Register (PORTD)

The port D data register, shown in Figure 7-10, contains a data latch for each of the eight port D pins.

PORTD — Port D Data Register **$0003**

	Bit 7	6	5	4	3	2	1	Bit 0
	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
RESET:	UNAFFECTED BY RESET							

Figure 7-10. Port D Data Register (PORTD)

PD7–PD0 — Port D Data Bits

These read/write bits are software-programmable. Data direction of each bit is under the control of the corresponding bit in data direction register D.

7.5.2 Data Direction Register D (DDRD)

Data direction register D, shown in Figure 7-11, determines whether each port D pin is an input or an output. Writing a logic one to a DDRD bit enables the output buffer for the corresponding port D pin; a logic zero disables the output buffer. A reset initializes all DDRD bits to logic zero, configuring all port D pins as inputs.

DDRD — Data Direction Register D **$0007**

	Bit 7	6	5	4	3	2	1	Bit 0
	DDRD7	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0
RESET:	0	0	0	0	0	0	0	0

Figure 7-11. Data Direction Register D (DDRD)

DDRD7–DDRD0 — Port D Data Direction Bits

These read/write bits control port D data direction.

1 = Corresponding port D pin configured as output

0 = Corresponding port D pin configured as input

NOTE

Avoid glitches on port D pins by writing to the port D data register before changing data direction register D bits from 0 to 1.

Figure 7-12 shows the port D I/O logic.

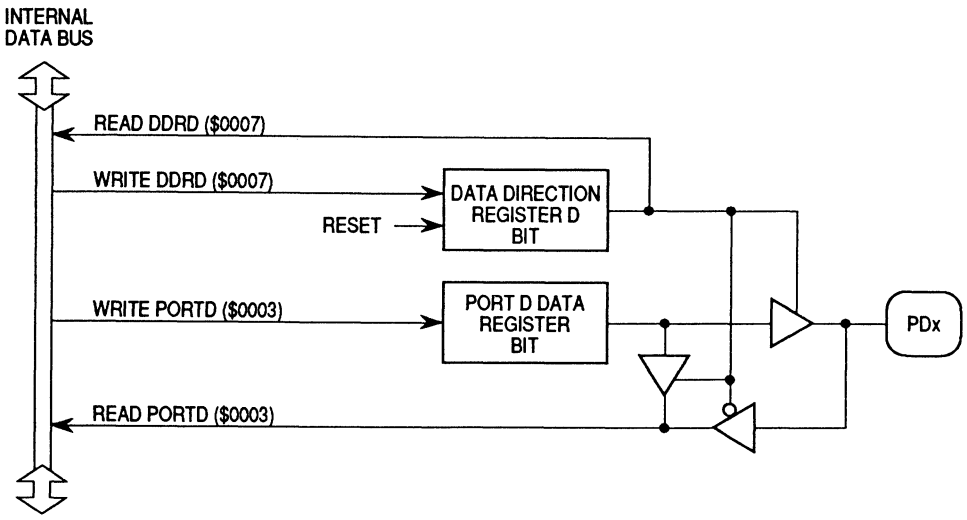


Figure 7-12. Port D I/O Circuit

When a port D pin is programmed as an output, reading the port bit reads the value of the data latch and not the voltage on the pin. When a port D pin is programmed as an input, reading the port bit reads the voltage level on the pin. The data latch can always be written regardless of the state of its DDRD bit. Table 7-4 summarizes the operation of the port D pins.

Table 7-4. Port D Pin Functions

DDRD Bit	PORTD Bit	I/O Pin Mode	Accesses to DDRD	Accesses to PORTD	
			Read/Write	Read	Write
0	X	Input, hi-Z	DDRD7–0	Pin	NOTE 2
1	X	Output	DDRD7–0	PD7–0	PD7–0

NOTES:
1. X = don't care.
2. Writing affects data register, but does not affect input.

SECTION 8
MULTIFUNCTION TIMER

This section describes the operation of the timer and the COP watchdog. Figure 8-1 shows the organization of the timer subsystem.

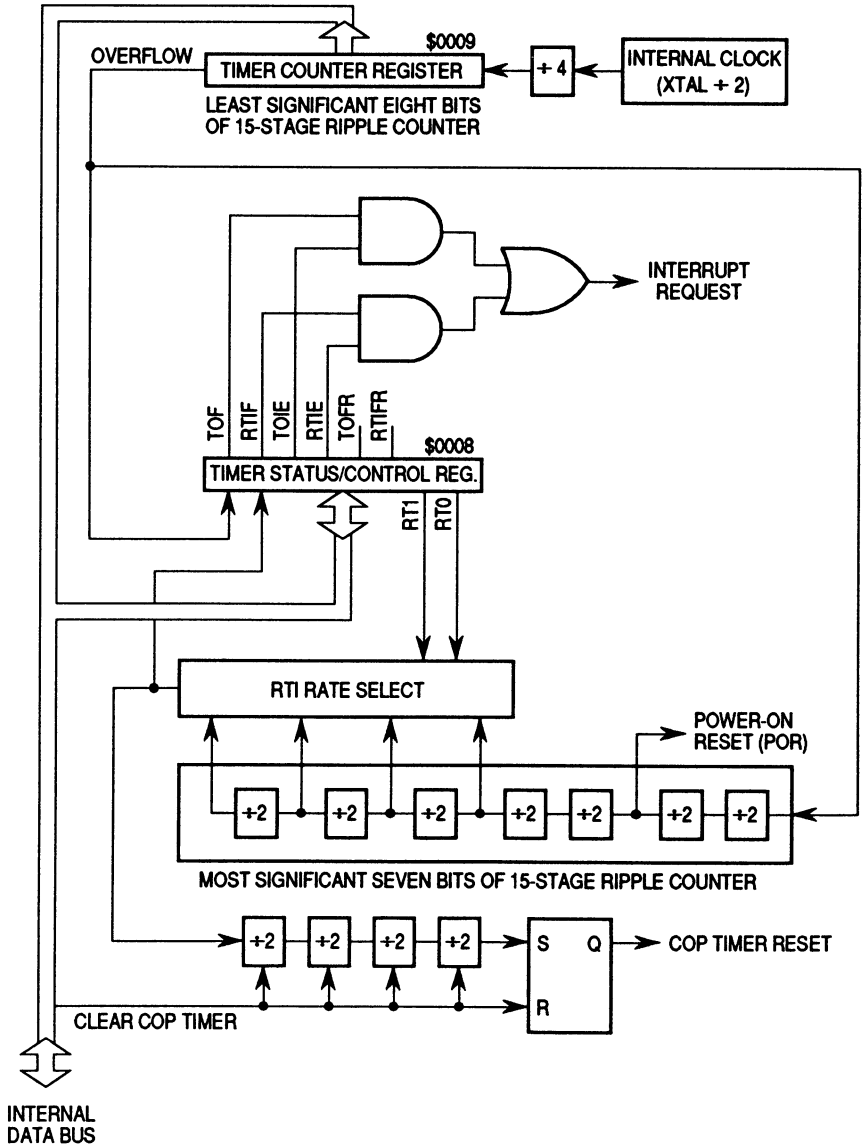


Figure 8-1. Timer Block Diagram

8.1 Timer Status and Control Register (TSCR)

Timer interrupt flags, timer interrupt enable bits, and real-time interrupt rate select bits are in the read/write timer status and control register.

TSCR — Timer Status and Control Register **$0008**

	Bit 7	6	5	4	3	2	1	Bit 0
	TOF	RTIF	TOIE	RTIE	TOFR	RTIFR	RT1	RT0
RESET:	0	0	0	0	U	U	1	1

U = UNAFFECTED

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

TOF — Timer Overflow Flag

This read-only bit becomes set when the first eight stages of the counter roll over from $FF to $00. TOF generates a timer overflow interrupt request if TOIE is also set. Clear TOF by writing a logic one to the TOFR bit. Writing to TOF has no effect.

RTIF — Real-Time Interrupt Flag

This read-only bit becomes set when the selected RTI output becomes active. RTIF generates a real-time interrupt request if RTIE is also set. Clear RTIF by writing a logic one to the RTIFR bit. Writing to RTIF has no effect.

TOIE — Timer Overflow Interrupt Enable

This read/write bit enables timer overflow interrupts.

1 = Timer overflow interrupts enabled
0 = Timer overflow interrupts disabled

RTIE — Real-Time Interrupt Enable

This read/write bit enables real-time interrupts

1 = Real-time interrupts enabled
0 = Real-time interrupts disabled

TOFR — Timer Overflow Flag Reset

Writing a logic one to this write-only bit clears the TOF bit. TOFR always reads as logic zero.

RTIFR — Real-Time Interrupt Flag Reset

Setting this write-only bit clears the RTIF bit. RTIFR always reads as logic zero.

RT1, RT0 — Real-Time Interrupt Selects 1 and 0

These read/write bits select one of four real-time interrupt rates, as shown in Table 8-1. Because the selected RTI output drives the COP watchdog, changing the real-time interrupt rate also changes the counting rate of the COP watchdog.

NOTE

Changing RT1 and RT0 when a COP timeout is imminent or uncertain may cause a real-time interrupt to be missed or an additional real-time interrupt to be generated. Clear the COP timer just before changing RT1 and RT0.

Table 8-1. Real-Time Interrupt Rate Selection

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

8.2 Timer Counter Register (TCNTR)

A 15-stage ripple counter is the core of the timer. The value of the first eight stages is readable at any time from the read-only timer counter register, shown in Figure 8-3.

TCNTR — Timer Counter Register **$0009**

	Bit 7	6	5	4	3	2	1	Bit 0
RESET:	0	0	0	0	0	0	0	0

Figure 8-3. Timer Counter Register (TCNTR)

Power-on clears the entire counter chain and begins clocking the counter. After 4064 cycles, the power-on reset circuit is released, clearing the counter again and allowing the MCU to come out of reset.

A timer overflow function at the eighth counter stage allows a timer interrupt every 1024 internal clock cycles.

8.3 COP Watchdog

Four counter stages at the end of the timer make up the mask-optional computer operating properly (COP) watchdog. (See Figure 8-4.) The COP watchdog is a software error detection system that automatically times out and resets the MCU if not cleared periodically by a program sequence. Writing a logic zero to bit 0 of the COP register clears the COP watchdog and prevents a COP reset.

COPR — COP Register **$1FF0**

	Bit 7	6	5	4	3	2	1	Bit 0
	—	—	—	—	—	—	—	COPC
RESET:	—	—	—	—	—	—	—	0

Figure 8-4. COP Register (COPR)

COPC — COP Clear

This write-only bit resets the COP watchdog. Reading address $1FF0 returns the ROM data at that address.

The STOP instruction turns off the COP watchdog. In applications that depend on the COP watchdog, the STOP instruction can be disabled by a mask option.

SECTION 9
DTMF RECEIVER

This section describes the dual-tone multiple-frequency (DTMF) receiver subsystem.

9.1 DTMF Receiver Operation

The DTMF receiver detects and qualifies DTMF signals. A DTMF signal is a composite of a high frequency tone and a low frequency tone, selected on a telephone keypad and generated by the handset. Figure 9-1 shows the 16 possible frequency combinations and the characters they represent.

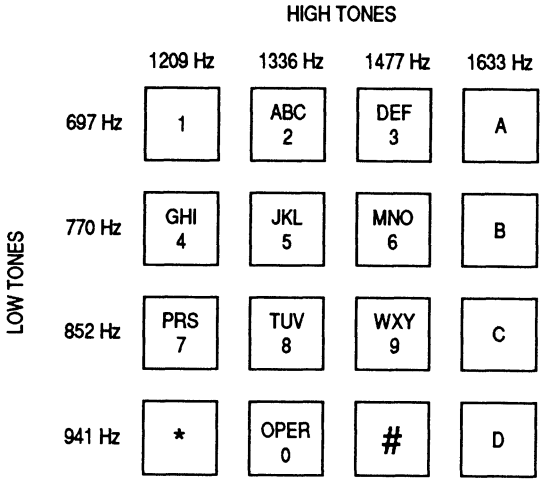


Figure 9-1. DTMF Keypad

9.1.1 Filtering

In the DTMF receiver, the following three filter stages separate the composite tones of DTMF signals:

- Pre-Emphasis Filter
- Band Separation Filters
- Bandpass Filters

Figure 9-2 is a block diagram of the DTMF receiver filters.

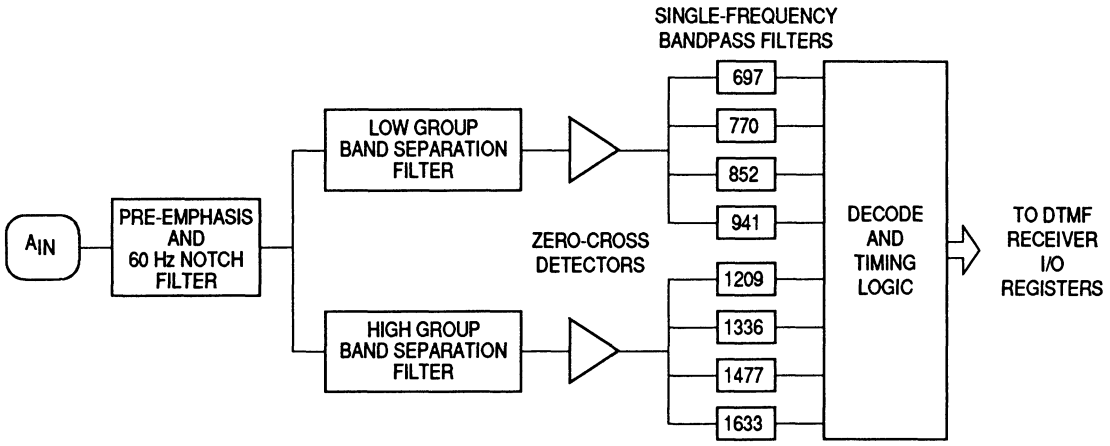


Figure 9-2. DTMF Receiver Filters

9.1.1.1 Pre-Emphasis Filter

The pre-emphasis filter provides high frequency gain to offset the effects of the twist imposed by the telephone lines. The pre-emphasis stage also has a 60 Hz notch filter to reduce power line interference.

9.1.1.2 Band Separation Filters

Two band-separation filters with different center frequencies split the signal into high and low frequency groups.

9.1.1.3 Bandpass Filters

Eight bandpass filters do the final frequency selection. Zero-crossing detectors before the bandpass filters serve as hard limiters that allow only the prime frequencies to pass.

9.1.2 Tone Detection and Qualification

Figure 9-3 is a block diagram of the DTMF tone detection and qualification logic.

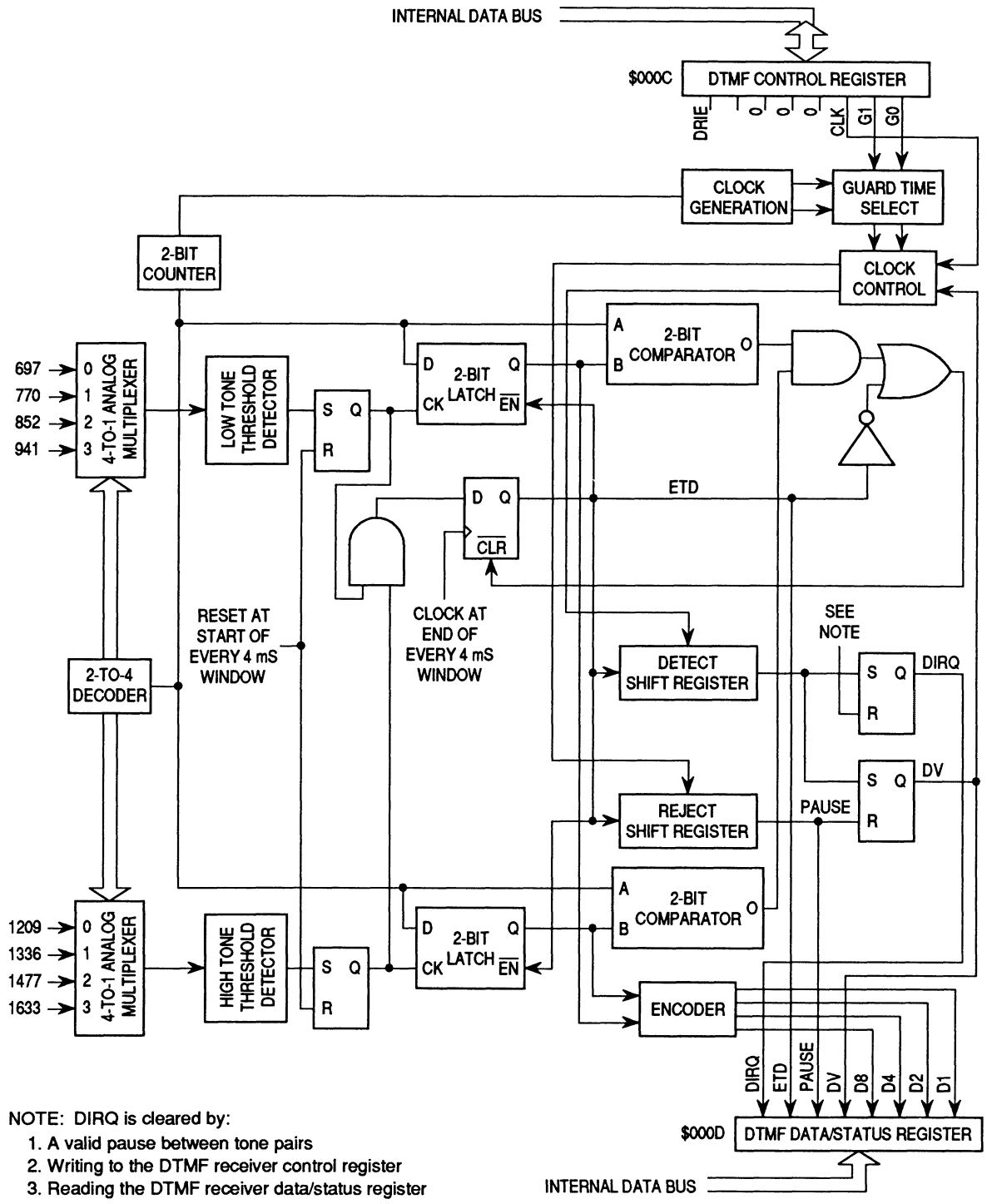


Figure 9-3. DTMF Receiver Logic

9.1.2.1 Tone Pair Detection

Two analog multiplexers continuously present the bandpass filter outputs to a pair of threshold detectors. The high tone and low tone threshold detectors are enabled during a 4-ms sample window and sample many tone pairs during each window. If either threshold detector senses a tone, it sets an RS latch. The high tone and low tone RS latches are ANDed together, and the result is clocked into the early tone detect (ETD) latch at the end of the sample window. The ETD latch is set when both of the threshold detectors detect tones within the same sample window.

When a threshold detector senses a tone, a counter state corresponding to the tone is clocked into a 2-bit latch.

9.1.2.2 Tone Pair Qualification

Two comparators continuously compare the counter value with the values in the 2-bit latches. If the same tone pair is present through successive sample windows, its counter values compare successfully with the values already in the 2-bit latches. If the same tone pair is not present, the comparison is not successful, and the ETD latch is cleared. The qualification logic thus invalidates tone pairs that change from window to window without a valid pause.

After a valid tone pair sets the ETD latch, the ETD signal begins to pass through the detect shift register. The speed of the detect shift register is programmable and determines the detect guard time. A longer detect guard time guards against talk-off, or the mistaken recognition of brief speech tones that simulate DTMF tones. If ETD remains at logic one throughout the detect guard time, the DTMF interrupt request (DIRQ) latch becomes set. If ETD does not remain high, the detect shift register is cleared.

If the tone pair changes or disappears, the ETD latch is cleared, and the negated ETD signal begins to pass through the reject shift register. The speed of the reject shift register is programmable and determines the reject guard time. A longer reject guard time guards against signal drop-outs in a noisy environment. If the negated ETD remains at logic zero throughout the reject guard time, the DIRQ latch is cleared, and the PAUSE bit is set. PAUSE remains set until another tone pair sets the ETD latch.

9.2 DTMF Receiver I/O Registers

The following I/O registers control and monitor the DTMF receiver:

- DTMF Receiver Data and Status Register
- DTMF Receiver Control Register

9.2.1 DTMF Receiver Data and Status Register (DDSR)

The DDSR, shown in Figure 9-4, contains the early tone detect and interrupt flags, the valid pause flag, and the code for the most recently detected character. The DDSR is a read-only register.

DDSR — DTMF Receiver Data and Status Register **$000D**

	Bit 7	6	5	4	3	2	1	Bit 0
	DIRQ	ETD	PAUSE	DV	D8	D4	D2	D1
RESET:	0	0	0	0	0	0	0	0

Figure 9-4. DTMF Receiver Data and Status Register (DDSR)

DIRQ — DTMF Receiver Interrupt Request

DIRQ is a clearable, read-only bit that signals that a tone pair has been detected, qualified, and encoded. The DIRQ bit initiates an interrupt if the DRIE bit in the DTMF receiver control register is set. Clear the DIRQ bit by reading the DDSR or by writing to the DTMF receiver control register (DCR). A valid pause between tone pairs also clears DIRQ.

1 = Valid tone pair present
0 = No valid tone pair present

ETD — Early Tone Detect

ETD is a read-only bit that becomes set when a tone pair is detected and is cleared if the tone pair disappears.

1 = Tone pair detected
0 = No tone pair detected

PAUSE — Pause Detect

PAUSE is a read-only bit that becomes set when a valid pause occurs between tone pairs.

1 = Valid pause detected
0 = No valid pause detected

DV — Data Valid

DV is a read-only bit that signals that a tone pair has been detected, qualified, and encoded. The DV bit follows the DIRQ bit, except that reading the DDSR does not clear DV. Clear the DV bit by writing to the DTMF control register (DCR). A valid pause between tone pairs also clears DV.

D8, D4, D2, D1 — DTMF Receiver Data

These read-only bits contain the code of the most recently detected character. (See Table 9-1.)

Table 9-1. DTMF Receiver Data Codes

Tone Pairs	Character	Output Codes D8:D4:D2:D1
941/1633	D	0000
697/1209	1	0001
697/1336	2	0010
697/1477	3	0011
770/1209	4	0100
770/1336	5	0101
770/1477	6	0110
852/1336	7	0111
852/1447	8	1000
941/1336	9	1001
941/1336	0	1010
941/1209	*	1011
941/1477	#	1100
697/1633	A	1101
770/1633	B	1110
852/1633	C	1111

9.2.2 DTMF Receiver Control Register (DCR)

The DCR, shown in Figure 9-5, controls DTMF interrupt masking, DTMF clock speed, and detect and reject guard times.

DCR — DTMF Receiver Control Register **$000C**

	Bit 7	6	5	4	3	2	1	Bit 0
	DRIE	0	0	0	0	CLK	G1	G0
RESET:	0	0	0	0	0	0	0	0

Figure 9-5. DTMF Receiver Control Register (DCR)

DRIE — DTMF Receiver Interrupt Enable

This read/write bit enables DTMF receiver interrupts.

1 = Interrupt enabled
0 = Interrupt disabled

CLK — Clock Speed Select

This read/write bit is a provision for future digital bus speeds of 4 MHz. Setting the CLK bit divides by two the clock input to the DTMF receiver.

1 = 4 MHz CPU bus speed
0 = 2 MHz CPU bus speed (default)

G1, G0 — Guard Time Select

These read/write bits control detect and reject guard time combinations as shown in Table 9-2. The minimum tone duration after which a tone is considered valid is t_{DETECT}. The minimum time after which a valid pause is recognized is t_{REJECT}.

Table 9-2. Guard Time Options

G1:G0	t_{DETECT} (ms)	t_{REJECT} (ms)
0 0	28–32	20–24
0 1	28–32	28–32
1 0	44–48	20–24
1 1	44–48	28–32

NOTE: f_{OSC} = 4.0 MHz

SECTION 10
SELF-CHECK ROM

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

10.1 Self-Check Tests

To check for proper MCU operation, the self-check ROM performs the following tests:

- I/O test — A functional exercise of ports A, B, C, and D
- RAM test — A complement test of each RAM byte
- Timer test — A test of the counter register and the TOF and RTIF flags
- ROM test — An exclusive OR odd parity check
- DTMF test — A test of DTMF digital logic
- Interrupt test — A test of external and timer interrupts

Figure 10-1 shows the circuit required to execute the self-check tests.

10.2 Self-Check Results

Table 10-1 shows the LED codes that indicate self-check test results.

Table 10-1. Self-Check Circuit LED Codes

PC3	PC2	PC1	PC0	Problem
ON	ON	ON	OFF	I/O FAILURE
ON	ON	OFF	ON	RAM FAILURE
ON	ON	OFF	OFF	TIMER FAILURE
ON	OFF	ON	ON	ROM FAILURE
ON	OFF	ON	OFF	DTMF LOGIC FAILURE
ON	OFF	OFF	ON	INTERRUPT FAILURE
ON	Flashing			NO FAILURES
All Others				DEVICE FAILURE

NOTES:

1. With no DTMF signal applied to the A_{IN} pin, the LEDs display the self-check code continuously.
2. With a DTMF signal applied to the A_{IN} pin, the LEDs periodically pause the self-check code display to display the DTMF receiver data code for about one second. (See Table 9-1.)

10.3 Self-Check Circuit

Figure 10-1 shows the self-check circuit.

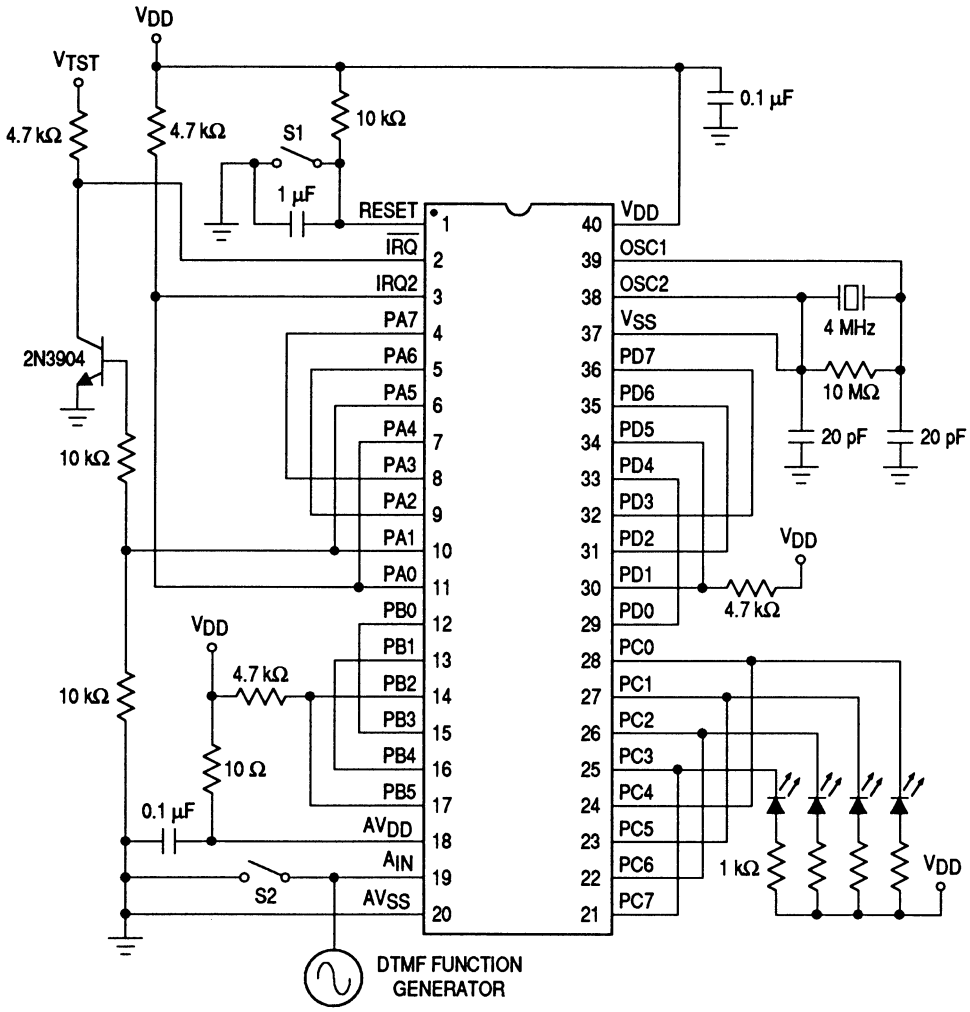


Figure 10-1. Self-Check Circuit

Perform the following steps to activate the self-check tests:

1. Apply V_{TST} to the $\overline{IRQ}$ pin.
2. Apply a logic zero to the $\overline{RESET}$ pin.
3. Apply a logic one to the PD1 pin.
4. Apply a logic one to the PB2 pin.
5. Apply a logic one to the $\overline{RESET}$ pin.

The self-check tests begin on the rising edge of the $\overline{RESET}$ pin.

SECTION 11
INSTRUCTION SET

This section describes the addressing modes and the types of instructions.

11.1 Addressing Modes

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

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

11.1.1 Inherent

Inherent instructions are those that have no operand, such as return from interrupt (RTI) and stop (STOP). Some of the inherent instructions act on data in the CPU registers, such as set carry flag (SEC) and increment accumulator (INCA). Inherent instructions require no memory address and are one byte long. Table 11-1 lists the instructions that use inherent addressing.

Table 11-1. 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 Index Register by Accumulator (Unsigned)	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 $\overline{\text{IRQ}}$ 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 Interrupts and Halt CPU	WAIT

11.1.2 Immediate

Immediate instructions are those that contain a value to be used in an operation with the value in the accumulator or index register. Immediate instructions require no memory address and are two bytes long. The opcode is the first byte and the immediate data value is the second byte. Table 11-2 lists the instructions that use immediate addressing.

Table 11-2. Immediate Addressing Instructions

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

11.1.3 Direct

Direct instructions can access any of the first 256 memory addresses with two bytes. The first byte is the opcode, and the second is the low byte of the operand address. In direct addressing, the CPU automatically uses $00 as the high byte of the operand address. BRSET and BRCLR are three-byte instructions that use direct addressing to access the operand and relative addressing to specify a branch destination. Table 11-3 lists the instructions that use direct addressing.

Table 11-3. Direct Addressing Instructions

Instruction	Mnemonic
Add Memory and Carry to Accumulator	ADC
Add Memory to Accumulator	ADD
Logical AND Memory with Accumulator	AND
Arithmetic Shift Left	ASL
Arithmetic Shift Right	ASR
Clear Bit	BCLR
Bit Test Memory with Accumulator (Logical Compare)	BIT
Branch if Bit Clear	BRCLR
Branch if Bit Set	BRSET
Set Bit	BSET
Clear	CLR
Arithmetic Compare Accumulator with Memory	CMP
Complement	COM
Arithmetic 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
Logical Inclusive OR Memory with Accumulator	ORA
Rotate Left through Carry	ROL
Rotate Right through Carry	ROR
Subtract Memory and Carry from Accumulator	SBC
Store Accumulator in Memory	STA
Store Index Register in Memory	STX
Subtract Memory from Accumulator	SUB
Test for Negative or Zero	TST

11.1.4 Extended

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

When using the Motorola assembler, the programmer does not need to specify whether an instruction is direct or extended. The assembler automatically selects the shortest form of the instruction. Table 11-4 lists the instructions that use extended addressing.

Table 11-4. Extended Addressing Instructions

Instruction	Mnemonic
Add Memory and Carry to Accumulator	ADC
Add Memory to Accumulator	ADD
Logical AND Memory with Accumulator	AND
Bit Test Memory with Accumulator (Logical Compare)	BIT
Arithmetic Compare Accumulator with Memory	CMP
Arithmetic 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
Logical Inclusive OR Memory with Accumulator	ORA
Subtract Memory and Carry from Accumulator	SBC
Store Accumulator in Memory	STA
Store Index Register in Memory	STX
Subtract Memory from Accumulator	SUB

11.1.5 Indexed, No Offset

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

Indexed, no offset instructions are often used to move a pointer through a table or to hold the address of a frequently used RAM or I/O location. Table 11-5 lists the instructions that use indexed, no offset addressing.

11.1.6 Indexed, 8-Bit Offset

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

Indexed, 8-bit offset instructions are useful for selecting the kth element in an n-element table. The table can begin anywhere within the first 256 memory locations and could extend as far as location 510 ($01FE). The k value would typically be in the index register, and the address of the beginning of the table would be in the byte following the opcode. Table 11-5 lists the instructions that use indexed, 8-bit offset addressing.

11.1.7 Indexed, 16-Bit Offset

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

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

As with direct and extended addressing, the Motorola assembler determines the shortest form of indexed addressing. Table 11-5 lists the instructions that use indexed, 16-bit offset addressing.

Table 11-5. Indexed Addressing Instructions

Instruction	Mnemonic	No Offset	8-Bit Offset	16-Bit Offset
Add Memory and Carry to Accumulator	ADC	√	√	√
Add Memory to Accumulator	ADD	√	√	√
Logical AND Memory with Accumulator	AND	√	√	√
Arithmetic Shift Left	ASL	√	√	
Arithmetic Shift Right	ASR	√	√	
Bit Test Memory with Accumulator (Logical Compare)	BIT	√	√	√
Clear	CLR	√	√	
Arithmetic Compare Accumulator with Memory	CMP	√	√	√
Complement	COM	√	√	
Arithmetic 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	√	√	
Logical Inclusive OR Memory with Accumulator	ORA	√	√	√
Rotate Left through Carry	ROL	√	√	
Rotate Right through Carry	ROR	√	√	
Subtract Memory and Carry from Accumulator	SBC	√	√	√
Store Accumulator in Memory	STA	√	√	√
Store Index Register in Memory	STX	√	√	√
Subtract Memory from Accumulator	SUB	√	√	√
Test for Negative or Zero	TST	√	√	

11.1.8 Relative

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

When using the Motorola assembler, the programmer does not need to calculate the offset, because the assembler determines the proper offset and verifies that it is within the span of the branch. Table 11-6 lists the instructions that use relative addressing.

Table 11-6. 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 High	BIH
Branch if Interrupt Line Low	BIL
Branch if Lower	BLO
Branch if Lower or Same	BLS
Branch if Interrupt Mask Clear	BMC
Branch if Minus	BMI
Branch if Interrupt Mask Set	BMS
Branch if Not Equal	BNE
Branch if Plus	BPL
Branch Always	BRA
Branch if Bit Clear	BRCLR
Branch if Bit Set	BRSET
Branch Never	BRN
Branch to Subroutine	BSR

11.2 Instruction Types

The MCU instructions fall into the following five categories:

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

11.2.1 Register/Memory Instructions

Most of these instructions use two operands. One operand is in either the accumulator or the index register. The CPU finds the other operand in memory. Register/memory instructions use all the addressing modes except relative.

Table 11-7 lists the register/memory instructions.

Table 11-7. 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 from Accumulator	SUB
Subtract Memory and Carry from Accumulator	SBC
Logical AND Memory with Accumulator	AND
Logical Inclusive OR Memory with Accumulator	ORA
Arithmetic Compare Accumulator with Memory	CMP
Arithmetic Compare Index Register with Memory	CPX
Bit Test Memory with Accumulator (Logical Compare)	BIT
Multiply Index Register by Accumulator (Unsigned)	MUL

11.2.2 Read-Modify-Write Instructions

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

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

Table 11-8 lists the read-modify-write instructions.

Table 11-8. Read-Modify-Write Instructions

Instruction	Mnemonic
Increment	INC
Decrement	DEC
Clear	CLR
Complement	COM
Negate (Two's 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

11.2.3 Jump/Branch Instructions

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

Bit test and branch instructions cause a branch based on the state of any readable bit in the first 256 memory locations. These three-byte instructions use a combination of direct addressing and relative addressing. The direct address of the byte to be tested is in the byte following the opcode. The third byte is the signed offset byte. The CPU finds the conditional branch destination by adding the third byte to the program counter if the specified bit tests true. The bit to be tested and its condition (set or clear) is part of the opcode. The span of branching is from –128 to +127 from the address of the next location after the branch instruction. The CPU also transfers the tested bit to the carry/borrow bit of the condition code register. Table 11-9 lists the jump and branch instructions.

Table 11-9. Jump and Branch Instructions

Instruction	Mnemonic
Branch Always	BRA
Branch Never	BRN
Branch if Bit Clear	BRCLR
Branch if Bit Set	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	BND
Branch if Equal	BEQ
Branch if Half-Carry Bit Clear	BHCC
Branch if Half-Carry Bit 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	JMP
Jump to Subroutine	JSR

11.2.4 Bit Manipulation Instructions

The CPU can set or clear any writable bit in the first 256 bytes of memory. Port registers, port data direction registers, timer registers, and on-chip RAM locations are in the first 256 bytes of memory. The CPU can also test and branch based on the state of any bit in any of the first 256 memory locations. Bit manipulation instructions use direct addressing. Table 11-10 lists these instructions.

Table 11-10. Bit Manipulation Instructions

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

11.2.5 Control Instructions

These register reference instructions control CPU operation during program execution. Control instructions, listed in Table 11-11, use inherent addressing.

Table 11-11. Control Instructions

Instruction	Mnemonic
Transfer Accumulator to Index Register	TAX
Transfer Index Register to Accumulator	TXA
Set Carry Bit	SEC
Clear Carry Bit	CLC
Set Interrupt Mask	SEI
Clear Interrupt Mask	CLI
Software Interrupt	SWI
Return from Subroutine	RTI
Reset Stack Pointer	RSP
No Operation	NOP
Stop	STOP
Wait	WAIT

11.3 Instruction Set Summary

Table 11-12 shows all MC68HC05F5 instructions in all possible addressing modes. The table shows the operand construction and the execution time in internal clock cycles (t_{CYC}) of each instruction. One internal clock cycle equals two oscillator input cycles. The following legend summarizes the symbols and abbreviations used in Table 11-12.

Abbreviations and Symbols

A	Accumulator	PCH	Program counter high byte
C	Carry/borrow flag	PCL	Program counter low byte
CCR	Condition code register	REL	Relative addressing
dd	Address of operand in direct addressing	rel	Offset byte for relative addressing
dd rr	Address (dd) of operand and offset (rr) of branch instruction for bit test instructions	rr	Offset byte of branch instruction
DIR	Direct addressing	SP	Stack pointer
ee ff	High (ee) and low (ff) bytes of offset in indexed, 16-bit offset addressing	X	Index register
EXT	Extended addressing	Z	Zero flag
ff	Offset byte in indexed, 8-bit offset addressing	•	AND
H	Half-carry flag	–	Not affected
hh ll	High (hh) and low (ll) bytes of operand address in extended addressing	?	If
I	Interrupt mask	—	NOT
ii	Operand byte for immediate addressing	( )	Contents of
IMM	Immediate addressing	←	Is loaded with
INH	Inherent addressing	:	Concatenated with
IX	Indexed, no offset addressing	×	Multiplication
IX1	Indexed, 8-bit offset addressing	–( )	Negation (two's complement)
IX2	Indexed, 16-bit offset addressing	+	Inclusive OR
M	Any memory location (1 byte)	↕	Set if true; clear if not true
N	Negative flag	⊕	Exclusive OR
n	Any bit (7,6,5 . . . 0)	+	Addition
opr	Operand byte	–	Subtraction
PC	Program counter		

Table 11-12. Instruction Set (Sheet 1 of 4)

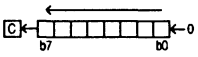
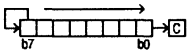
Source Form(s)	Operation	Description	Addressing Mode for Operand	Machine Coding (hexadecimal)		Cycles	Condition Code				
				Opcode	Operand		H	I	N	Z	C
ADC opr	Add with carry	A ← (A) + (M) + C	IMM	A9	ii	2	↕	–	↕	↕	↕
			DIR	B9	dd	3					
			EXT	C9	hh ll	4					
			IX2	D9	ee ff	5					
			IX1	E9	ff	4					
			IX	F9		3					
ADD opr	Add without carry	A ← (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 ← (A) • (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	Arithmetic shift left		DIR	38	dd	5	–	–	↕	↕	↕
ASLA			INH	48		3					
ASLX			INH	58		3					
ASL opr			IX1	68	ff	6					
ASL opr			IX	78		5					
ASR opr	Arithmetic shift right		DIR	37	dd	5	–	–	↕	↕	↕
ASRA			INH	47		3					
ASRX			INH	57		3					
ASR opr			IX1	67	ff	6					
ASR opr			IX	77		5					
BCC rel	Branch if carry bit clear	? C = 0	REL	24	rr	3	–	–	–	–	–
BCLR n opr	Clear bit n	Mn ← 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) • (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	–	–	–	–	–

Table 11-12. 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	–	–	–	–	0
CLI	Clear interrupt mask	I ← 0	INH	9A		2	–	0	–	–	–
CLR opr	Clear register	M ← $00	DIR	3F	dd	5	–	–	0	1	–
CLRA		A ← $00	INH	4F		3					
CLRX		X ← $00	INH	5F		3					
CLR opr		M ← $00	IX1	6F	ff	6					
CLR opr		M ← $00	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	Complement register contents (ones complement)	M ← $\overline{M}$ = $FF – (M)	DIR	33	dd	5	–	–	↕	↕	1
COMA		A ← $\overline{A}$ = $FF – (A)	INH	43		3					
COMX		X ← $\overline{X}$ = $FF – (X)	INH	53		3					
COM opr		M ← $\overline{M}$ = $FF – (M)	IX1	63	ff	6					
COM opr		M ← $\overline{M}$ = $FF – (M)	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	Decrement register contents	M ← (M) – 1	DIR	3A	dd	5	–	–	↕	↕	–
DECA		A ← (A) – 1	INH	4A		3					
DECX		X ← (X) – 1	INH	5A		3					
DEC opr		M ←(M) – 1	IX1	6A	ff	6					
DEC opr		M ←(M) – 1	IX	7A		5					

Table 11-12. Instruction Set (Sheet 3 of 4)

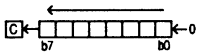
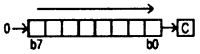
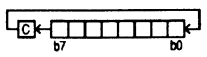
Source Form(s)	Operation	Description	Addressing Mode for Operand	Machine Coding (hexadecimal)		Cycles	Condition Code				
				Opcode	Operand		H	I	N	Z	C
EOR opr	Exclusive OR accumulator contents with memory contents	A ← (A) ⊕ (M)	IMM	A8	ii	2	–	–	↕	↕	–
			DIR	B8	dd	3					
			EXT	C8	hh ll	4					
			IX2	D8	ee ff	5					
			IX1	E8	ff	4					
			IX	F8		3					
INC opr	Increment memory or register contents	M ← (M) + 1	DIR	3C	dd	5	–	–	↕	↕	–
INCA		A ← (A) + 1	INH	4C		3					
INCX		X ← (X) + 1	INH	5C		3					
INC opr		M ← (M) + 1	IX1	6C	ff	6					
INC opr		M ← (M) + 1	IX	7C		5					
JMP opr	Unconditional jump	PC ← 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 ← (PC) + n (n = 1, 2, or 3) Push (PCL); SP ← (SP) – 1 Push (PCH); SP ← (SP) – 1 PC ← 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 ← (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 ← (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	Logical shift left		DIR	38	dd	5	–	–	↕	↕	↕
LSLA			INH	48		3					
LSLX			INH	58		3					
LSL opr			IX1	68	ff	6					
LSL opr			IX	78		5					
LSR opr	Logical shift right		DIR	34	dd	5	–	–	0	↕	↕
LSRA			INH	44		3					
LSRX			INH	54		3					
LSR opr			IX1	64	ff	6					
LSR opr			IX	74		5					
MUL	Unsigned multiply	X : A ← (X) × (A)	INH	42		11	0	–	–	–	0
NEG opr	Negate memory or register contents (twos complement)	M ← –(M) = $00 – (M)	DIR	30	dd	5	–	–	↕	↕	↕
NEGA		A ← –(A) = $00 – (A)	INH	40		3					
NEGX		X ← –(X) = $00 – (X)	INH	50		3					
NEG opr		M ← –(M) = $00 – (M)	IX1	60	ff	6					
NEG opr		M ← –(M) = $00 – (M)	IX	70		5					
NOP	No operation		INH	9D		2	–	–	–	–	–
ORA opr	Inclusive OR accumulator contents with memory contents	A ← (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	Rotate left through carry		DIR	39	dd	5	–	–	↕	↕	↕
ROLA			INH	49		3					
ROLX			INH	59		3					
ROL opr			IX1	69	ff	6					
ROL opr			IX	79		5					

Table 11-12. Instruction Set (Sheet 4 of 4)

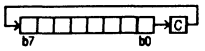
Source Form(s)	Operation	Description	Addressing Mode for Operand	Machine Coding (hexadecimal) Opcode	Machine Coding (hexadecimal) Operand	Cycles	Condition Code H	I	N	Z	C
ROR opr RORA RORX ROR opr ROR opr	Rotate right through carry		DIR INH INH IX1 IX	36 46 56 66 76	dd ff 	5 3 3 6 5	–	–	↕	↕	↕
RSP	Reset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

Table 11-13 is an opcode map of the M68HC05 instruction set.

Table 11-13. Opcode Map

	Bit-Manipulation		Branch	Read-Modify-Write					Control		Register/Memory						
	DIR	DIR	REL	DIR	INH	INH	IX1	IX	INH	INH	IMM	DIR	EXT	IX2	IX1	IX	
HI / LO	0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111	8 1000	9 1001	A 1010	B 1011	C 1100	D 1101	E 1110	F 1111	HI / LO
0 0000	5 BRSET0 3 DIR	5 BSET0 2 DIR	3 BRA 2 REL	5 NEG 2 DIR	3 NEGA 1 INH		6 NEG 2 IX1	5 NEG 1 IX	9 RTI 1 INH		2 SUB 2 IMM	3 SUB 2 DIR	4 SUB 3 EXT	5 SUB 3 IX2	4 SUB 2 IX1	3 SUB 1 IX	0 0000
1 0001	5 BRCLR0 3 DIR	5 BCLR0 2 DIR	3 BRN 2 REL						6 RTS 1 INH		2 CMP 2 IMM	3 CMP 2 DIR	4 CMP 3 EXT	5 CMP 3 IX2	4 CMP 2 IX1	3 CMP 1 IX	1 0001
2 0010	5 BRSET1 3 DIR	5 BSET1 2 DIR	3 BHI 2 REL		11 MUL 1 INH						2 SBC 2 IMM	3 SBC 2 DIR	4 SBC 3 EXT	5 SBC 3 IX2	4 SBC 2 IX1	3 SBC 1 IX	2 0010
3 0011	5 BRCLR1 3 DIR	5 BSCLR1 2 DIR	3 BLS 2 REL	5 COM 2 DIR	3 COMA 1 INH	3 COMX 1 INH	6 COM 2 IX1	5 COM 1 IX	10 SWI 1 INH		2 CPX 2 IMM	3 CPX 2 DIR	4 CPX 3 EXT	5 CPX 3 IX2	4 CPX 2 IX1	3 CPX 1 IX	3 0011
4 0100	5 BRSET2 3 DIR	5 BSET2 2 DIR	3 BCC 2 REL	5 LSR 2 DIR	3 LSRA 1 INH	3 LSRX 1 INH	6 LSR 2 IX1	5 LSR 1 IX			2 AND 2 IMM	3 AND 2 DIR	4 AND 3 EXT	5 AND 3 IX2	4 AND 2 IX1	3 AND 1 IX	4 0100
5 0101	5 BRCLR2 3 DIR	5 BCLR2 2 DIR	3 BCS 2 REL								2 BIT 2 IMM	3 BIT 2 DIR	4 BIT 3 EXT	5 BIT 3 IX2	4 BIT 2 IX1	3 BIT 1 IX	5 0101
6 0110	5 BRSET3 3 DIR	5 BSET3 2 DIR	3 BNE 2 REL	5 ROR 2 DIR	3 RORA 1 INH	3 RORX 1 INH	6 ROR 2 IX1	5 ROR 1 IX			2 LDA 2 IMM	3 LDA 2 DIR	4 LDA 3 EXT	5 LDA 3 IX2	4 LDA 2 IX1	3 LDA 1 IX	6 0110
7 0111	5 BRCLR3 3 DIR	5 BCLR3 2 DIR	3 BEQ 2 REL	5 ASR 2 DIR	3 ASRA 1 INH	3 ASRX 1 INH	6 ASR 2 IX1	5 ASR 1 IX		2 TAX 1 INH		4 STA 2 DIR	5 STA 3 EXT	6 STA 3 IX2	5 STA 2 IX1	4 STA 1 IX	7 0111
8 1000	5 BRSET4 3 DIR	5 BSET4 2 DIR	3 BHCC 2 REL	5 LSL 2 DIR	3 LSLA 1 INH	3 LSLX 1 INH	6 LSL 2 IX1	5 LSL 1 IX		2 CLC 1 INH	2 EOR 2 IMM	3 EOR 2 DIR	4 EOR 3 EXT	5 EOR 3 IX2	4 EOR 2 IX1	3 EOR 1 IX	8 1000
9 1001	5 BRCLR4 3 DIR	5 BCLR4 2 DIR	3 BHCS 2 REL	5 ROL 2 DIR	3 ROLA 1 INH	3 ROLX 1 INH	6 ROL	5 ROL 1 IX		2 SEC 1 INH	2 ADC 2 IMM	3 ADC 2 DIR	4 ADC 3 EXT	5 ADC 3 IX2	4 ADC 2 IX1	3 ADC 1 IX	9 1001
A 1010	5 BRSET5 3 DIR	5 BSET5 2 DIR	3 BPL 2 REL	5 DEC 2 DIR	3 DECA 1 INH	3 DECX 1 INH	6 DEC 2 INH	5 DEC 1 IX		2 CLI 1 INH	2 ORA 2 IMM	3 ORA 2 DIR	4 ORA 3 EXT	5 ORA 3 IX2	4 ORA 2 IX1	3 ORA 1 IX	A 1010
B 1011	5 BRCLR5 3 DIR	5 BCLR5 2 DIR	3 BMI 2 REL							2 SEI 1 INH	2 ADD 2 IMM	3 ADD 2 DIR	4 ADD 3 EXT	5 ADD 3 IX2	4 ADD 2 IX1	3 ADD 1 IX	B 1011
C 1100	5 BRSET6 3 DIR	5 BSET6 2 DIR	3 BMC 2 REL	5 INC 2 DIR	3 INCA 1 INH	3 INCX 1 INH	6 INC 2 INH	5 INC 1 IX		2 RSP 1 INH		2 JMP 2 DIR	3 JMP 3 EXT	4 JMP 3 IX2	3 JMP 2 IX1	2 JMP 1 IX	C 1100
D 1101	5 BRCLR6 3 DIR	5 BCLR6 2 DIR	3 BMS 2 REL	4 TST 2 DIR	3 TSTA 1 INH	3 TSTX 1 INH	6 TST 2 INH	4 TST 1 IX		2 NOP 1 INH	6 BSR 2 REL	5 JSR 2 DIR	6 JSR 3 EXT	7 JSR 3 IX2	6 JSR 2 IX1	5 JSR 1 IX	D 1101
E 1110	5 BRSET7 3 DIR	5 BSET7 2 DIR	3 BIL 2 REL						2 STOP 1 INH		2 LDX 2 IMM	3 LDX 2 DIR	4 LDX 3 EXT	5 LDX 3 IX2	4 LDX 2 IX1	3 LDX 1 IX	E 1110
F 1111	5 BRCLR7 3 DIR	5 BCLR7 2 DIR	3 BIH 2 REL	5 CLR 2 DIR	3 CLRA 1 INH	3 CLRX 1 INH	6 CLR 2 INH	5 CLR 1 IX	2 WAIT 1 INH	2 TXA 1 INH		4 STX 2 DIR	5 STX 3 EXT	6 STX 3 IX2	5 STX 2 IX1	4 STX 1 IX	F 1111

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

LEGEND

	F 1111	High Byte of Opcode in Hexadecimal High Byte of Opcode in Binary
Number of Cycles Opcode Mnemonic Number of Bytes/Addressing Mode	3 SUB 1 IX	0 0000	Low Byte of Opcode in Hexadecimal Low Byte of Opcode in Binary

SECTION 12
ELECTRICAL SPECIFICATIONS

This section contains parametric and timing information.

12.1 Maximum Ratings

The MCU contains circuitry that protects the inputs against damage from high static voltages; however, do not apply voltages higher than those shown in Table 12-1. Keep V_{IN} and V_{OUT} within the range $V_{SS} \le (V_{IN}$ or $V_{OUT}) \le V_{DD}$. Connect unused inputs to the appropriate logic level, either V_{SS} or V_{DD}.

Table 12-1. Maximum Ratings

Rating	Symbol	Value	Unit
Supply Voltage	V_{DD}	–0.3 to +7.0	V
Input Voltage Normal Operation ($\overline{IRQ}$ Pin Only) Self-Check Mode ($\overline{IRQ}$ Pin Only)	V_{IN}	 V_{SS} – 0.3 to V_{DD} + 0.3 V_{SS} – 0.3 to 2 × V_{DD} + 0.3	V
Current Drain Per Pin (Excluding V_{DD} and V_{SS})	I	25	mA
Operating Temperature Range MC68HC05F5P (Standard) MC68HC05F5FN (Standard) MC68HC05F5CP (Extended) MC68HC05F5CFN (Extended)	T_A	 0 to +70 0 to +70 –40 to +85 –40 to +85	°C
Storage Temperature Range	T_{STG}	–65 to +150	°C

NOTES:
1. P = Plastic dual in-line package (PDIP)
2. FN = Plastic-leaded chip carrier (PLCC)
3. C = Extended temperature range (–40 to +85 °C)

12.2 Thermal Characteristics

Table 12-2. Thermal Resistance

Characteristic	Symbol	Value	Unit
Maximum Junction Temperature	T_J	150	°C
Thermal Resistance Plastic DIP Plastic SOIC	θ_{JA}	 60 60	 °C/W °C/W

12.3 Power Considerations

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

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

where:

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

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

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

$$P_D = K \div (T_J + 273\ °C) \quad (2)$$

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

$$K = P_D \times (T_A + 273\ °C) + \theta_{JA} \times (P_D)^2 \quad (3)$$

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.

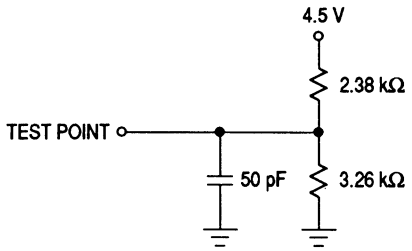


Figure 12-1. Equivalent Test Load

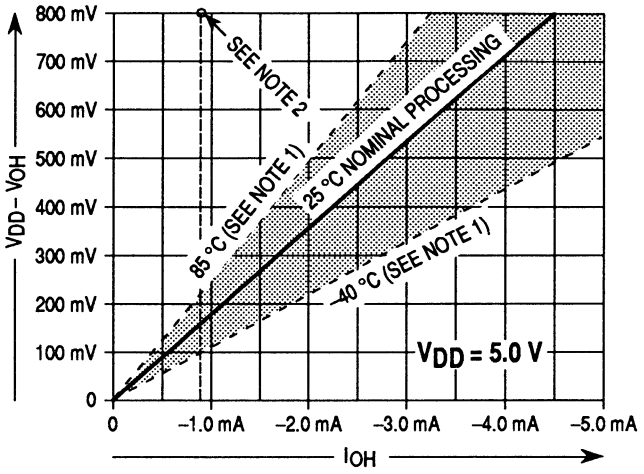
12.4 DC Electrical Characteristics (V_{DD} = 5.0 Vdc)

Table 12-3. DC Electrical Characteristics (V_{DD} = 5.0 Vdc)

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage Iload = 10.0 µA Iload = –10.0 µA	 V_{OL} V_{OH}	 — V_{DD} – 0.1	 — —	 0.1 —	 V V
Output High Voltage (I_{LOAD} = –0.8 mA) PA7–0, PB5–0, PC7–0, PD7–0	V_{OH}	V_{DD} – 0.8	—	—	V
Output Low Voltage (I_{LOAD} = 1.6 mA) PA7–0, PB5–0, PC7–0, PD7–0	V_{OL}	—	—	0.4	V
Input High Voltage PA7–0, PB5–0, PC7–0, PD7–0, $\overline{IRQ}$, $\overline{RESET}$, OSC1, IRQ2	V_{IH}	0.7 × V_{DD}	—	V_{DD}	V
Input Low Voltage PA7–0, PB5–0, PC7–0, PD7–0, $\overline{IRQ}$, $\overline{RESET}$, OSC1, IRQ2	V_{IL}	V_{SS}	—	0.2 × V_{DD}	V
Analog Input Voltage	V_{AIN}	—	—	V_{DD}	V
Analog Input Impedance	R_{IN}	90	100	—	kΩ
Supply Current (See NOTES.) Run Wait Stop 25 °C 0 to +70 °C (Standard) –40 to +85 °C (Extended)	 I_{DD}	 — — — —	 1.8 550 2 4 5	 TBD TBD TBD TBD TBD	 mA µA µA µA µA
I/O Ports High-Z Leakage Current PA7–0, PB5–0, PC7–0, PD7–0	I_{OZ}	—	—	±10	µA
Input Current $\overline{RESET}$, $\overline{IRQ}$, IRQ2, OSC1	I_{IN}	—	—	±1	µA
Capacitance Ports (Input or Output) $\overline{RESET}$, $\overline{IRQ}$, IRQ2, A_{IN}	 C_{OUT} C_{IN}	 — —	 — —	 12 8	pF
Analog Supply Current 25 °C –40 to +85 °C	AI_{DD}	 — —	 5 5.5	 — —	 mA mA

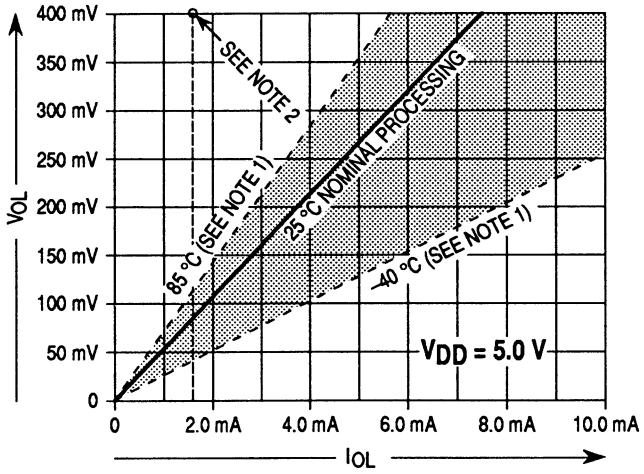
NOTES:

1. V_{DD} = 5.0 Vdc ±10%. V_{SS} = 0 Vdc.
2. All values reflect average measurements.
3. Typical values at midpoint of voltage range, 25 °C only.
4. Run (operating) I_{DD} and Wait I_{DD} measured using external square wave clock source (f_{OSC} = 4.2 MHz) with all inputs 0.2 V from rail; no dc loads; less than 50 pF on all outputs; C_L = 20 pF on OSC2.
5. Wait I_{DD} and Stop I_{DD}: All ports configured as inputs; V_{IL} = 0.2 V, V_{IH} = V_{DD} – 0.2 V.
6. Wait I_{DD} measured with only timer active.
7. Stop I_{DD} measured with OSC1 = V_{SS}.
8. OSC2 capacitance linearly affects Wait I_{DD} .



NOTES:
1. 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.
2. At V_{DD} = 5.0 V, devices are specified and tested for (V_{DD} – V_{OH}) ≤ 800 mV @ I_{OH} = –0.8 mA.

Figure 12-2. Typical High-Side Driver Characteristics



NOTES:
1. 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.
2. At V_{DD} = 5.0 V, devices are specified and tested for V_{OL} ≤ 400 mV @ I_{OL} = 1.6 mA.

Figure 12-3. Typical Low-Side Driver Characteristics

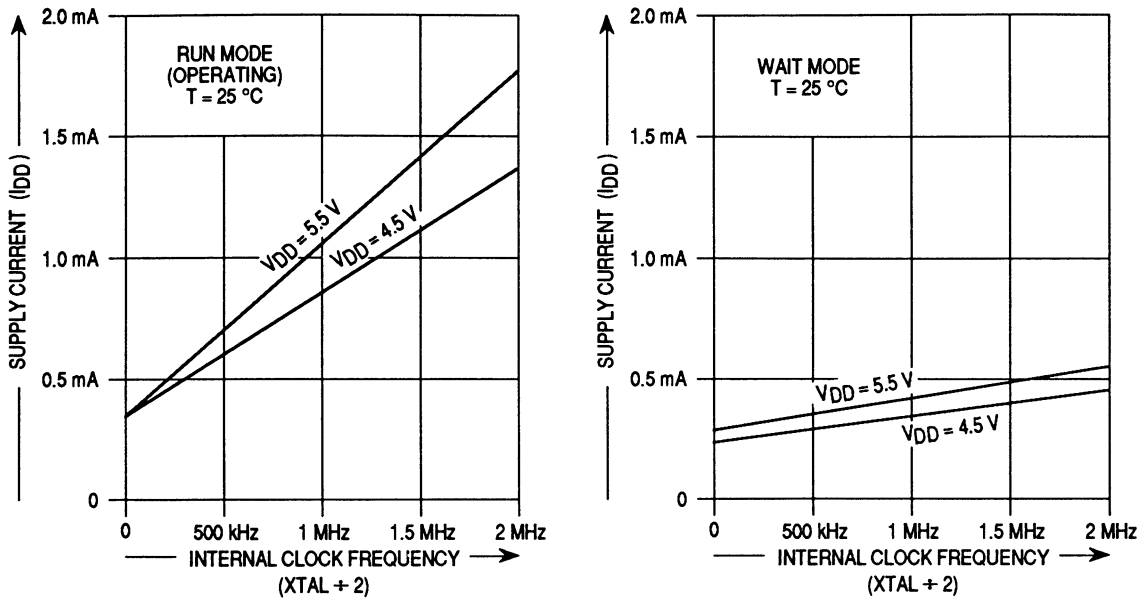


Figure 12-4. Typical Supply Current vs Clock Frequency

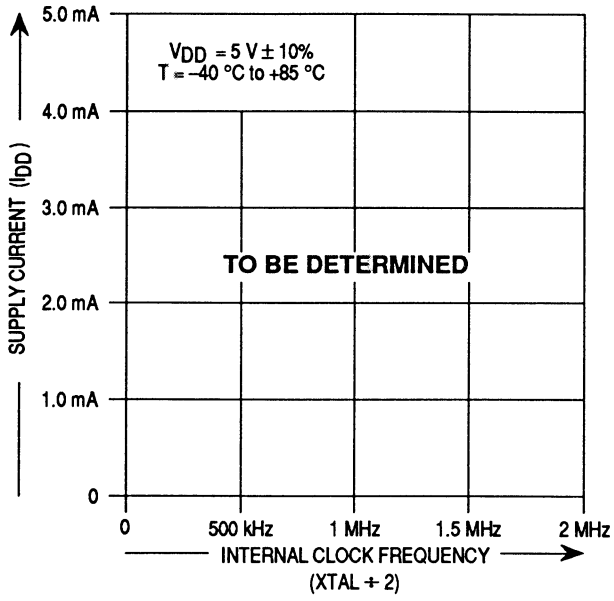


Figure 12-5. Maximum Supply Current vs Clock Frequency

12.5 Analog Characteristics

Table 12-4. Analog Characteristics

Characteristic	Min	Typ	Max	Unit
Signal Level for Detection, A_{IN} (NOTE 2)	–35	—	–2	dBm
Twist (HighTone/Low Tone)	–10	—	10	dB
Frequency Detect Bandwidth (NOTES 3, 4)	+(1.5 + 2 Hz)	±2.3	—	% of F_0
60 Hz Tolerance	—	—	0.8	V rms
Dial Tone Tolerance (Dial Tone 330+440; NOTE 5)	—	—	0	dB
Noise Tolerance (NOTES 5, 6, 7)	—	—	–12	dB
Power Supply Noise (Wide Band)	—	—	TBD	mV p-p
Talk Off (NOTE 7)	—	TBD	—	Hits

NOTES:
1. V_{DD} = 5.0 Vdc ± 10%; V_{SS} = 0 Vdc; T_A = –40 °C to +85 °C unless otherwise noted.
2. Z = 600 Ω.
3. F_0 is the center frequency of the bandpass filters.
4. Referenced to lower amplitude tone.
5. Bandwidth limited (0 to 3.4 kHz) Gaussian noise.
6. Using Mitel Tape #CM7290.

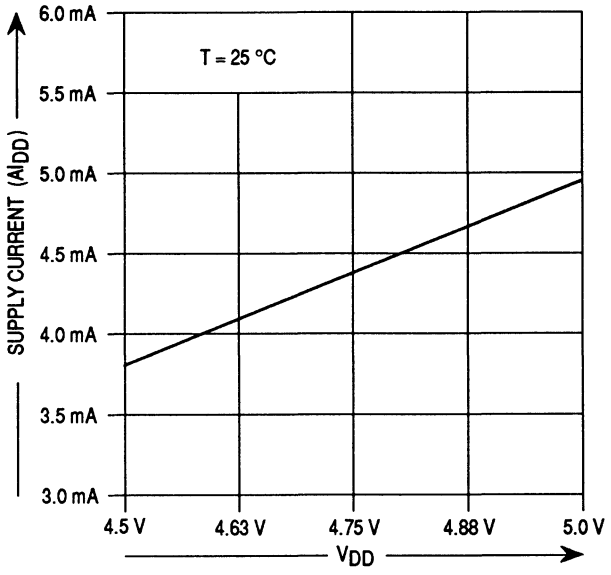


Figure 12-6. Typical Analog Supply Current

12.6 DTMF Receiver Timing

Table 12-5. DTMF Receiver Timing

Characteristic	Symbol	Min	Typ	Max	Unit
Tone On Time For Detection For Rejection	$Tone_{ON}$	 40 —	 — —	 — 20	 ms ms
Pause Time For Detection For Rejection	$Tone_{OFF}$	 40 —	 — —	 — 20	 ms ms
Detect Time	t_{DET}	28	—	48	ms
Reject Time	t_{REJ}	20	—	32	ms
Receiver Propagation Delay	t_{PROP}	—	TBD	—	ms

NOTE: V_{DD} = 5.0 Vdc ±10%; V_{SS} = 0 Vdc; T_A = –40 °C to +85 °C unless otherwise noted.

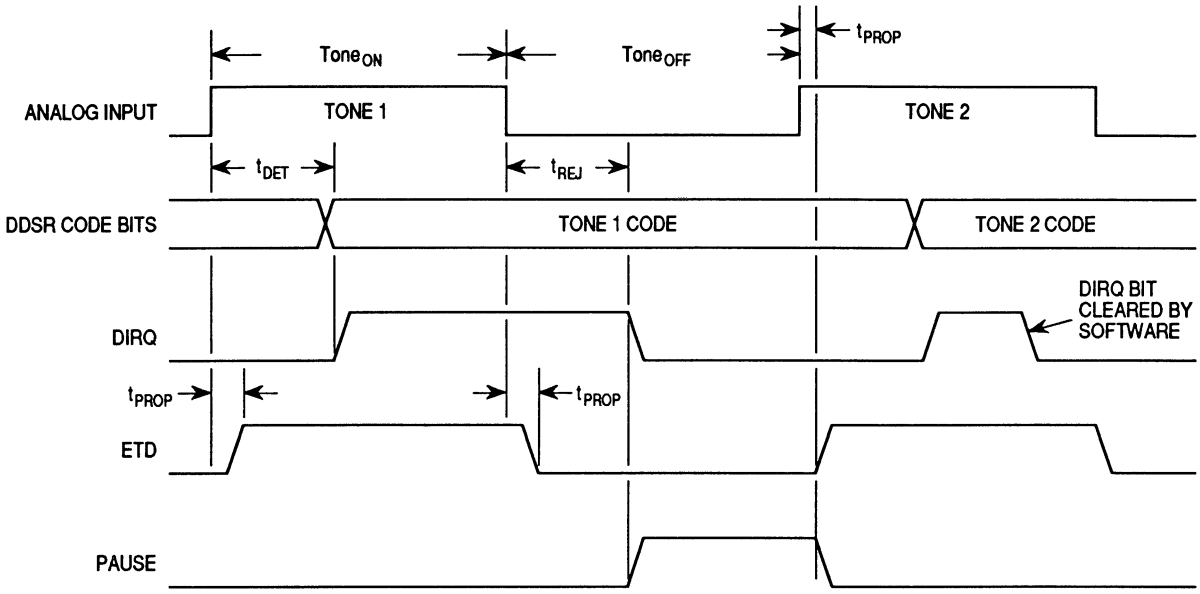


Figure 12-7. DTMF Receiver Timing

12.7 Control Timing (V_{DD} = 5.0 Vdc)

Table 12-6. Control Timing (V_{DD} = 5.0 Vdc)

Characteristic	Symbol	Min	Max	Unit
Internal Oscillator Frequency Crystal Option External Clock Option	 f_{OSC} f_{OSC}	 — dc	 4.0 4.0	 MHz MHz
Internal Operating Frequency Crystal Oscillator (f_{OSC} ÷ 2) External Clock (f_{OSC} ÷ 2)	 f_{OP} f_{OP}	 — dc	 2.0 2.0	 MHz MHz
Cycle Time (1 ÷ f_{OP})	t_{CYC}	480	—	ns
Crystal Oscillator Start-Up Time	t_{OXON}	—	100	ms
Stop Recovery Start-Up Time (Crystal Oscillator)	t_{ILCH}	—	100	ms
$\overline{\text{RESET}}$ Pulse Width Low	t_{RL}	1.5	—	t_{CYC}
Timer Resolution (NOTE 2)	t_{RESL}	4.0	—	t_{CYC}
Interrupt Pulse Width Low (Edge-Triggered)	t_{ILIH}	125	—	ns
Interrupt Pulse Period	t_{ILIL}	(NOTE 3)	—	t_{CYC}
OSC1 Pulse Width	t_{OH}, t_{OL}	90	—	ns
V_{DD} Slew Rate Rising Falling	t_{VDDR}	 — —	 0.05 0.1	 V/µs V/µs

NOTES:

1. V_{DD} = 5.0 Vdc ±10%, V_{SS} = 0 Vdc.
2. The 2-bit timer prescaler is the limiting factor in determining timer resolution.
3. The minimum period t_{ILIL} or t_{IHIH} should not be less than the number of cycles it takes to execute the interrupt service routine plus 19 t_{CYC}.

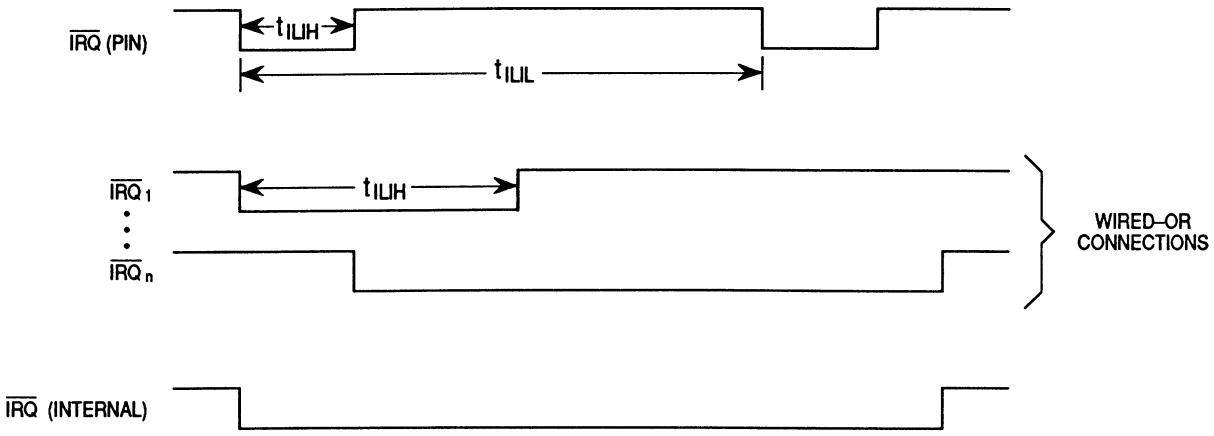
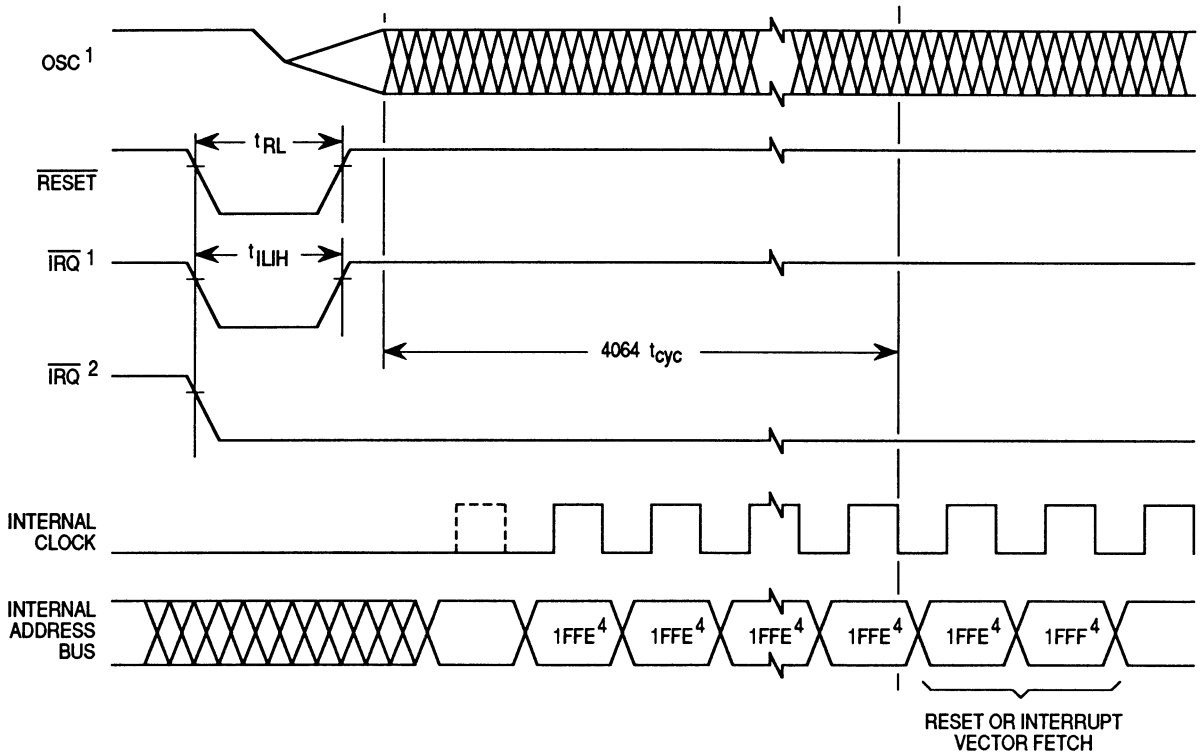
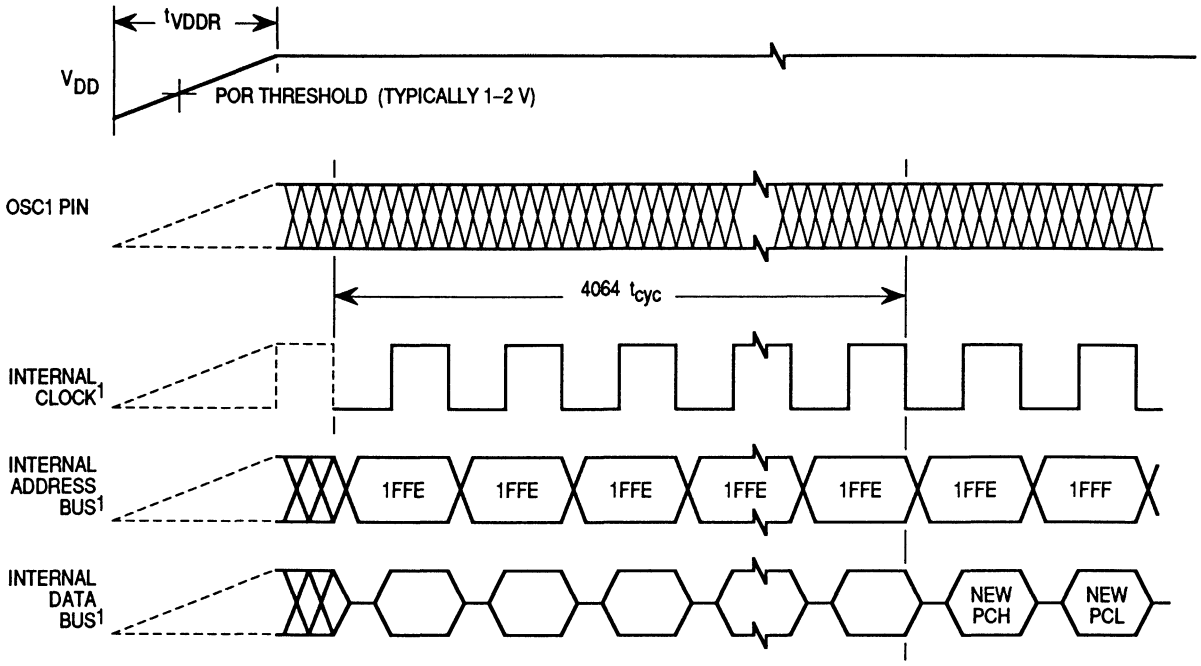


Figure 12-8. External Interrupt Timing



NOTES:
1. Represents the internal clocking of OSC1 pin
2. External interrupt edge-triggered mask option
3. External interrupt edge- and level-triggered mask option
4. Reset vector shown for timing example

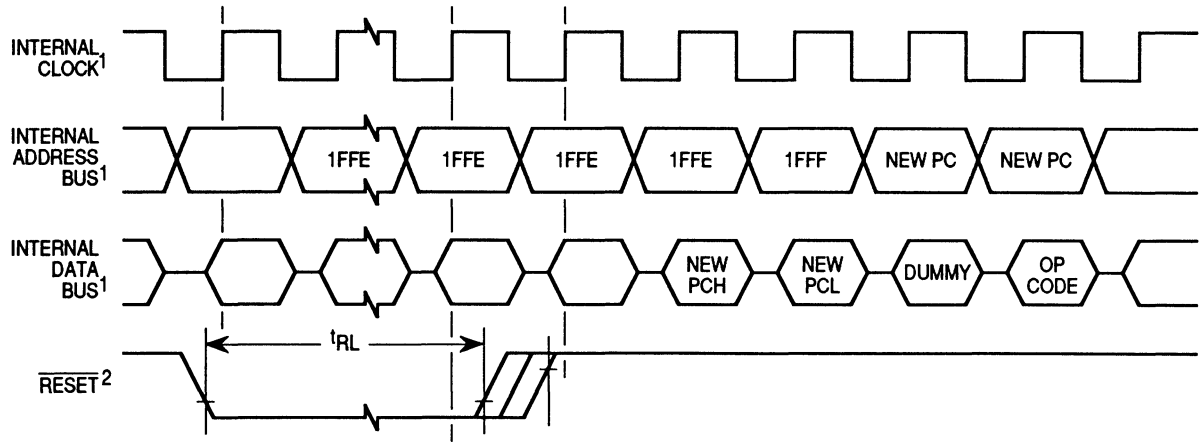
Figure 12-9. STOP Recovery Timing



NOTES:

1. Internal clock, internal address bus, and internal data bus are not available externally.

Figure 12-10. Power-On Reset Timing



NOTES:

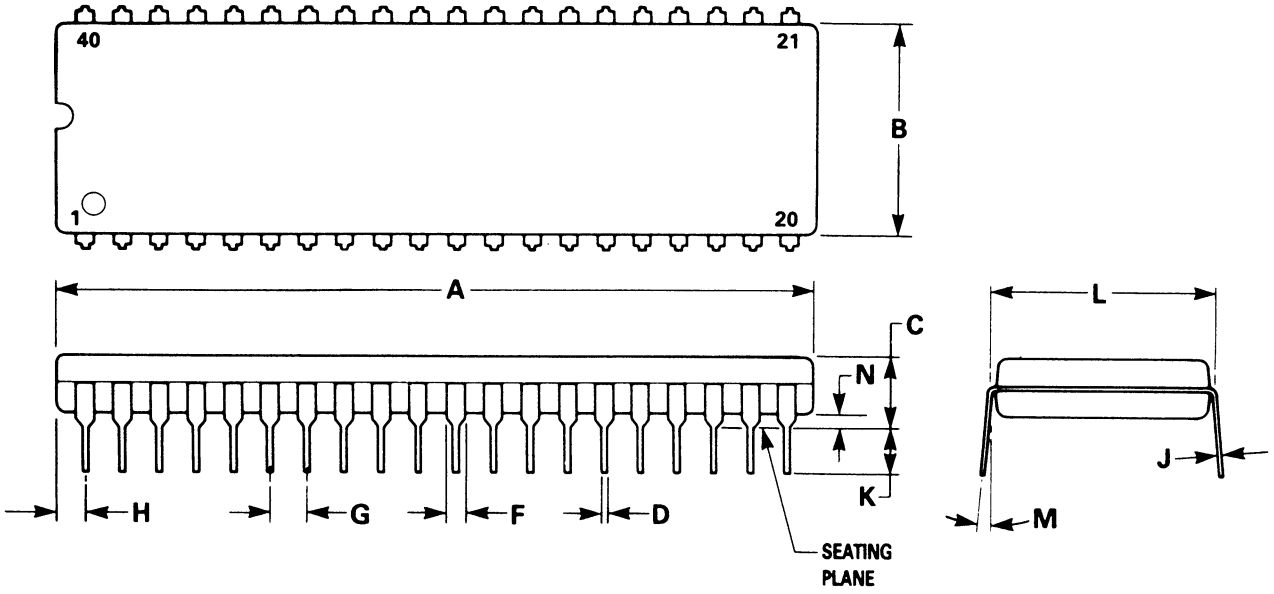
1. Internal clock, internal address bus, and internal data bus signals are not available externally.
2. Next rising edge of internal clock after rising edge of $\overline{\text{RESET}}$ initiates reset sequence.

Figure 12-11. External Reset Timing

SECTION 13
MECHANICAL SPECIFICATIONS

This section gives the dimensions of the plastic dual in-line package (PDIP) and plastic-leaded chip carrier (PLCC) packages.

13.1 PDIP

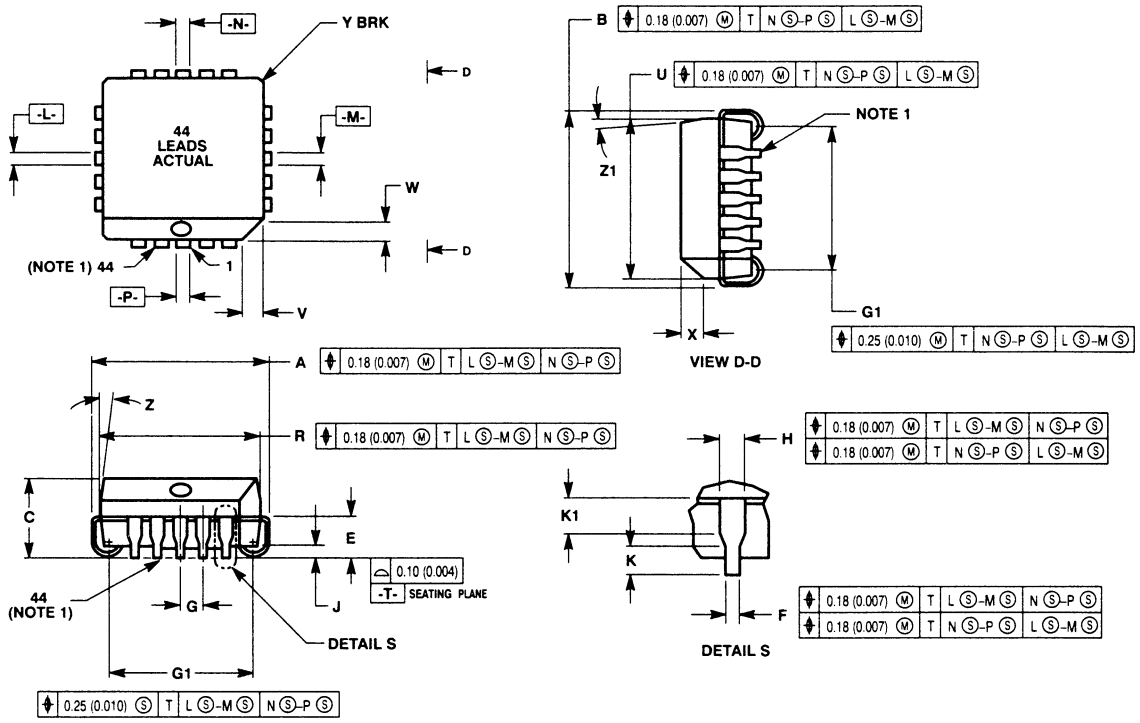


DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	51.69	52.45	2.035	2.065
B	13.72	14.22	0.540	0.560
C	3.94	5.08	0.155	0.200
D	0.36	0.56	0.014	0.022
F	1.02	1.52	0.040	0.060
G	2.54 BSC		0.100 BSC	
H	1.65	2.16	0.065	0.085
J	0.20	0.38	0.008	0.015
K	2.92	3.43	0.115	0.135
L	15.24 BSC		0.600 BSC	
M	0°	15°	0°	15°
N	0.51	1.02	0.020	0.040

NOTES:
1. POSITIONAL TOLERANCE OF LEADS (D), SHALL BE WITHIN 0.25 mm (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.

Figure 13-1. MC68HC05F5P (Case 711-03)

13.2 PLCC



DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	17.40	17.65	0.685	0.695
B	17.40	17.65	0.685	0.695
C	4.20	4.57	0.165	0.180
E	2.29	2.79	0.090	0.110
F	0.33	0.48	0.013	0.019
G	1.27 BSC		0.050 BSC	
H	0.66	0.81	0.026	0.032
J	0.51	—	0.020	—
K	0.64	—	0.025	—
R	16.51	16.66	0.650	0.656
U	16.51	16.66	0.650	0.656
V	1.07	1.21	0.042	0.048
W	1.07	1.21	0.042	0.048
X	1.07	1.42	0.042	0.056
Y	—	0.50	—	0.020
Z	2°	10°	2°	10°
G1	15.50	16.00	0.610	0.630
K1	1.02	—	0.040	—
Z1	2°	10°	2°	10°

NOTES:
1. **DUE TO SPACE LIMITATION, CASE 777-02 SHALL BE REPRESENTED BY A GENERAL (SMALLER) CASE OUTLINE DRAWING RATHER THAN SHOWING ALL 44 LEADS.**
2. DATUMS -L-, -M-, -N-, AND -P- DETERMINED WHERE TOP OF LEAD SHOULDER EXIT PLASTIC BODY AT MOLD PARTING LINE.
3. DIM G1, TRUE POSITION TO BE MEASURED AT DATUM -T-, SEATING PLANE.
4. DIM R AND U DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE MOLD PROTRUSION IS 0.25 (0.010) PER SIDE.
5. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
6. CONTROLLING DIMENSION: INCH.

Figure 13-2. MC68HC05F5FN (Case 777-02)

SECTION 14
ORDERING INFORMATION

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

14.1 MCU Ordering Forms

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

- A current MCU ordering form that is **completely filled out** (Contact your Motorola sales office for assistance.)
- A signed XC status letter, if applicable (See **14.5 XC Status Letter**.)
- A copy of the customer specification if the customer specification deviates from the Motorola specification for the MCU
- Customer's application program on one of the media listed in **14.2 Application Program Media**

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

14.2 Application Program Media

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

- Macintosh®[1] 3-1/2-inch diskette (double-sided 800K or double-sided high-density 1.4M)
- MS-DOS®[2] or PC-DOS®[3] 3-1/2-inch diskette (double-sided 720K or double-sided high-density 1.44M)
- MS-DOS® or PC-DOS® 5-1/4-inch diskette (double-sided double-density 360K or double-sided high-density 1.2M)
- EPROM(s) 2716, 2732, 2764, 27128, 27256, or 27512 (depending on the size of the memory map of the MCU)

Use positive logic for data and addresses.

14.2.1 Diskettes

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

- Customer name
- Customer part number
- Project or product name
- Filename of object code
- Date
- Name of operating system that formatted diskette
- Formatted capacity of diskette

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

---

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

[2] MS-DOS is a registered trademark of Microsoft, Inc.

[3] PC-DOS is a registered trademark of International Business Machines Corporation.

NOTE

Begin the application program at the first user ROM location. Program addresses must correspond exactly to the available on-chip user ROM addresses as shown in the memory map. **Write $00 in all non-user ROM locations or leave all non-user ROM locations blank.** See the current MCU ordering form for additional requirements.

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

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

14.2.2 EPROMs

If submitting the application program in an EPROM, clearly label the EPROM with the following information:

- Customer name
- Customer part number
- Checksum
- Project or product name
- Date

NOTE

Begin the application program at the first user ROM location. Program addresses must correspond exactly to the available on-chip user ROM addresses as shown in the memory map. **Write $00 in all non-user ROM locations.** See the current MCU ordering form for additional requirements.

Submit the application program in one EPROM large enough to contain the entire memory map. If the memory map has two user ROM areas with the same addresses, then write the two areas on separate EPROMs. Label the EPROMs with the addresses they contain.

Pack EPROMs securely in a conductive IC carrier for shipment. Do not use Styrofoam.

14.3 ROM Program Verification

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

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

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

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

14.4 ROM Verification Units (RVUs)

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

14.5 XC Status Letter

The XC status letter is for customer acknowledgement that the MCU is a pilot production device. As a pilot production device, the MCU part number has an XC prefix. When the MCU meets all of Motorola's formal quality and reliability requirements, the XC prefix is replaced with MC. The MCU ordering form indicates whether or not the MCU order requires an XC status letter.

Date:

To:

Subject: XC Status of Device

This letter requests formal authorization from _____ for Motorola Microcontroller Division to ship XC _____ devices as pilot production MCUs.

The XC prefix indicates that the MCU has yet to meet Motorola's formal quality and reliability requirements and is still in the pilot production phase. The pilot production phase lasts approximately six months as the necessary qualification and stress tests are completed to bring the MCU to fully qualified MC status. The manufacture of XC MCUs is similar to the manufacture of standard production MCUs and includes the following:

- Processing per production shop order
- 100% testing per current data sheet
- Standard QA inspection and tests
- Complete traceability
- Preliminary reliability testing

This letter requests that a representative of _____ acknowledge by signing below that he understands the XC pilot production status and that he will receive XC _____ pilot production MCUs.

_____

John H. Sayce
Reliability and Quality Assurance Manager
Advanced Microcontroller Division/CSIC Microcontroller Division

Customer representative: Please sign below and return this letter to the following address:

Motorola CSIC Microcontroller Division
6501 William Cannon Drive West
Austin, TX 78735
Mail Drop OE 39

_____ _____

Customer Representative Title

_____

Date

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.

Freescale Semiconductor, Inc.