HC05

MC68HC05C9

TECHNICAL
DATA

MC68HC05C9

HCMOS MICROCONTROLLER UNIT

Freescale Semiconductor, Inc.

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters can and do vary in different applications. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part. Motorola and Ⓜ are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

© MOTOROLA INC., 1992

TABLE OF CONTENTS

Section **Title** **Page**

SECTION 1
GENERAL DESCRIPTION

SECTION 2
MEMORY

TABLE OF CONTENTS
(Continued)

Section **Title** **Page**

SECTION 3
CENTRAL PROCESSOR UNIT

SECTION 4
INTERRUPTS

SECTION 5
RESETS

TABLE OF CONTENTS
(Continued)

SECTION 6
LOW POWER MODES

SECTION 7
PARALLEL I/O

TABLE OF CONTENTS (Continued)

Section **Title** **Page**

SECTION 8
CAPTURE/COMPARE TIMER

SECTION 9
SERIAL COMMUNICATIONS INTERFACE

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

Section Title Page

SECTION 12
INSTRUCTION SET

SECTION 13
ELECTRICAL SPECIFICATIONS

SECTION 14
MECHANICAL SPECIFICATIONS

TABLE OF CONTENTS (Concluded)

Section **Title** **Page**

SECTION 15 ORDERING INFORMATION

LIST OF FIGURES

LIST OF FIGURES
(Continued)

LIST OF FIGURES (Concluded)

LIST OF TABLES

LIST OF TABLES
(Concluded)

SECTION 1
GENERAL DESCRIPTION

The MC68HC05C9 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 MC68HC05C9 includes 15936 bytes of user ROM, 240 bytes of self-check ROM, and 352 bytes of RAM.

1.1 Features

Features of the MCU include the following:

- Popular HC05 CPU
- 15936 Bytes of User ROM
- 352 Bytes of RAM
- 240 Bytes of Self-Check ROM
- Memory Mapped Input/Output (I/O)
- 31 Bidirectional I/O Lines
- Selectable Memory Configurations
- Fully Static Operation (No Minimum Clock Speed)
- On-Chip Oscillator with Crystal/Ceramic Resonator Connections
- 16-Bit Capture/Compare Timer
- Asynchronous Serial Communications Interface (SCI)
- Synchronous Serial Peripheral Interface (SPI)
- Power Saving Stop, Wait, and Data-Retention Modes
- Single 3.0- to 5.5-Volt Power Supply Requirement
- Computer Operating Properly (COP) Watchdog Timer
- Clock Monitor
- Software-Programmable External Interrupt Sensitivity
- 40-Pin Plastic Dual-In-Line Package (PDIP)
- 44-Lead Plastic-Leaded Chip Carrier (PLCC)
- 44-Pin Quad Flat Pack (QFP)
- 42-Pin Plastic Shrink DIP (SDIP)

1.2 Mask Option

The following MC68HC05C9 mask option is available:

- Crystal/ceramic resonator oscillator
  (default is crystal)

1.3 Programmable Options

The option register (OR) shown in Figure 1-1 contains programmable bits for the following options:

- Edge-triggered only or edge- and level-triggered external interrupt pin ($\overline{\text{IRQ}}$ pin)
- Increase of RAM by 48 and/or 128 bytes

OR — Option Register **$3FDF**

	Bit 7	6	5	4	3	2	1	Bit 0
	RAM0	RAM1	0	0	0	0	IRQ	0
RESET:	0	0	0	0	0	0	1	0

Figure 1-1. Option Register (OR)

RAM0 — Random Access Memory Control Bit 0

1 = Maps 48 bytes of RAM into page zero starting at address $0020. This replaces 48 bytes of ROM. RAM0 is readable and writable at all times, allowing memory configuration to be changed during program execution.

RAM1 — Random Access Memory Control Bit 1

1 = Maps 128 bytes of RAM into page zero starting at address $0100. This replaces 128 bytes of ROM. RAM1 is readable and writable at all times, allowing memory configuration to be changed during program execution.

IRQ — Interrupt Request

1 = Edge and level interrupt option selected

0 = Edge-only interrupt option selected

Bits 5–2 and 0 — Not used; always read zero

1.4 Block Diagram

Figure 1-2 shows the structure of the MC68HC05C9 MCU.

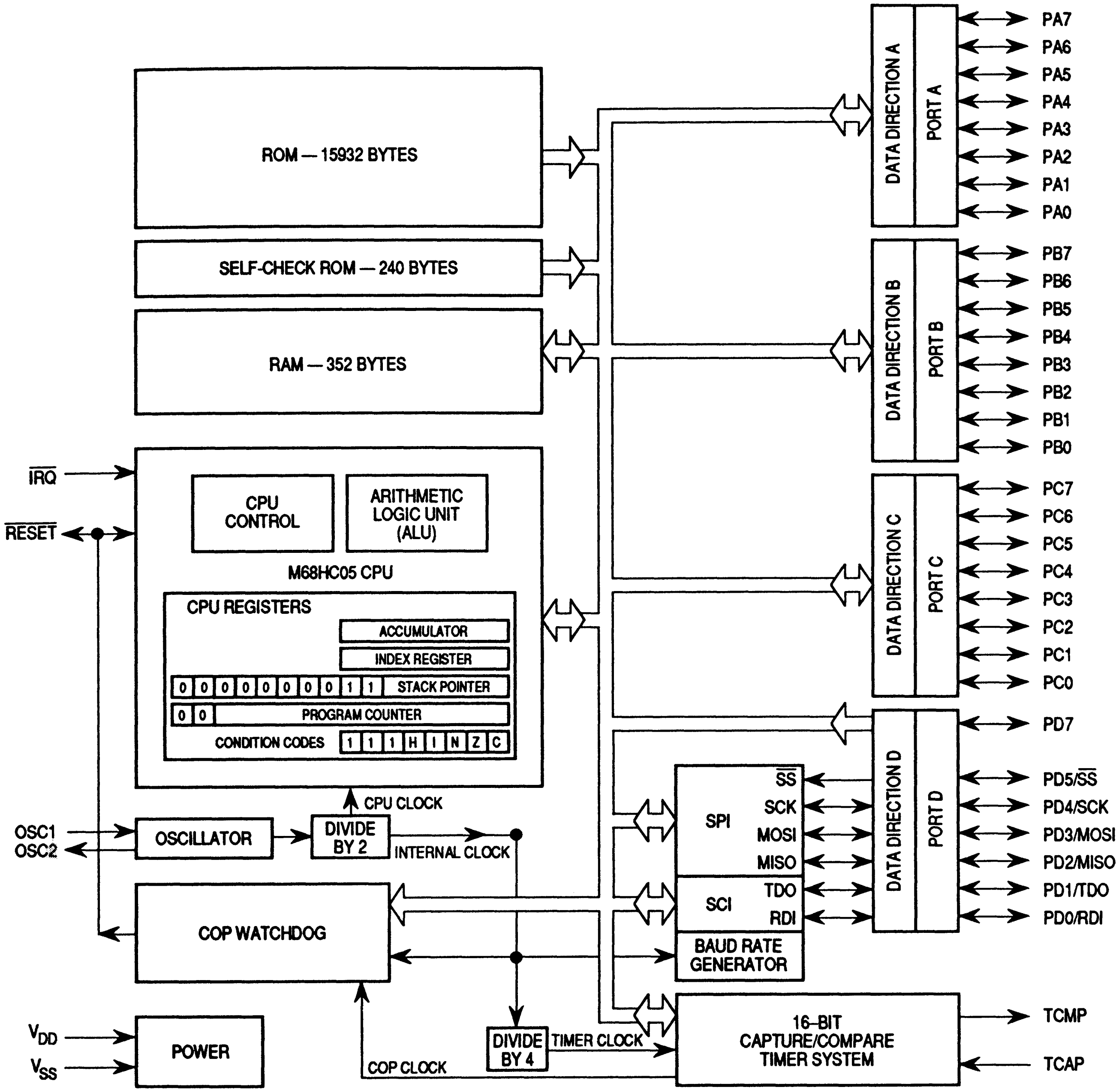
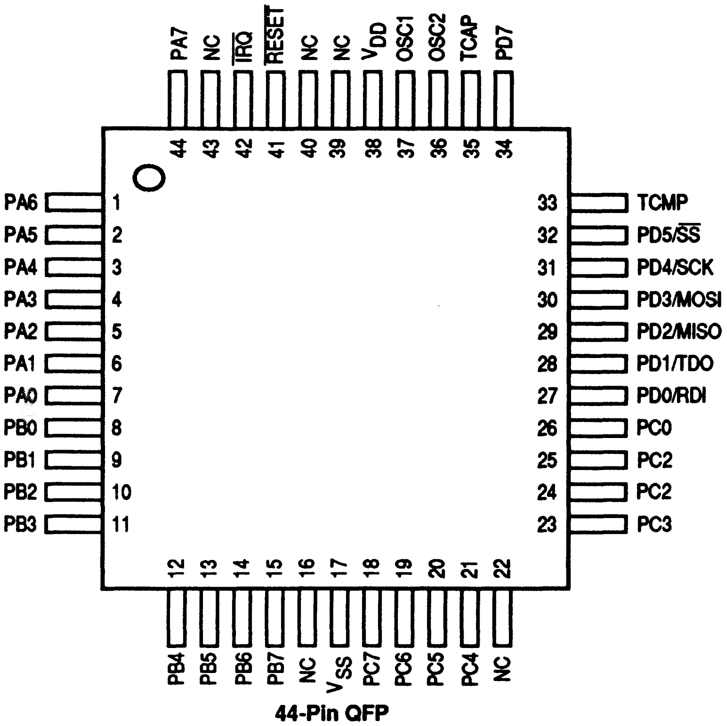


Figure 1-2. MC68HC05C9 Block Diagram

1.5 Pin Assignments

Figure 1-3 shows the pin assignments.



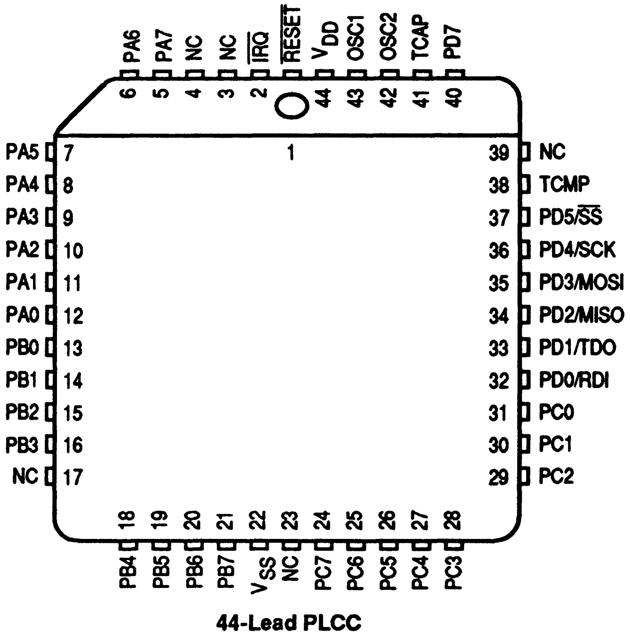
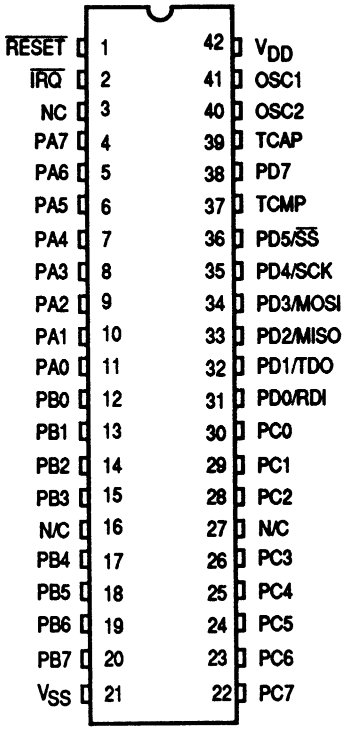
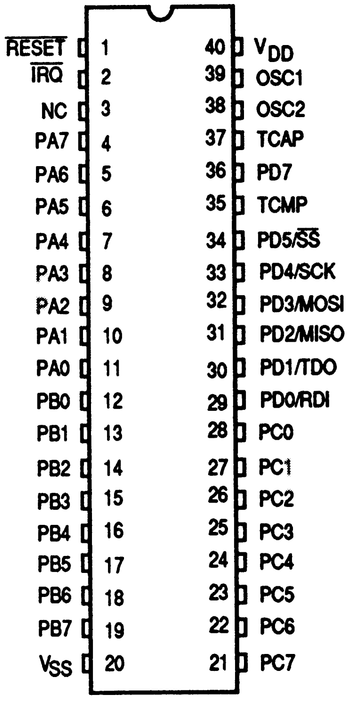


Figure 1-3. Pin Assignments (Page 1 of 2)



42-Pin SDIP



40-Pin PDIP

Figure 1-3. Pin Assignments (Page 2 of 2)

1.5.1 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. The short rise and fall times place 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. Use bypass capacitors with good high-frequency characteristics, and position them as close to the MCU as possible. Bypassing requirements vary, depending on how heavily loaded the MCU pins are.

1.5.2 OSC1 and OSC2

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

- Oscillator driven by a crystal or ceramic resonator (See Figure 1-4.)

1.5.2.1 Crystal/Ceramic Resonator

The circuit in Figure 1-4 shows a typical crystal oscillator circuit for a parallel resonant crystal. Follow the crystal supplier's recommendations, as the crystal parameters determine the external component values required to provide maximum stability and reliable start-up. The load capacitance values used in the oscillator circuit design should include all stray layout capacitances. Mount the crystal and components as close as possible to the pins for start-up stabilization and to minimize output distortion.

In cost-sensitive applications, use a ceramic resonator in place of the crystal. Use the circuit in Figure 1-4 for a ceramic resonator, and follow the resonator manufacturer's recommendations, as the resonator parameters determine the external component values required for maximum stability and reliable starting. The load capacitance values used in the oscillator circuit design should include all stray layout capacitances.

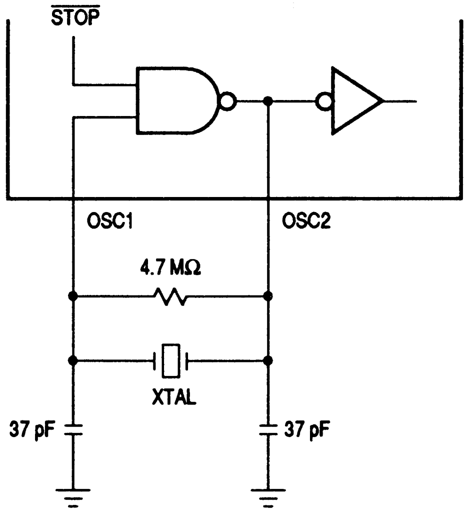


Figure 1-4. Crystal/Ceramic Resonator Connections

The crystal/ceramic resonator mask option also allows for an external clock from another CMOS-compatible device to drive the OSC1 input, with the OSC2 pin not connected, as Figure 1-5 shows.

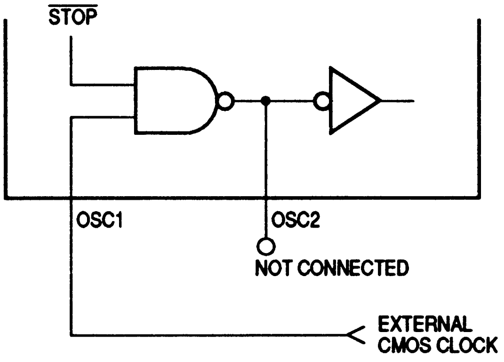


Figure 1-5. External Clock Connections

1.5.3 RESET

A logic zero on the RESET pin forces the MCU to a known start-up state. As an output, the RESET pin indicates that an internal MCU failure has been detected. (Refer to **SECTION 5 RESETS** for more information.)

1.5.4 IRQ (External Interrupt Request)

The IRQ pin has the following functions:

- Applying asynchronous external interrupt signals (Refer to **4.1.2 External Interrupt**.)
- Putting the MCU in self-check mode (Refer to **11.1 Self Check**.)

1.5.5 TCAP (Timer Capture)

This pin controls the input capture feature for the on-chip capture/compare timer. (Refer to **SECTION 8 CAPTURE/COMPARE TIMER**.)

1.5.6 TCMP (Timer Compare)

This pin is the output for the output compare feature of the on-chip capture/compare timer. (Refer to **SECTION 8 CAPTURE/COMPARE TIMER**.)

1.5.7 Input/Output Ports (PA7–PA0, PB7–PB0, PC7–PC0; PD7, PD5–PD0)

Thirty-one I/O lines are arranged into three 8-bit ports (A, B, and C) and one 7-bit special-function port (port D). These I/O lines are programmable as either inputs or outputs under software control of the data direction registers. (Refer to **SECTION 7 PARALLEL I/O** for a description of ports A–D.)

PD7 is the only programmable I/O pin in port D independent of the other subsystems (timer, SCI, and SPI).

1.5.8 Port D

Port D shares its pins with the timer, SCI, and SPI subsystems. Port D pin functions are defined in the following paragraphs. (Refer also to **SECTION 7 PARALLEL I/O.**)

1.5.8.1 PD5/$\overline{\text{SS}}$ (Slave Select)

PD5 is an SPI input when the SPI is enabled.

PD5 is a general-purpose I/O pin when the SPI is disabled.

1.5.8.2 PD4/SCK (Serial Clock)

PD4 is an input when the SPI is enabled in slave mode. PD4 is an output when the SPI is enabled in master mode, except when overridden by bit 4 of port D data direction register. When bit 4 = 0, the pin is forced into a high-impedance state.

PD4 is a general-purpose I/O pin when the SPI is disabled.

1.5.8.3 PD3/MOSI (Master Out, Slave In)

PD3 is an input when the SPI is enabled in slave mode. PD3 is an output when the SPI is enabled in master mode, except when overridden by bit 3 of port D data direction register. When bit 3 = 0, PD3 is forced into high-impedance state.

PD3 is a general-purpose I/O pin when the SPI is disabled.

1.5.8.4 PD2/MISO (Master In, Slave Out)

PD2 is an input when the SPI is enabled in master mode. PD3 is an output when the SPI is enabled in slave mode, except when bit 2 = 0 of port D data direction register, in which case PD2 is forced into a high-impedance state.

PD2 is a general-purpose I/O pin when the SPI is disabled.

1.5.8.5 PD1/TDO (Transmit Data Output)

PD1 is the transmit data output when the SCI is enabled.

PD1 is a general-purpose I/O pin when the SCI is disabled.

1.5.8.6 PD0/RDI (Receive Data Input)

When the SCI is enabled, PD1 is the high-impedance receive data input, and the SCI receiver is active.

PD1 is a general-purpose I/O pin when the SCI is disabled.

SECTION 2
MEMORY

Section 2 describes the organization of the on-chip memory.

2.1 Memory Map

The CPU can address 16 Kbytes of memory space. The ROM portion of memory holds the program instructions, fixed data, user-defined vectors, and 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. Refer to Figure 2-2 for a more detailed memory map of the 32-byte I/O register section.

2.2 Input/Output Section

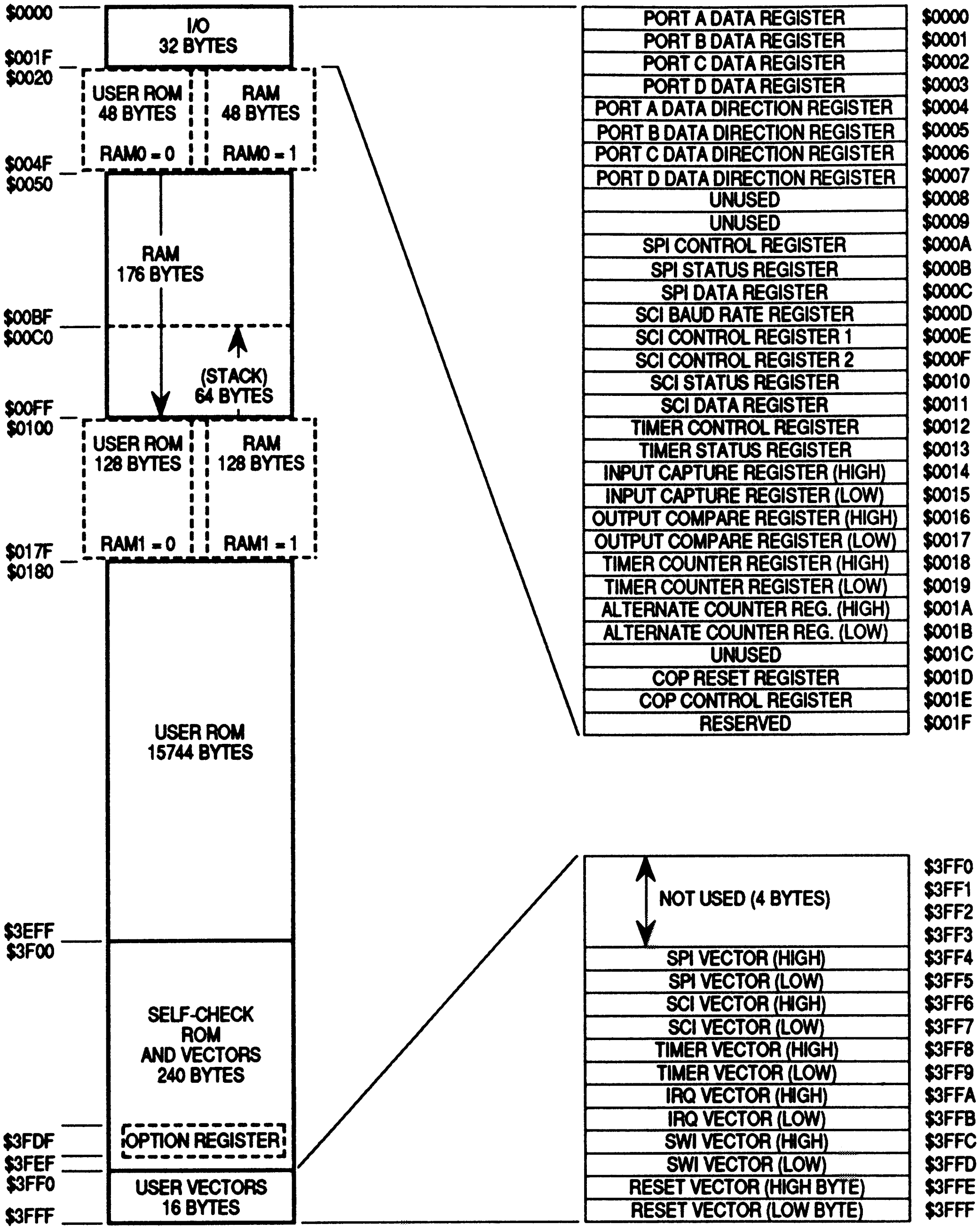
The I/O section resides in the first 32 addresses of the memory space ($0000–$001F) and contains the I/O control and status registers and the data registers.

2.3 RAM

The MCU has 176 bytes of fully static read/write memory for storage of variable and temporary data during program execution. RAM addresses $00C0–$00FF serve as the stack. The CPU uses the stack to save CPU register contents before processing an interrupt or subroutine call. The stack pointer decrements during pushes and increments during pulls.

If RAM0 in the option register (OR) is set, 48 bytes of RAM replace the 48 ROM bytes at addresses $0020–$004F. The corresponding 48 bytes of user ROM become inaccessible. (Refer to **1.3 Programmable Options**.)

If RAM1 in the option register is set, 128 bytes of RAM replace the 128 ROM bytes at addresses $0100–$017F. The corresponding 128 bytes of user ROM become inaccessible. (Refer to **1.3 Programmable Options**.)



PORT A DATA REGISTER	$0000
PORT B DATA REGISTER	$0001
PORT C DATA REGISTER	$0002
PORT D DATA REGISTER	$0003
PORT A DATA DIRECTION REGISTER	$0004
PORT B DATA DIRECTION REGISTER	$0005
PORT C DATA DIRECTION REGISTER	$0006
PORT D DATA DIRECTION REGISTER	$0007
UNUSED	$0008
UNUSED	$0009
SPI CONTROL REGISTER	$000A
SPI STATUS REGISTER	$000B
SPI DATA REGISTER	$000C
SCI BAUD RATE REGISTER	$000D
SCI CONTROL REGISTER 1	$000E
SCI CONTROL REGISTER 2	$000F
SCI STATUS REGISTER	$0010
SCI DATA REGISTER	$0011
TIMER CONTROL REGISTER	$0012
TIMER STATUS REGISTER	$0013
INPUT CAPTURE REGISTER (HIGH)	$0014
INPUT CAPTURE REGISTER (LOW)	$0015
OUTPUT COMPARE REGISTER (HIGH)	$0016
OUTPUT COMPARE REGISTER (LOW)	$0017
TIMER COUNTER REGISTER (HIGH)	$0018
TIMER COUNTER REGISTER (LOW)	$0019
ALTERNATE COUNTER REG. (HIGH)	$001A
ALTERNATE COUNTER REG. (LOW)	$001B
UNUSED	$001C
COP RESET REGISTER	$001D
COP CONTROL REGISTER	$001E
RESERVED	$001F

NOT USED (4 BYTES)	$3FF0 $3FF1 $3FF2 $3FF3
SPI VECTOR (HIGH)	$3FF4
SPI VECTOR (LOW)	$3FF5
SCI VECTOR (HIGH)	$3FF6
SCI VECTOR (LOW)	$3FF7
TIMER VECTOR (HIGH)	$3FF8
TIMER VECTOR (LOW)	$3FF9
IRQ VECTOR (HIGH)	$3FFA
IRQ VECTOR (LOW)	$3FFB
SWI VECTOR (HIGH)	$3FFC
SWI VECTOR (LOW)	$3FFD
RESET VECTOR (HIGH BYTE)	$3FFE
RESET VECTOR (LOW BYTE)	$3FFF

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	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0	PORTB
$0002	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0	PORTC
$0003	PD7	—	PD5	PD4	PD3	PD2	PD1	PD0	PORTD
$0004	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0	DDRA
$0005	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0	DDRB
$0006	DDRC7	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0	DDRC
$0007	DDRD7	—	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0	DDRD
$0008	—	—	—	—	—	—	—	—	UNUSED
$0009	—	—	—	—	—	—	—	—	UNUSED
$000A	SPIE	SPE	DWOM	MSTR	CPOL	CPHA	SPR1	SPR0	SPCR
$000B	SPIF	WCOL	—	MODF	—	—	—	—	SPSR
$000C	SPD7	SPD6	SPD5	SPD4	SPD3	SPD2	SPD1	SPD0	SPDR
$000D	—	—	SCP1	SCP0	—	SCR2	SCR1	SCR0	BAUD
$000E	R8	T8	—	M	WAKE	—	—	—	SCCR1
$000F	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK	SCCR2
$0010	TDRE	TC	RDRF	IDLE	OR	NF	FE	—	SCSR
$0011	SCD7	SCD6	SCD5	SCD4	SCD3	SCD2	SCD1	SCD0	SCDR
$0012	ICIE	OCIE	TOIE	—	—	—	IEDG	OLVL	TCR
$0013	ICF	OCF	TOF	—	—	—	—	—	TSR
$0014	Bit 15	14	13	12	11	10	9	Bit 8	ICRH
$0015	Bit 7	6	5	4	3	2	1	Bit 0	ICRL
$0016	Bit 15	14	13	12	11	10	9	Bit 8	OCRH
$0017	Bit 7	6	5	4	3	2	1	Bit 0	OCRL
$0018	Bit 15	14	13	12	11	10	9	Bit 8	TRH
$0019	Bit 7	6	5	4	3	2	1	Bit 0	TRL
$001A	Bit 15	14	13	12	11	10	9	Bit 8	ATRH
$001B	Bit 7	6	5	4	3	2	1	Bit 0	ATRL
$001C	—	—	—	—	—	—	—	—	UNUSED
$001D	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	COPRST
$001E	—	—	—	COPF	CME	COPE	CM1	CM0	COPCR
$001F	—	—	—	—	—	—	—	—	RESERVED
$3FDF	RAM0	RAM1	0	0	0	0	IRQ	0	OR

Figure 2-2. I/O Registers and Control Bits

NOTE

Be careful if using the stack addresses ($00C0–$00FF) for data storage or as a temporary work area. The CPU may overwrite data in the stack during a subroutine or interrupt.

2.4 ROM

On-chip user ROM includes 15872 bytes at addresses $0100–$3EFF, 240 bytes at $3F00–$3FEF (containing instructions for a series of self-check tests), and 16 bytes at $3FF0–$3FFF (containing user-defined vectors for servicing interrupts and resets). This main array totals 16128 bytes.

A second ROM array of 48 bytes exists on page zero at addresses $0020–$004F.

SECTION 3
CENTRAL PROCESSOR UNIT

This section describes the CPU registers.

3.1 CPU Registers

Figure 3-1 shows the five CPU registers. These are hard-wired registers within the CPU and are not part of the memory map.

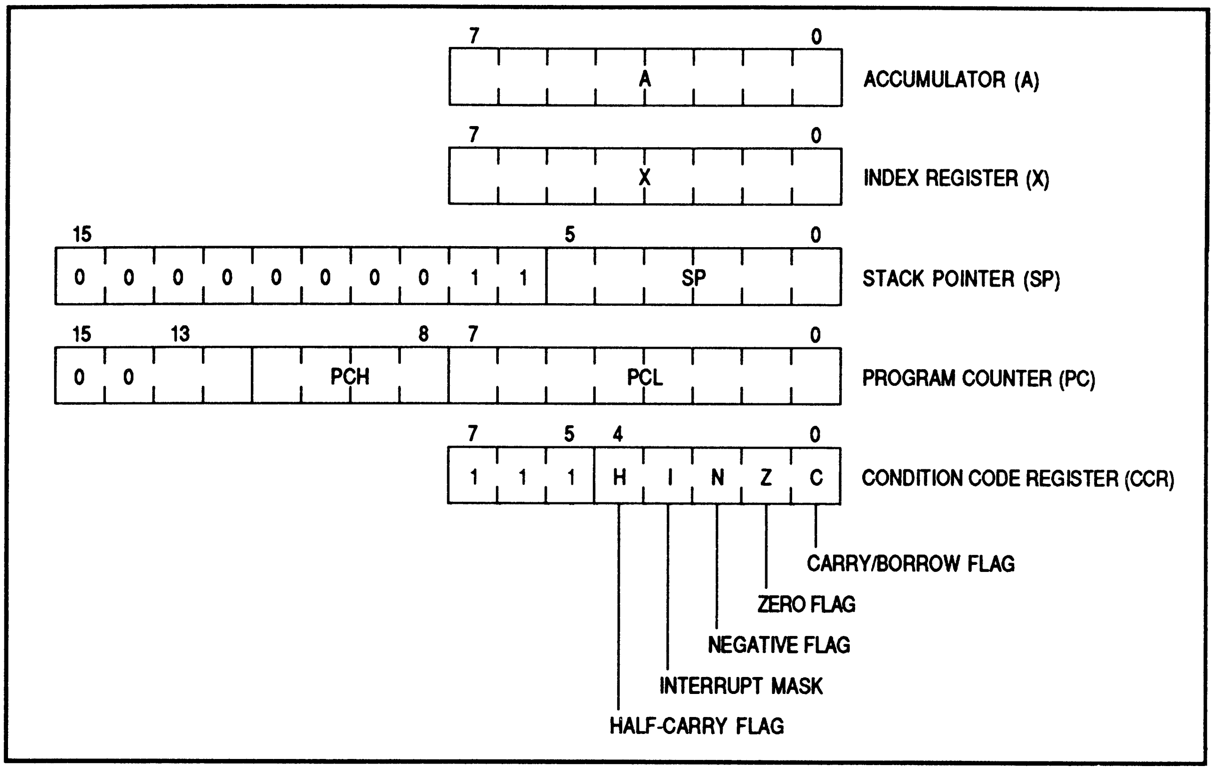


Figure 3-1. CPU Programming Model

3.1.1 Accumulator

The accumulator shown in Figure 3-2 is a general-purpose 8-bit register. The CPU uses the accumulator to hold operands and 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

The 8-bit index register shown in Figure 3-3 can perform two functions:

- Indexed addressing
- Temporary storage

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

Figure 3-3. Index Register

The index register contains an 8-bit value that can be added to a 0-, 8-, or 16-bit immediate value to create a conditional address. The index register can also serve as an auxiliary accumulator for temporary storage. (Refer to **12.1 Addressing Modes**.)

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 free location on the stack. During a reset or after the reset stack pointer (RSP) instruction, the stack pointer contents are 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.

	Bit15	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, and 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 $00C0 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 two most significant bits of the program counter are permanently fixed at 00.

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.

	Bit 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	Bit 0
	0	0														
RESET:	0	0	LOADED WITH RESET VECTOR $3FFE AND $3FFF													

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 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.

3.1.5.4 Zero Flag

The CPU sets the zero flag when an arithmetic operation, logical operation, or data manipulation produces a $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
- Capture/compare timer
- Serial peripheral interface (SPI)
- Serial communications interface (SCI)

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. When the current instruction is complete, the processor checks all pending hardware interrupts. If interrupts are not masked (condition code register I bit clear) and if the corresponding interrupt enable bit is set, the processor proceeds with interrupt processing; otherwise, the next instruction is fetched and executed. Interrupt processing automatically saves the CPU registers on the stack and loads the program counter with a user-defined interrupt vector address. (Table 4-1 lists vector addresses for all interrupts including reset.)

4.1.1 Software Interrupt

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

4.1.2 External Interrupt

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. If the I bit is clear, the CPU then begins the interrupt sequence. The CPU clears the interrupt latch during interrupt processing, so that another interrupt signal on the IRQ pin can latch another interrupt request during the interrupt service routine. 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 of this programmable option.

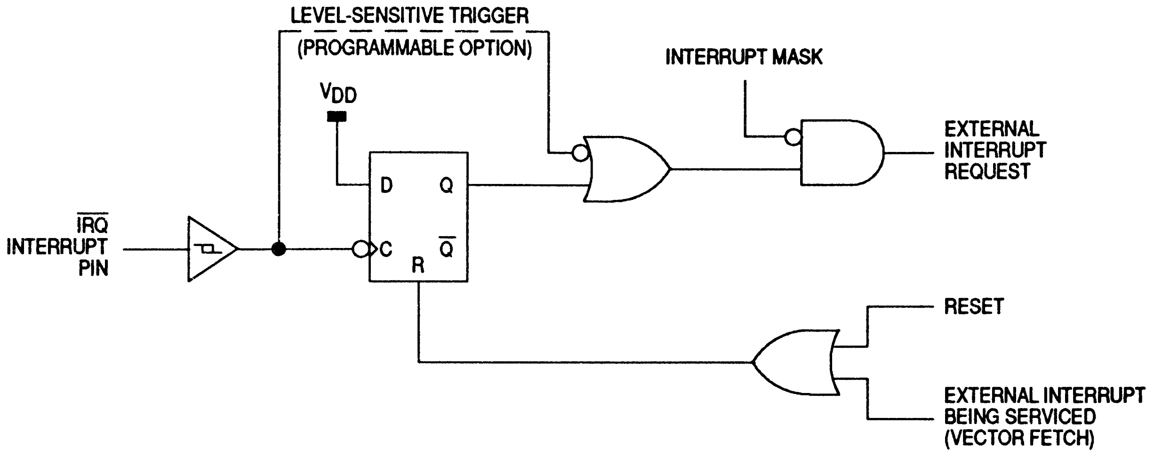


Figure 4-1. External Interrupt Trigger Option

Setting the interrupt mask in the condition code register disables external interrupts.

The $\overline{\text{IRQ}}$ pin can be edge-triggered or edge- and level-triggered. $\overline{\text{IRQ}}$ pin sensitivity is programmable in the option register (OR). (Refer to **1.3 Programmable Options**.) The level-sensitive trigger option 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.

4.1.3 Capture/Compare Timer Interrupts

The capture/compare timer can generate the following interrupts:

- Input capture interrupt
- Output compare interrupt
- Timer overflow interrupt

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

4.1.3.1 Input Capture Interrupt

An input capture interrupt request occurs if the input capture flag, ICF, becomes set while the input capture interrupt enable bit, ICIE, is also set. ICF is in the timer status register, and ICIE is in the timer control register. (Refer to **SECTION 8 CAPTURE/COMPARE TIMER**.)

4.1.3.2 Output Compare Interrupt

An output compare interrupt request occurs if the output compare flag, OCF, becomes set while the output compare interrupt enable bit, OCIE, is also set. OCF is in the timer status register, and OCIE is in the timer control register. (Refer to **SECTION 8 CAPTURE/COMPARE TIMER**.)

4.1.3.3 Timer Overflow Interrupt

A timer overflow interrupt request occurs if the timer overflow flag, TOF, is set while the timer overflow interrupt enable bit ,TOIE, is also set. TOF is in the timer status register, and TOIE is in the timer control register. (Refer to **SECTION 8 CAPTURE/COMPARE TIMER**.)

4.1.4 SCI Interrupt

An interrupt in the SCI occurs when one of the interrupt flag bits in the serial communications interface status register (SCSR) is set, provided the I bit in the condition code register is clear and the SCI transmit interrupt enable bit (TIE) in the serial communications control register 2 (SCCR2) is set. Software in the SCI interrupt service routine examines the interrupt flag bits in SCSR to determine the cause and priority of the SCI interrupt. (Refer to **SECTION 9 SERIAL COMMUNICATIONS INTERFACE**.)

4.1.5 SPI Interrupt

An interrupt in the SPI occurs when one of the interrupt flag bits in the serial peripheral status register (SPSR) is set, provided the I bit in the condition code register is clear and the serial peripheral interrupt enable bit (SPIE) in the serial peripheral control register (SPCR) is set. Software in the SPI interrupt service routine examines the interrupt flag bits in SPSR to determine the cause and priority of the SPI interrupt. (Refer to **SECTION 10 SERIAL PERIPHERAL INTERFACE**.)

4.2 Interrupt Processing

The CPU takes the following actions to begin servicing an interrupt:

- Stores the CPU registers on the stack as shown in Figure 4-2

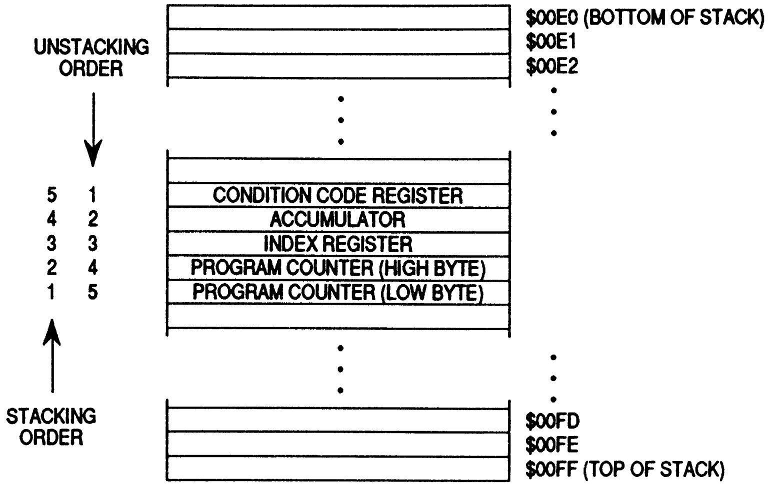


Figure 4-2. Interrupt Stacking Order

- Sets the I bit in the condition code register to prevent further interrupts
- Loads the program counter with contents of appropriate interrupt vector locations:
  - — $3FFC and $3FFD (software interrupt vector)
  - — $3FFA and $3FFB (external interrupt vector)
  - — $3FF8 and $3FF9 (timer interrupt vector)
  - — $3FF6 and $3FF7 (SCI vector)
  - — $3FF4 and $3FF5 (SPI vector)

The return from interrupt (RTI) instruction causes the CPU to recover the CPU register from the stack as shown in Figure 4-2.

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

Table 4-1. Reset/Interrupt Vector Addresses

Type	Source	Local Mask	Global Mask	Priority (1 = High)	Vector Address
Reset	Power-on Reset Reset Pin COP Watchdog * Clock Monitor *	None None	None None	1 1	\$3FFE–\$3FFF \$3FFE–\$3FFF
Software Interrupt (SWI)	User Code	None	None	Same Priority As Instruction	\$3FFC–\$3FFD
External Interrupt	IRQ Pin	None	I Bit	2	\$3FFA–\$3FFB
Timer Interrupts	ICF OCF TOF	ICIE OCIE TOIE	I Bit	3	\$3FF8–\$3FF9
SCI Interrupt	TDRE	TIE	I Bit	4	\$3FF6–\$3FF7
SPI Interrupt	SPIF	SPIE	I Bit	5	\$3FF4–\$3FF5

* The COP watchdog and clock monitor are programmable in the COP control register (COPCR).

Figure 4-3 shows the sequence of events caused by an interrupt.

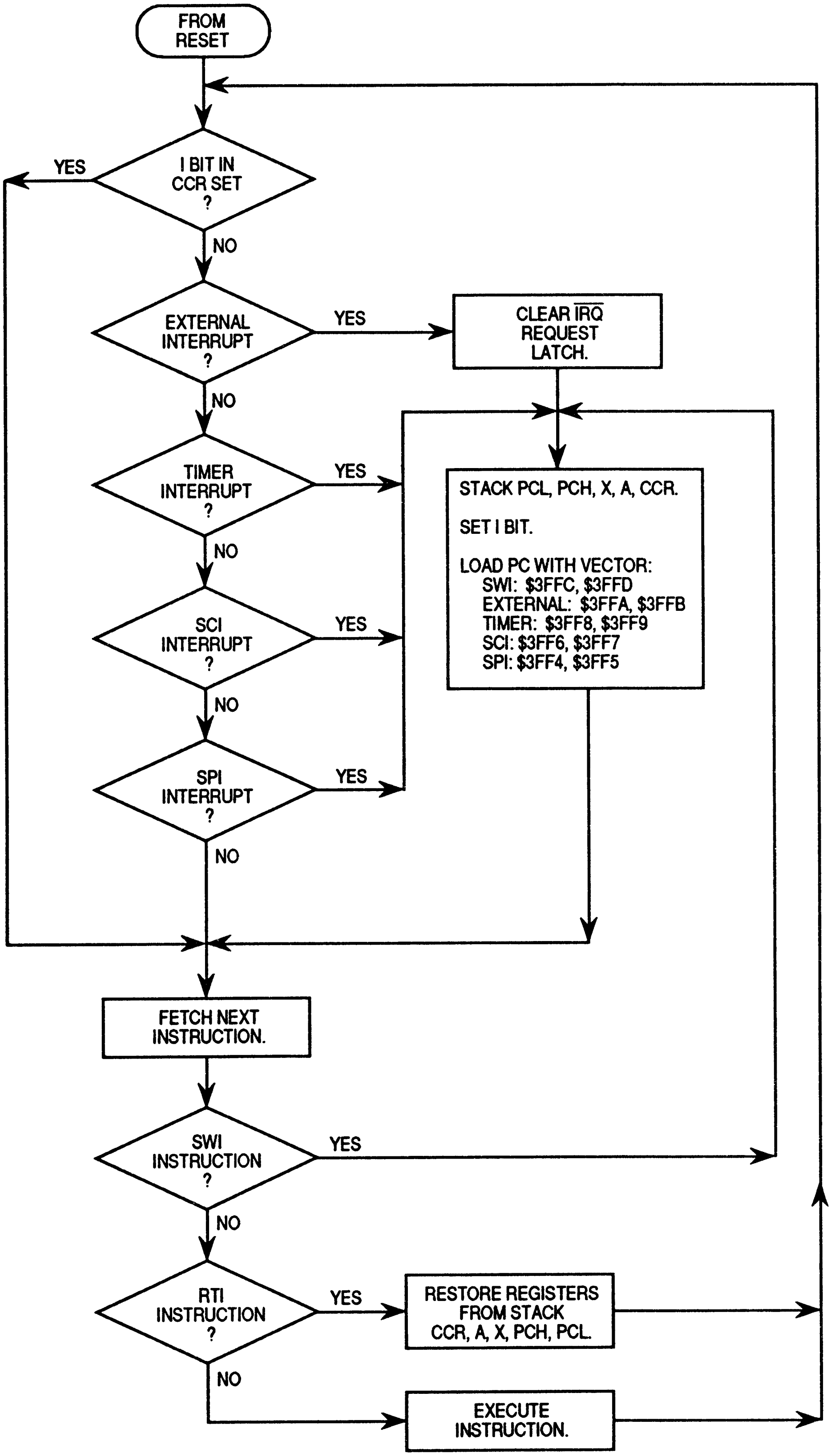


Figure 4-3. Reset and Interrupt Processing Flowchart

SECTION 5
RESETS

This section describes how resets initialize the MCU.

5.1 Reset Sources

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

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 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. (See Figure 5-1.) After 6 t_{CYC}, $\overline{\text{RESET}}$ input is sampled. If the pin is still low, an external reset has occurred. If the $\overline{\text{RESET}}$ input is high, then the reset was initiated internally by either the COP watchdog timer or by the clock monitor. This method of differentiating between external and internal reset conditions assumes that the $\overline{\text{RESET}}$ pin will rise to a logic 1 less than 2 t_{CYC} after its release, and that an externally generated reset should stay active for at least 8 t_{CYC}.

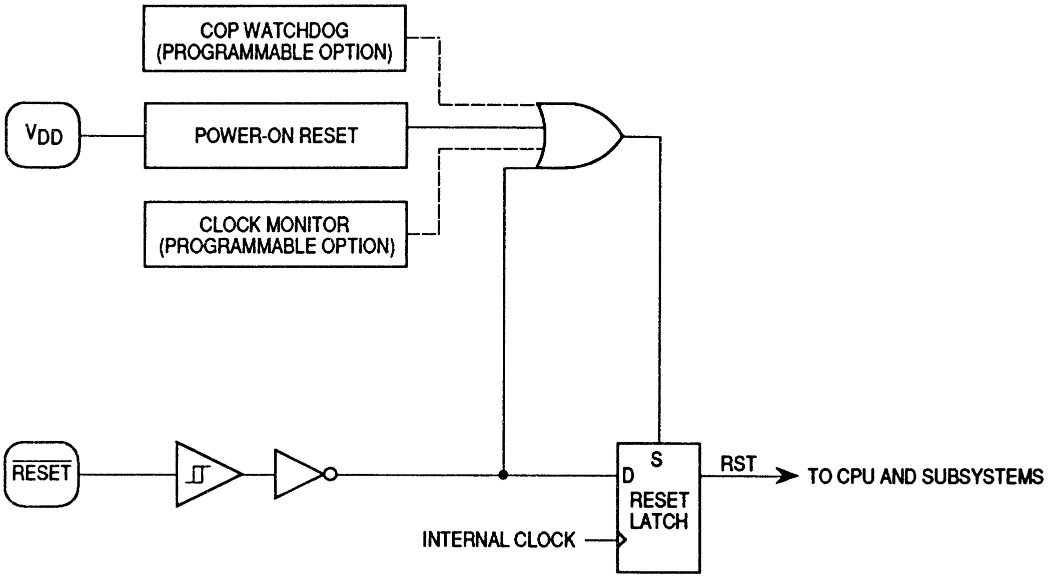


Figure 5-1. Reset Sources

5.1.3 COP Watchdog Timer Reset

A timeout of the computer operating properly (COP) timer generates a COP reset. 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.

5.1.3.1 COP Reset Register (COPRST)

The COPRST register shown in Figure 5-2 resets the COP watchdog timer.

COPRST — COP Reset **$001D**

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

Figure 5-2. COP Reset (COPRST) Register

The sequence required to reset the COP timer is as follows:

1. Write $55 to the COP reset register.
2. Write $AA to the COP reset register.

Both write operations must occur in the order listed, but any number of instructions can be executed between the two write operations. The elapsed time between software resets must not be greater than the COP timeout period. Reading the COP reset register does not return valid data and does not affect the watchdog timer.

5.1.3.2 COP Control Register (COPCR)

Because the COP timer uses the internal clock, a clock monitor is included to guard against clock failure. The cop control register shown in Figure 5-3 is used to control the clock monitor functions.

COPCR — COP Control Register **$001E**

	Bit 7	6	5	4	3	2	1	Bit 0
	0	0	0	COPF	CME	COPE	CM1	CM0
RESET:	0	0	0	0	0	0	0	0

Figure 5-3. COP Control (COPCR) Register

COPF — Computer Operating Properly

Reading the COP control register clears COPF.

1 = COP or clock monitor reset occurred
0 = No COP or clock monitor reset occurred

CME — Clock Monitor Enable

CME is readable at any time, but can be written only once after reset.

1 = Clock monitor enabled
0 = Clock monitor disabled

COPE — Computer Operating Properly Enable

COPE is readable any time, but can be written only once after reset.

1 = COP timeout enabled
0 = COP timeout disabled

CM1 — Computer Operating Properly Mode 1

CM1 and CM0 control the COP timeout period. CM1 can be read and set any time, but is cleared only by reset. (Refer to Table 5-1.)

CM0 — Computer Operating Properly Mode 0

CM0 and CM1 control the COP timeout period. CM1 can be read and set any time, but is cleared only by reset. (Refer to Table 5-1.)

Bits 7–5 — Not used; always read zero

Table 5-1. COP Timeout Periods

CM1	CM0	$f_{OP}/2^{15}$ Divided By	XTAL=4.0 MHZ f_{OP}=2.0 MHZ	XTAL=3.5795 MHZ f_{OP}=1.7897 MHz	XTAL=2.0 MHz f_{OP}=1.0 MHz	XTAL=1.0 MHz f_{OP}=0.5 MHz
0	0	1	16.38 ms	18.31 ms	32.77 ms	65.54 ms
0	1	4	65.54 ms	73.24 ms	131.07 ms	262.14 ms
1	0	16	262.14 ms	292.95 ms	524.29 ms	1.048 s
1	1	64	1.048 s	1.172 s	2.097 s	4.194 s

f_{OP} = Crystal ÷ 2

5.1.4 Clock Monitor Reset

When the CME bit in the COP control register is set, a clock monitor detects the absence of the internal clock for a certain period of time. The timeout period depends on processing parameters and varies from 5 to 100 µs, which implies that systems using an internal clock rate of 200 kHz or less should not use the clock monitor function.

If a slow or absent clock is detected, the clock monitor causes a system reset. The reset is issued to the external system for four internal clock cycles via the bidirectional $\overline{\text{RESET}}$ pin.

Special consideration is required when using the STOP instruction with the clock monitor. Because STOP causes the internal clock to halt, the clock monitor issues a system reset when STOP is executed.

The clock monitor is a useful backup to the COP watchdog system. Because the watchdog timer requires the internal clock to function, it cannot indicate a system clock failure. The clock monitor would detect such a condition and force the MCU to a reset state. The internal clock is not required for the MCU to reach a reset condition. It is, however, required to bring the MCU through the reset sequence and back to the run condition.

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 interrupt mask (I bit), inhibiting interrupts
- Loads the program counter with the user-defined reset vector from locations $3FFE and $3FFF
- Clears the stop latch, enabling the CPU clock
- Clears the wait latch, waking the CPU from the wait mode

5.2.2 I/O Port Registers

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

- Clears all implemented data direction register bits so that the corresponding I/O pins are inputs
- Has no effect on port data registers

5.2.3 Capture/Compare Timer

A reset has the following effects on the capture/compare timer:

- Clears timer control register except for IEDG bit, which is unaffected
- Loads the timer counter with $FFFC
- Does not affect flags in timer status register (input capture, output compare, timer overflow); clears remaining bits

5.2.4 SCI

A reset has the following effects on the SCI registers:

- Clears control bits (RDRF, IDLE, OR, NF, FE); sets TDRE and TC to 1
- Clears serial communications control register 2
- Does not affect bits in serial communications control register 1
- Does not affect bits in serial communications data register

5.2.5 SPI

A reset has the following effects on the SPI registers:

- Clears flags in SPI status register (SPIF, WCOL, MODF)
- Clears control bits of high nibble in SPI control register (SPIE, SPE, DWOM, MSTR); lower nibble bits pertaining to clock are unaffected (CPOL, CPHA, SPR1:0)
- Does not affect bits in SPI data register

5.2.6 COP Watchdog

A reset clears the COP watchdog (if the COP watchdog is enabled by the COPE bit in the COP control register).

SECTION 6
LOW POWER MODES

This section describes stop, wait, and data-retention modes, and the timer, SCI, and SPI in low power modes.

6.1 Stop Mode

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

- Stops the internal oscillator, turning off the CPU clock, the timer clock, and the COP watchdog
- Stops the baud rate generator, halting all SCI activity and master mode SPI operations.
- Clears the I bit in the condition code register, enabling external interrupts
- Clears the ICF, OCF, and TOF interrupt flags in the timer status register, removing any pending timer interrupts
- Clears the ICIE, OCIE, and TOIE bits in the timer control register, disabling further timer interrupts

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

The following conditions bring the MCU out of stop mode:

- External interrupt signal on $\overline{\text{IRQ}}$ pin. A high-to-low transition on the $\overline{\text{IRQ}}$ pin loads the program counter with the contents of locations \$3FFA and \$3FFB (IRQ vector). The timer resumes counting from the last value before the STOP instruction.
- External reset. A logic zero on the RESET pin resets the MCU and loads the program counter with the contents of locations \$3FFE and \$3FFF (reset vector). The timer begins counting from \$FFFC.

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

An active edge on the TCAP pin during stop mode sets the ICF flag when an external interrupt brings the MCU out of stop mode. An external interrupt also latches the value in the timer registers into the input capture registers.

If a reset brings the MCU out of stop mode, then an active edge on the TCAP pin during stop mode has no effect on the ICF flag or the input capture registers.

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

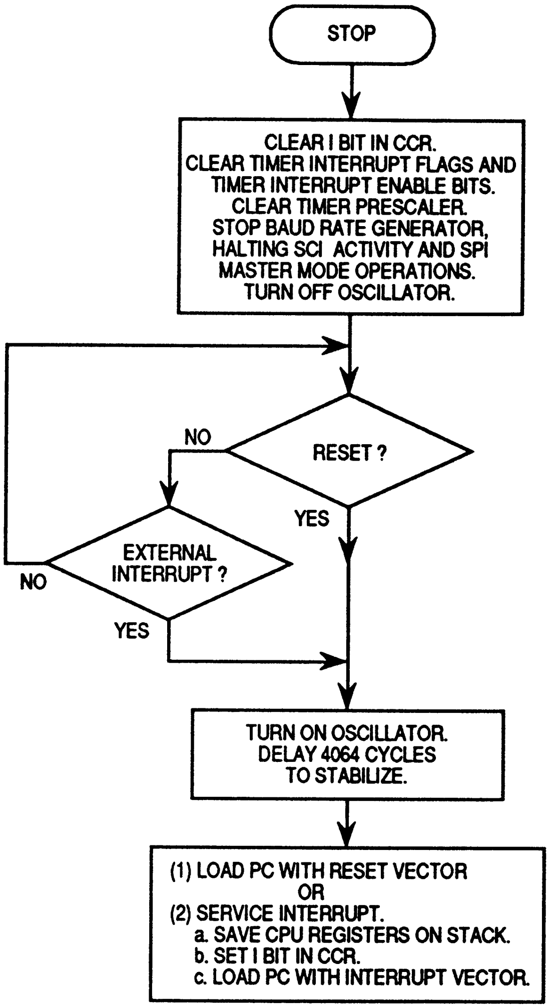


Figure 6-1. STOP Instruction Flowchart

6.1.1 Timer During Stop Mode

In stop mode, the capture/compare timer stops counting and holds the last count value. If $\overline{\text{IRQ}}$ is used to exit stop mode, the timer resumes counting from the count value present when stop mode was entered. If $\overline{\text{RESET}}$ is used to exit stop mode, the counter is forced to $FFFC.

If an active edge occurs on the TCAP pin during stop mode, ICF goes high as soon as an external interrupt brings the MCU out of stop mode. If a power-on reset or a logic zero on the $\overline{\text{RESET}}$ pin brings the MCU out of stop mode, all timer interrupt enable bits are cleared.

6.1.2 SCI During Stop Mode

When the MCU enters stop mode, the baud rate generator stops, halting all SCI activity. If the STOP instruction is executed during a transmitter transfer, that transfer is halted. If a low input to the $\overline{\text{IRQ}}$ pin is used to exit stop mode, the transfer resumes.

If the SCI receiver is receiving data and the stop mode is entered, received data sampling stops because the baud rate generator stops, and all subsequent data is lost. For these reasons, all SCI transfers should be in the idle state when the STOP instruction is executed.

6.1.3 SPI During Stop Mode

When the MCU enters stop mode, the baud rate generator stops, terminating all master mode SPI operations. If the STOP instruction is executed during an SPI transfer, that transfer halts until the MCU exits the stop mode by a low signal on the $\overline{\text{IRQ}}$ pin.

If reset is used to exit stop mode, then the SPI control and status bits are cleared and the SPI is disabled.

If the MCU is in slave mode when the STOP instruction is executed, the slave SPI continues to operate and can still accept data and clock information in addition to transmitting its own data back to a master device.

At the end of a possible transmission with a slave SPI in the stop mode, no flags are set until a low on the $\overline{\text{IRQ}}$ pin wakes up the MCU. Care should be taken when operating the SPI as a slave during the stop mode because the protective circuitry (WCOL, MODF, etc.) is inactive.

6.2 Wait Mode

The WAIT instruction puts the MCU in an intermediate power-consumption mode. All CPU activity is suspended, but the oscillator, capture/compare timer, SCI, and SPI remain active. Any interrupt or reset will cause the MCU to exit wait mode.

The WAIT instruction 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 clock to drive the capture/compare timer, SCI, and SPI

The WAIT instruction does not affect any other registers, I/O lines, or memory. The capture/compare timer, SCI, and SPI can be enabled to allow a periodic exit from the wait mode.

The following conditions bring the MCU out of wait mode:

- External interrupt — A high-to-low transition on the $\overline{\text{IRQ}}$ pin loads the program counter with the contents of locations $3FFA and $3FFB (IRQ vector).
- Timer interrupt — Input capture, output compare, and timer overflow interrupt requests load the program counter with the contents of locations $3FF8 and $3FF9 (timer vector).
- SCI interrupt — An SCI interrupt loads the program counter with the contents of locations $3FF6 and $3FF7 (SCI vector).
- SPI interrupt — An SPI interrupt loads the program counter with the contents of locations $3FF4 and $3FF5 (SPI vector).
- COP watchdog reset — A timeout of the COP watchdog resets the MCU and loads the program counter with the contents of locations $3FFE and $3FFF (reset vector). Software can enable timer 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 $3FFE and $3FFF (reset vector).

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

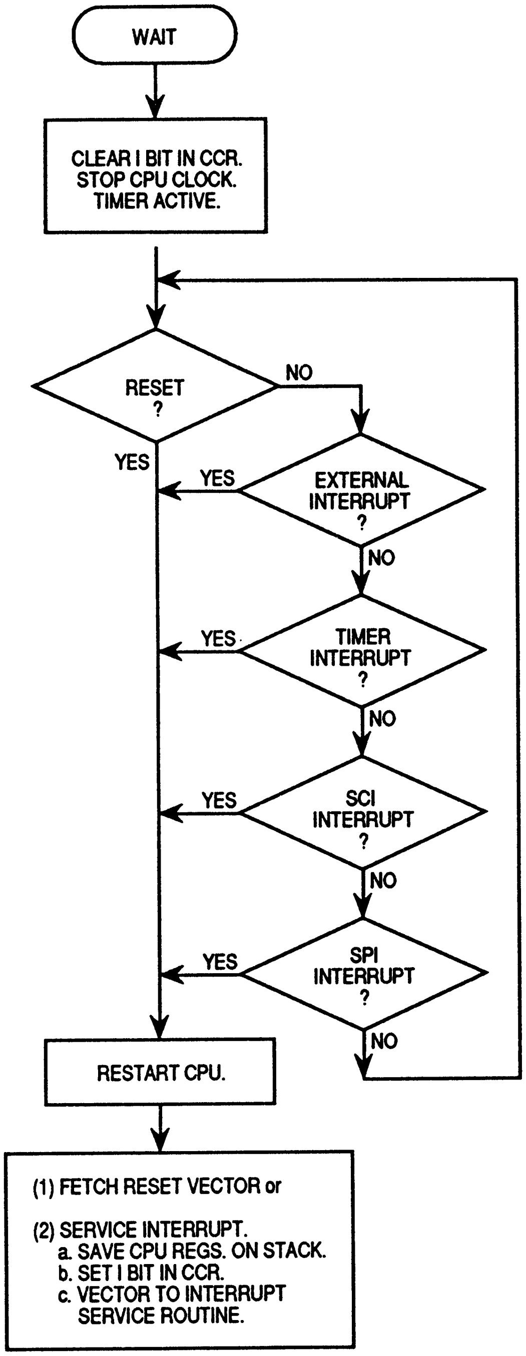


Figure 6-2. WAIT Instruction Flowchart

6.3 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 $\overline{\text{RESET}}$ pin to zero.
2. Lower V_{DD} voltage. $\overline{\text{RESET}}$ line 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 $\overline{\text{RESET}}$ pin to logic one.

SECTION 7
PARALLEL I/O

This section describes the four parallel I/O ports.

7.1 I/O Port Function

Thirty-one bidirectional I/O pins of the MCU form four parallel I/O ports. (Bit 6 of port D is implemented input-only; no data direction bit or output latch is provided). 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:	NOT CHANGED BY RESET							

Figure 7-1. Port A Data Register

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 DDRA bit.

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 clears all DDRA bits, 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

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.

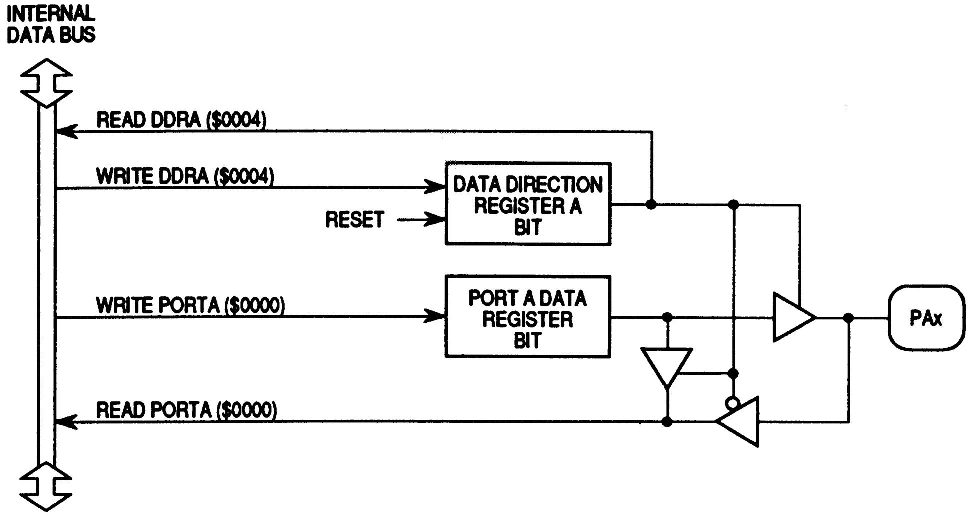


Figure 7-3. Port A I/O Circuit

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 an 8-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 eight port B pins.

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

	Bit 7	6	5	4	3	2	1	Bit 0
	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
RESET:	NOT CHANGED BY RESET							

Figure 7-4. Port B Data Register

PB7–PB0 — Port B Data Bits

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

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 clears all DDRB bits, configuring all port B pins as inputs.

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

	Bit 7	6	5	4	3	2	1	Bit 0
	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
RESET:	0	0	0	0	0	0	0	0

Figure 7-5. Data Direction Register B

DDRB7–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

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.

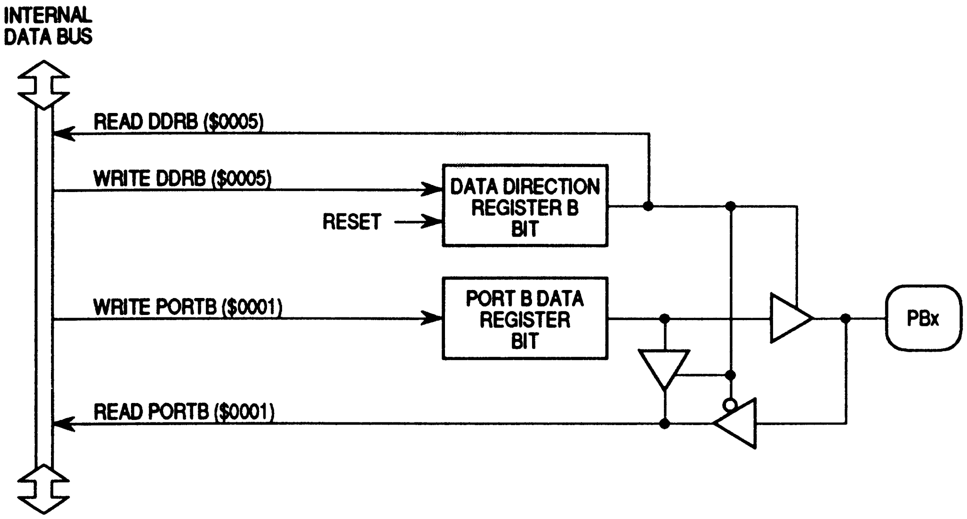


Figure 7-6. Port B I/O Circuit

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	DDRB7–0	Pin	NOTE 2
1	X	Output	DDRB7–0	PB7–0	PB7–0

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 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:	NOT CHANGED BY RESET							

Figure 7-7. Port C Data Register

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 DDRC bit.

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 clears all DDRC bits, 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

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.

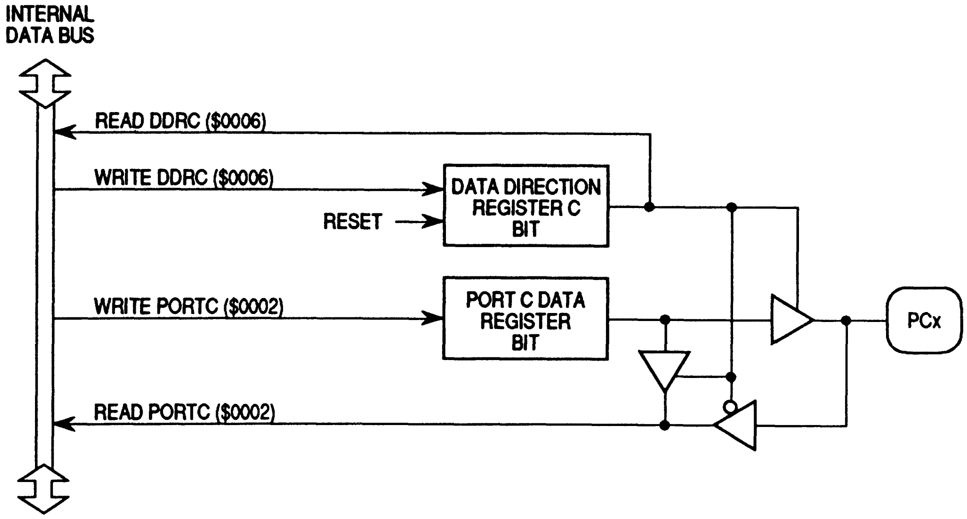


Figure 7-9. Port C I/O Circuit

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 a 7-bit port that shares its pins with the SCI and SPI subsystems. When the SCI or the SPI is disabled, the pins become available as I/O pins under software control.

The SPI requires four of the port D pins for slave select ($\overline{SS}$), serial clock (SCK), serial data output/input (MOSI), and serial data input/output (MISO).

The SCI requires two pins for its transmit data output (TDO) and receive data input (RDI).

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 seven port D pins.

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

	Bit 7	6	5	4	3	2	1	Bit 0
	PD7	—	PD5	PD4	PD3	PD2	PD1	PD0
RESET:	NOT CHANGED BY RESET							
ALTERNATE FUNCTION:	—	—	$\overline{SS}$	SCK	MOSI	MISO	TDO	RDI

Figure 7-10. Port D Data Register

PD7, PD5–PD0 — Port D Data Bits

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

7.5.2 Data Direction Register D (DDRD)

Data direction register D, shown in Figure 7-11, determines whether the seven port D pins are inputs or outputs. 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 clears bits DDRD7, DDRD5–DDRD0, configuring all six port D pins as inputs.

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

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

Figure 7-11. Port D Data Direction Register

DDRD7, DDRD5–DDRD0 — Port D Data Direction Bits

These read/write bits control port D data direction.

1 = Corresponding port pin configured as output
0 = Corresponding port 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.

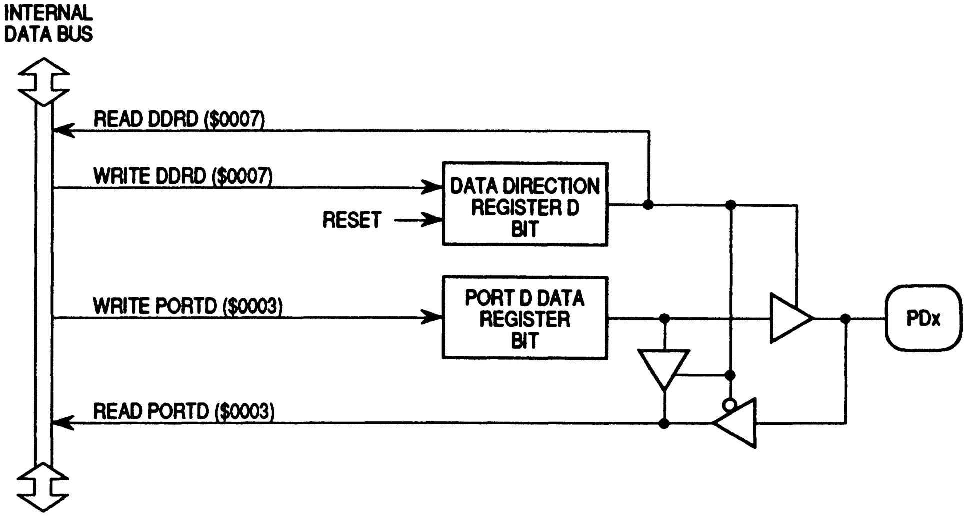


Figure 7-12. Port D Circuit

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, DDRD5–0	Pin	NOTE 2
1	X	Output	DDRD7, DDRD5–0	PD7, PD5–0	PD7, PD5–0

NOTES:

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

7.5.3 Port D Inputs

When a port D I/O pin is programmed as an input, reading the port bit reads the voltage level on the I/O pin, even for pins used as outputs by enabled subsystems.

NOTE

SCI, SPI, or timer output compare function operations can cause transitions on port D output pins that are asynchronous with the timing of read cycles on the port D data register. The programmer should ensure that a program does not depend on the value of any port D bit read while the operation of a subsystem may have caused transitions on the corresponding pin.

A zero on a port D data direction register bit forces the associated SPI output pin to high-impedance and input mode.

7.5.4 Port D Outputs

When a serial subsystem is disabled, the associated port D pins become available as I/O pins. The corresponding DDRD bits must be written to one for the pins to become outputs, in which case the pins get their drive signals from the port D data register.

When SCI or SPI is enabled, pins associated with that subsystem become outputs, and their drive signals come from the serial subsystem logic rather than from the port D data register.

The SPI output pins, however, remain subject to port D DDRD bits. A DDRD bit can always disable the output drive of its associated SPI driver, even when the SPI subsystem would otherwise be driving that pin as an ouput.

The state of any port D pin can be read at any time, provided that its corresponding data direction bit is set to zero.

SPI output pins not needed can be used as programmed input bits of port D.

7.5.5 Open-Drain Control

A special port D control bit, DWOM, is provided as bit 5 of the SPI control register (SPCR). When set, this bit disables all active pullup devices on port D. Any pin set as an output then becomes open-drain, allowing Wired-OR use with other drivers. DWOM affects all drivers on port D.

SECTION 8
CAPTURE/COMPARE TIMER

This section describes the operation of the 16-bit capture/compare timer. Figure 8-1 shows the structure of the capture/compare timer subsystem.

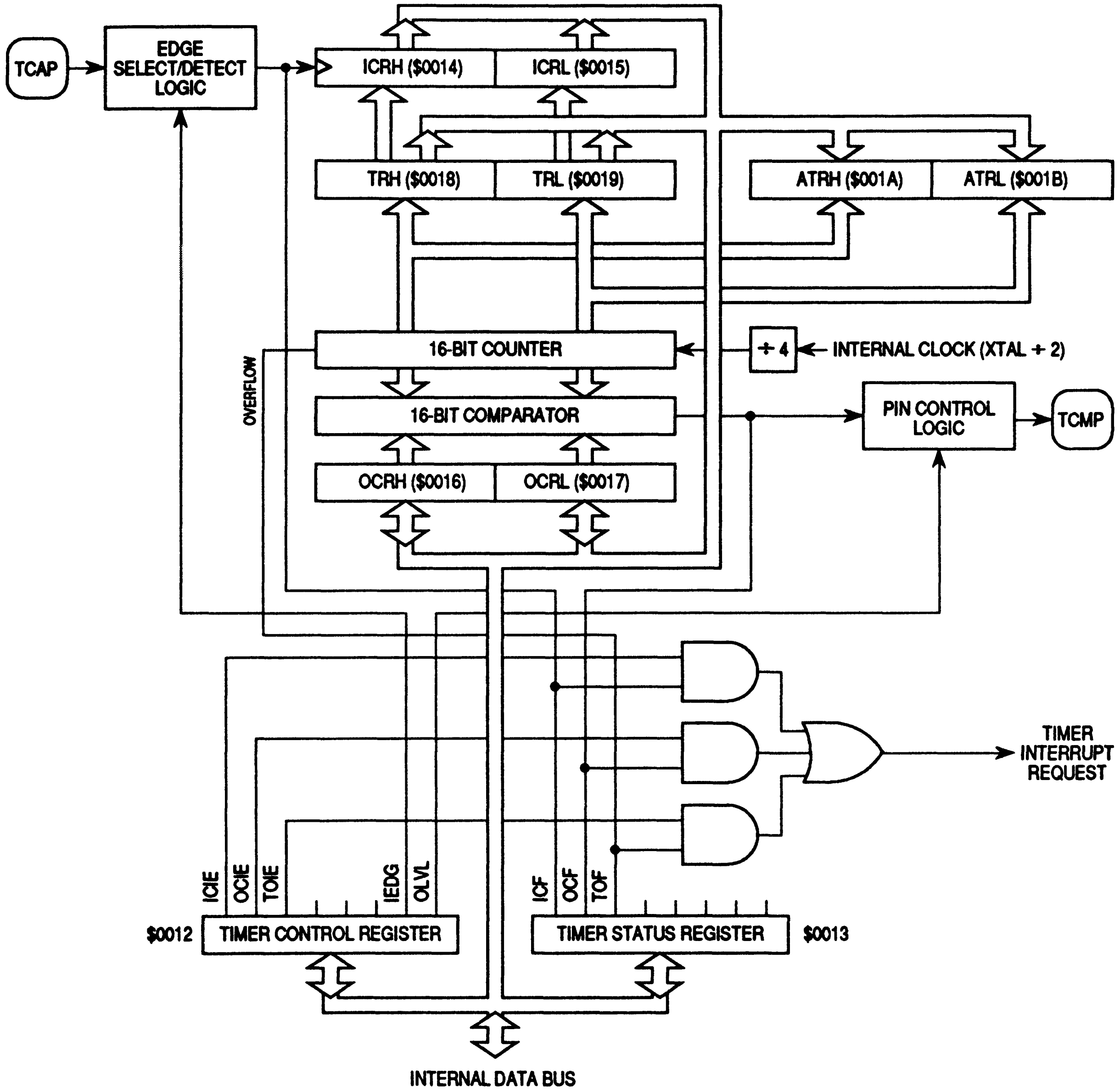


Figure 8-1. Capture/Compare Timer Block Diagram

8.1. Timer Operation

The core of the capture/compare timer is a 16-bit free-running counter. The counter provides the timing reference for the input capture and output compare functions. The input capture and output compare functions provide a means to latch the times at which external events occur, to measure input waveforms, and to generate output waveforms and timing delays. Software can read the value in the 16-bit free-running counter at any time without affecting the counter sequence.

Because of the 16-bit timer architecture, the I/O registers for the input capture and output compare functions are pairs of 8-bit registers.

Because the counter is 16 bits long and preceded by a fixed divide-by-four prescaler, the counter rolls over every 262,144 internal clock cycles. Timer resolution with a 4-MHz crystal is 2 µs.

8.1.1 Input Capture

The input capture function is a means to record the time at which an external event occurs. When the input capture circuitry detects an active edge on the TCAP pin, it latches the contents of the timer registers into the input capture registers. The polarity of the active edge is programmable.

Latching values into the input capture registers at successive edges of the same polarity measures the period of the input signal on the TCAP pin. Latching the counter values at successive edges of opposite polarity measures the pulse width of the signal. Figure 8-2 shows the logic of the input capture function.

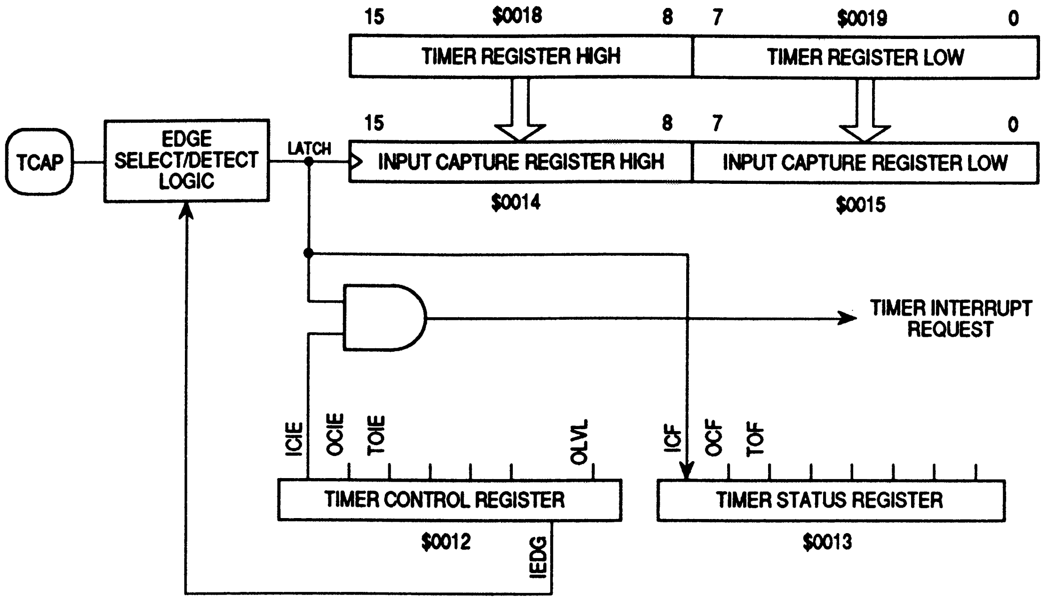


Figure 8-2. Input Capture Operation

8.1.2 Output Compare

The output compare function is a means of generating an output signal when the 16-bit counter reaches a selected value. Software writes the selected value into the output compare registers. On every fourth internal clock cycle the output compare circuitry compares the value of the counter to the value written in the output compare registers. When a match occurs, the timer transfers the programmable output level bit (OLVL) from the timer control register to the TCMP pin.

The programmer can use the output compare register to measure time periods, to generate timing delays, or to generate a pulse of specific duration or a pulse train of specific frequency and duty cycle on the TCMP pin. Figure 8-3 shows the logic of the output compare function.

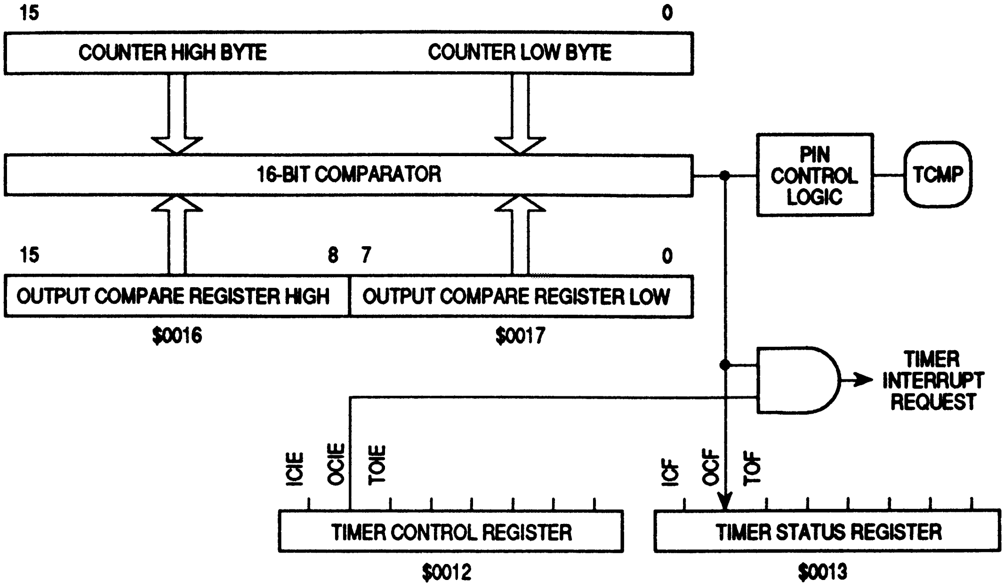


Figure 8-3. Output Compare Operation

8.2 Timer I/O Registers

The following registers control and monitor the operation of the timer:

- Timer control register (TCR)
- Timer status register (TSR)
- Timer registers (TRH and TRL)
- Alternate timer registers (ATRH and ATRL)
- Input Capture registers (ICRH and ICRL)
- Output compare registers (OCRH and OCRL)

8.2.1 Timer Control Register (TCR)

The timer control register, shown in Figure 8-4, performs the following functions:

- Enables input capture interrupts
- Enables output compare interrupts
- Enables timer overflow interrupts
- Controls the active edge polarity of the TCAP signal
- Controls the active level of the TCMP ouput

TCR — Timer Control Register **$0012**

	Bit 7	6	5	4	3	2	1	Bit 0
	ICIE	OCIE	TOIE	0	0	0	IEDG	OLVL
RESET:	0	0	0	0	0	0	U	0

(U = UNAFFECTED)

Figure 8-4. Timer Control Register (TCR)

ICIE — Input Capture Interrupt Enable

This read/write bit enables interrupts caused by an active signal on the TCAP pin. Resets clear the ICIE bit.

1 = Input capture interrupts enabled
0 = Input capture interrupts disabled

OCIE — Output Compare Interrupt Enable

This read/write bit enables interrupts caused by an active signal on the TCMP pin. Resets clear the OCIE bit.

1 = Output compare interrupts enabled
0 = Output compare interrupts disabled

TOIE — Timer Overflow Interrupt Enable

This read/write bit enables interrupts caused by a timer overflow. Resets clear the TOIE bit.

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

IEDG — Input Edge

The state of this read/write bit determines whether a positive or negative transition on the TCAP pin triggers a transfer of the contents of the timer register to the input capture register. Resets have no effect on the IEDG bit.

1 = Positive edge (low to high transition) triggers input capture
0 = Negative edge (high to low transition) triggers input capture

OLVL — Output Level

The state of this read/write bit determines whether a logic one or a logic zero appears on the TCMP pin when a successful output compare occurs. Resets clear the OLVL bit.

1 = TCMP goes high on output compare
0 = TCMP goes low on output compare

8.2.2 Timer Status Register (TSR)

The timer status register, shown in Figure 8-5, contains flags for the following events:

- An active signal on the TCAP pin, transferring the contents of the timer registers to the input capture registers
- A match between the 16-bit counter and the output compare registers, transferring the OLVL bit to the TCMP pin
- A timer rollover from $FFFF to $0000

TSR — Timer Status Register **$0012**

	Bit 7	6	5	4	3	2	1	Bit 0
	ICF	OCF	TOF	0	0	0	0	0
RESET:	U	U	U	0	0	0	0	0

U = UNAFFECTED

Figure 8-5. Timer Status Register (TSR)

ICF — Input Capture Flag

The ICF bit is automatically set when an edge of the selected polarity occurs on the TCAP pin. Clear the ICF bit by reading the timer status register with ICF set, and then reading the low byte ($0015) of the input capture registers. Resets have no effect on ICF.

OCF — Output Compare Flag

The OCF bit is automatically set when the value of the timer registers matches the contents of the output compare registers. Clear the OCF bit by reading the timer status register with OCF set, and then accessing the low byte ($0017) of the output compare registers. Resets have no effect on OCF.

TOF — Timer Overflow Flag

The TOF bit is automatically set when the 16-bit counter rolls over from $FFFF to $0000. Clear the TOF bit by reading the timer status register with TOF set, and then accessing the low byte ($0019) of the timer registers. Resets have no effect on TOF.

8.2.3 Timer Registers (TRH and TRL)

The timer registers, shown in Figure 8-6, contain the current high and low bytes of the 16-bit counter. Reading TRH before reading TRL causes TRL to be latched until TRL is read. Reading TRL after reading the timer status register clears the timer overflow flag (TOF). Writing to the timer registers has no effect.

TRH and TRL — Timer Registers High/Low **$0018 and $0019**

$0018	Bit 15	14	13	12	11	10	9	Bit 8
$0019	Bit 7	6	5	4	3	2	1	Bit 0

Reset initializes the timer registers to $FFFC.

Figure 8-6. Timer Registers (TRH and TRL)

Reading TRH returns the current value of the high byte of the counter and causes the low byte to be latched into a buffer, as shown in Figure 8-7. The buffer value remains fixed even if the high byte is read more than once. Reading TRL reads the transparent low byte buffer and completes the read sequence of the timer registers.

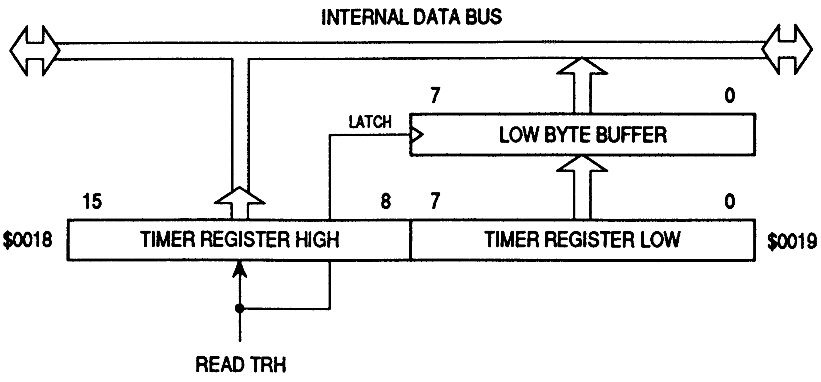


Figure 8-7. Timer Register Reads

8.2.4 Alternate Timer Registers (ATRH and ATRL)

The alternate timer registers, shown in Figure 8-8, contain the current high and low bytes of the 16-bit counter. Reading ATRH before reading ATRL causes ATRL to be latched until ATRL is read. Reading does not affect the timer overflow flag (TOF). Writing to the alternate timer registers has no effect.

ATRH and ATRL — Alternate Timer Registers High/Low **$001A and $001B**

$001A	Bit 15	14	13	12	11	10	9	Bit 8
$001B	Bit 7	6	5	4	3	2	1	Bit 0

Reset initializes the alternate timer registers to $FFFC.

Figure 8-8. AlternateTimer Registers (ATRH and ATRL)

Reading ATRH returns the current value of the high byte of the counter and causes the low byte to be latched into a buffer, as shown in Figure 8-9. The buffer value remains fixed even if the high byte is read more than once. Reading ATRL reads the transparent low byte buffer and completes the read sequence of the alternate timer registers.

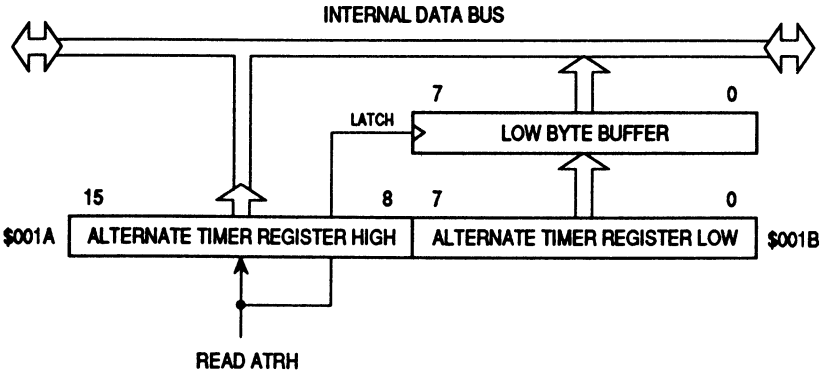


Figure 8-9. Alternate Timer Register Reads

NOTE

To prevent interrupts from occurring between readings of ATRH and ATRL, set the interrupt flag in the condition code register before reading ATRH, and clear the flag after reading ATRL.

8.2.5 Input Capture Registers (ICRH and ICRL)

When a selected edge occurs on the TCAP pin, the current high and low bytes of the 16-bit counter are latched into the input capture registers. Reading ICRH before reading ICRL inhibits further captures until ICRL is read. Reading ICRL after reading the timer status register clears the input capture flag (ICF). Writing to the input capture registers has no effect.

ICRH and ICRL — Input Capture Registers High/Low **$0014 and $0015**

$0014	Bit 15	14	13	12	11	10	9	Bit 8
$0015	Bit 7	6	5	4	3	2	1	Bit 0

Reset does not affect the input capture registers.

Figure 8-10. Input Capture Registers (ICRH and ICRL)

NOTE

To prevent interrupts from occurring between readings of ICRH and ICRL, set the interrupt flag in the condition code register before reading ICRH, and clear the flag after reading ICRL.

8.2.6 Output Compare Registers (OCRH and OCRL)

When the value of the 16-bit counter matches the value in the output compare registers, the planned TCMP pin action takes place. Writing to OCRH before writing to OCRL inhibits timer compares until OCRL is written. Reading or writing to OCRL after reading the timer status register clears the output compare flag (OCF).

OCRH and OCRL — Output Compare Registers High/Low **$0016 and $0017**

$0016	Bit 15	14	13	12	11	10	9	Bit 8
$0017	Bit 7	6	5	4	3	2	1	Bit 0

Reset does not affect the output compare registers.

Figure 8-11. Output Compare Registers (OCRH and OCRL)

To prevent OCF from being set between the time it is read and the time the output compare registers are updated, use the following procedure:

1. Disable interrupts by setting the I bit in the condition code register.
2. Write to OCRH. Compares are now inhibited until OCRL is written.
3. Clear bit OCF by reading the timer status register (TSR).
4. Enable the output compare function by writing to OCRL.
5. Enable interrupts by clearing the I bit in the condition code register.

SECTION 9
SERIAL COMMUNICATIONS INTERFACE (SCI)

This section describes the on-chip asynchronous serial communications interface (SCI).

9.1 Features

Features of the SCI include the following:

- Standard Mark/Space Non-Return-to-Zero Format
- Full Duplex Operation
- 32 Programmable Baud Rates
- Programmable 8-bit or 9-bit Character Length
- Separately Enabled Transmitter and Receiver
- Two Receiver Wakeup Methods:
  - Idle Line Wakeup
  - Address Mark Wakeup
- Interrupt-Driven Operation Capability with Five Interrupt Flags:
  - Transmitter Data Register Empty
  - Transmission Complete
  - Receiver Data Register Full
  - Receiver Overrun
  - Idle Receiver Input
- Receiver Framing Error Detection
- 1/16 Bit-Time Noise Detection

9.2 SCI Data Format

The SCI uses the standard non-return-to-zero mark/space data format illustrated in Figure 9-1.

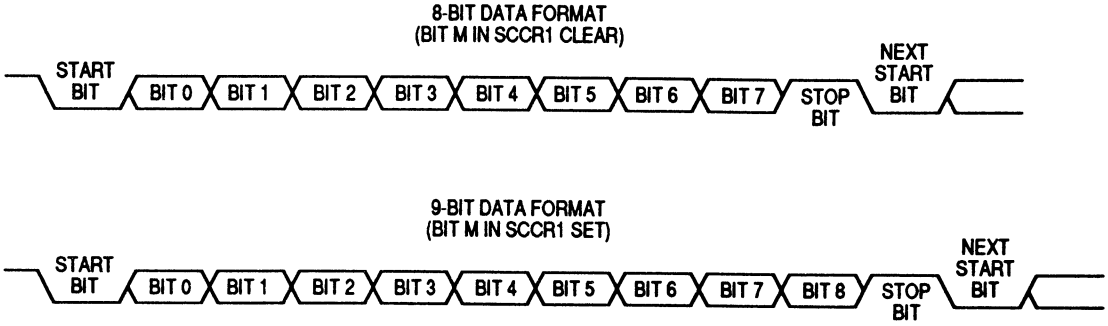


Figure 9-1. SCI Data Format

9.3 SCI Operation

The SCI allows full-duplex, asynchronous, RS232 or RS422 serial communication between the MCU and remote devices, including other MCUs. The transmitter and receiver of the SCI operate independently, although they use the same baud-rate generator. The following paragraphs describe the operation of the SCI transmitter and receiver.

9.3.1 Transmitter

Figure 9-2 shows the structure of the SCI transmitter.

9.3.1.1 Character Length

The transmitter can accommodate either 8-bit or 9-bit data. The state of the M bit in SCI control register 1 (SCCR1) determines character length. When transmitting 9-bit data, bit T8 in SCCR1 is the ninth bit (bit 8).

9.3.1.2 Character Transmission

During transmission, the transmit shift register shifts a character out to the PD1/TDO pin. The SCI data register (SCDR) is the write-only buffer between the internal data bus and the transmit shift register.

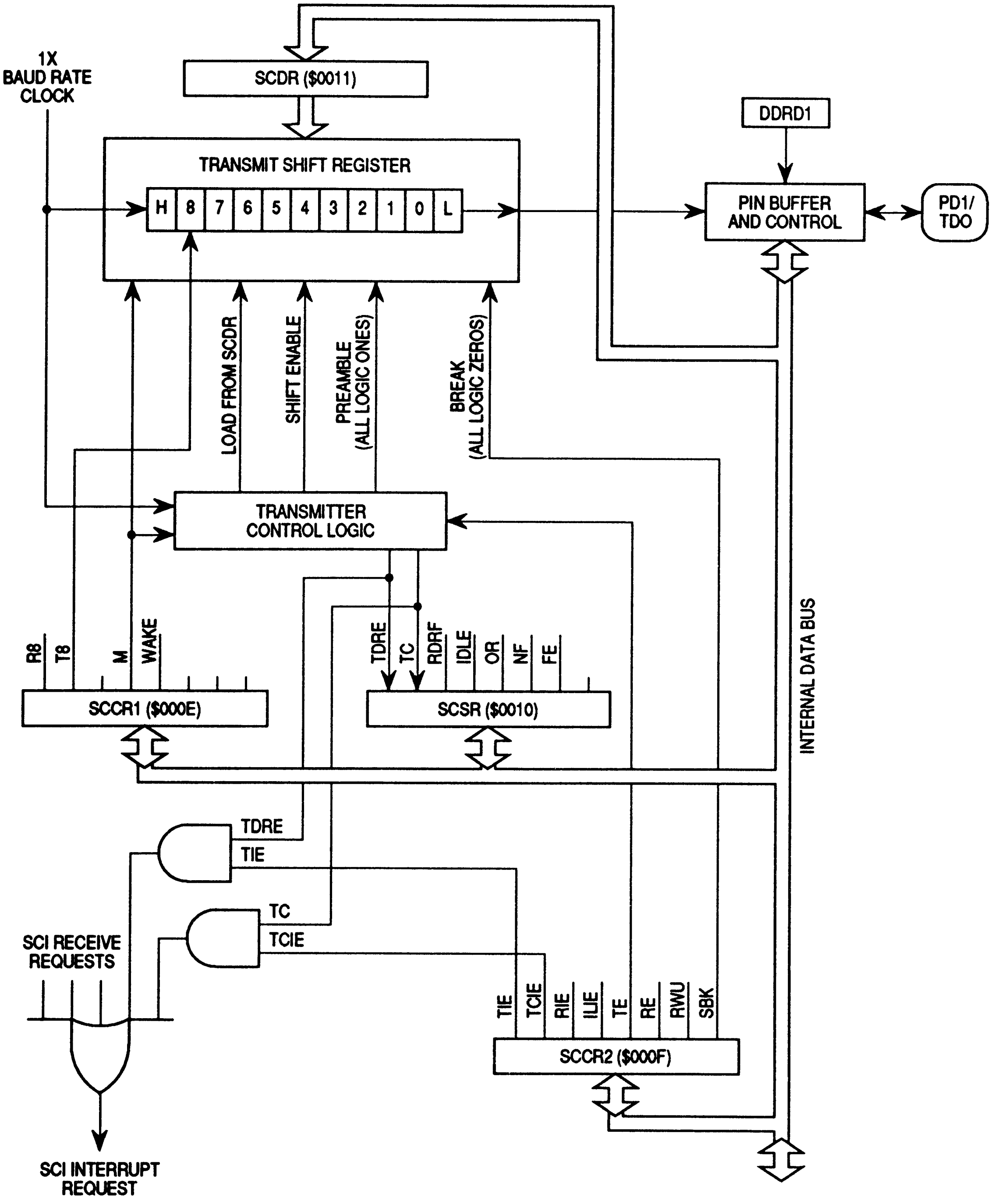


Figure 9-2. SCI Transmitter

Writing a logic one to the TE bit in SCI control register 2 (SCCR2) and then writing data to the SCDR begins the transmission. At the start of a transmission, transmitter control logic automatically loads the transmit shift register with a preamble of logic ones. After the preamble shifts out, the control logic transfers the SCDR data into the shift register. A logic zero start bit automatically goes into the least significant bit position of the shift register, and a logic one stop bit goes into the most significant bit position.

When the data in the SCDR transfers to the transmit shift register, the TDRE flag in the SCI status register (SCSR) becomes set. The TDRE flag indicates that the SCDR can accept new data from the internal data bus.

When the shift register is not transmitting a character, the PD1/TDO pin goes to the idle condition, logic one. If software clears the TE bit during the idle condition, and while TDRE is set, the transmitter relinquishes control of the PD1/TDO pin.

9.3.1.3 Break Characters

Writing a logic one to the SBK bit in SCCR2 loads the shift register with a break character. A break character contains all logic zeros and has no start and stop bits. Break character length depends on the M bit in SCCR1. As long as SBK is at logic one, transmitter logic continuously loads break characters into the shift register. After software clears the SBK bit, the shift register finishes transmitting the last break character and then transmits at least one logic one. The automatic logic one at the end of a break character is to guarantee the recognition of the start bit of the next character.

9.3.1.4 Idle Characters

An idle character contains all logic ones and has no start or stop bits. Idle character length depends on the M bit in SCCR1. The preamble is a synchronizing idle character that begins every transmission.

Clearing the TE bit during a transmission relinquishes the PD1/TDO pin after the last character to be transmitted is shifted out. The last character may already be in the shift register, or waiting in the SCDR, or a break character generated by writing to the SBK bit. Toggling TE from logic zero to logic one while the last character is in transmission generates an idle character (a preamble) that allows the receiver to maintain control of the PD1/TDO pin.

9.3.1.5 Transmitter Interrupts

The following sources can generate transmitter interrupt requests:

- Transmit Data Register Empty (TDRE) — The TDRE bit in the SCSR indicates that the SCDR has transferred a character to the transmit shift register.
- Transmission Complete (TC) — The TC bit in the SCSR indicates that both the transmit shift register and the SCDR are empty and that no break or idle character has been generated.

9.3.2 Receiver

Figure 9-3 shows the structure of the SCI receiver.

9.3.2.1 Character Length

The receiver can accommodate either 8-bit or 9-bit data. The state of the M bit in SCI control register 1 (SCCR1) determines character length. When receiving 9-bit data, bit R8 in SCCR1 is the ninth bit (bit 8).

9.3.2.2 Character Reception

During reception, the receive shift register shifts characters in from the PD0/RDI pin. The SCI data register (SCDR) is the read-only buffer between the internal data bus and the receive shift register.

After a complete character shifts into the receive shift register, the data portion of the character is transferred to the SCDR, setting the RDRF flag. The RDRF flag can be used to generate an interrupt.

9.3.2.3 Receiver Wakeup

So that the MCU can ignore transmissions intended only for other receivers in multiple-receiver systems, the MCU can be put into a standby state. Setting the RWU bit in SCI control register 2 (SCCR2) puts the MCU into a standby state during which receiver interrupts are disabled.

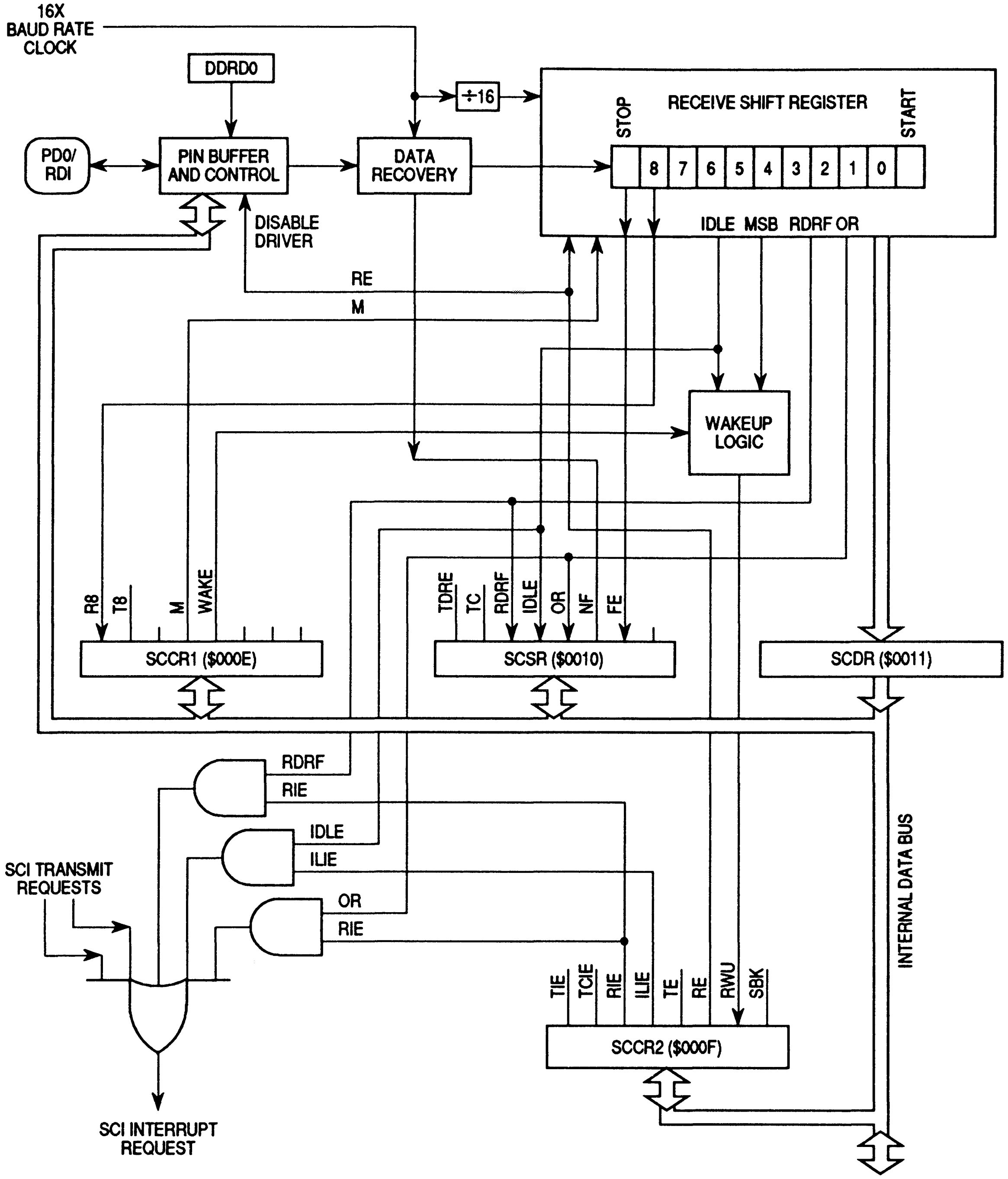


Figure 9-3. SCI Receiver

Either of two conditions on the PD0/RDI pin can bring the MCU out of the standby state:

- Idle input line condition — If the PD0/RDI pin is at logic one long enough for 10 or 11 logic ones to shift into the receive shift register, receiver interrupts are again enabled.
- Address mark — If a logic one occurs in the most significant bit position of a received character, receiver interrupts are again enabled.

The state of the WAKE bit in SCCR1 determines which of the two conditions wakes up the MCU.

9.3.2.4 Receiver Noise Immunity

The data recovery logic samples each bit 16 times to identify and verify the start bit and to detect noise. Any conflict between noise-detection samples sets the NF flag in the SCSR. The NF flag is set at the same time that the RDRF flag is set.

9.3.2.5 Framing Errors

If the data recovery logic does not detect a logic one where the stop bit should be in an incoming character, it sets the FE flag in the SCSR. The FE flag is set at the same time that the RDRF flag is set.

9.3.2.6 Receiver Interrupts

The following sources can generate receiver interrupt requests:

- Receive Data Register Full (RDRF) — The RDRF flag in the SCSR indicates that the receive shift register has transferred a character to the SCDR.
- Receiver Overrun (OR) — The OR flag in the SCSR indicates that the receive shift register shifted in a new character before the previous character was read from the SCDR.
- Idle Input (IDLE) — The IDLE flag in the SCSR indicates that 10 or 11 consecutive logic ones shifted in from the PD0/RDI pin.

9.4 SCI I/O Registers

The following I/O registers control and monitor SCI operation:

- SCI Data Register (SCDR)
- SCI Control Register 1 (SCCR1)
- SCI Control Register 2 (SCCR2)
- SCI Status Register (SCSR)

9.4.1 SCI Data Register (SCDR)

The SCI data register, shown in Figure 9-4, is the buffer for characters received and for characters transmitted.

SCDR — SCI Data Register **$0011**

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

Figure 9-4. SCI Data Register (SCDR)

9.4.2 SCI Control Register 1 (SCCR1)

SCI control register 1, shown in Figure 9-5, has the following functions:

- Stores ninth SCI data bit received and ninth SCI data bit transmitted
- Controls SCI character length
- Controls SCI wakeup method

SCCR1 — SCI Control Register 1 **$000E**

	Bit 7	6	5	4	3	2	1	Bit 0
	R8	T8	—	M	WAKE	—	—	—
RESET:	U	U	0	U	U	0	0	0

U = UNAFFECTED

Figure 9-5. SCI Control Register 1 (SCCR1)

R8 — Bit 8 (Received)

When the SCI is receiving 9-bit characters, R8 is the ninth bit of the received character. R8 receives the ninth bit at the same time that the SCDR receives the other eight bits. Resets have no effect on the R8 bit.

T8 — Bit 8 (Transmitted)

When the SCI is transmitting 9-bit characters, T8 is the ninth bit of the transmitted character. T8 is loaded into the transmit shift register at the same time that SCDR is loaded into the transmit shift register. Resets have no effect on the T8 bit.

M — Character Length

This read/write bit determines whether SCI characters are 8 bits long or 9 bits long. The ninth bit can be used as an extra stop bit, as a receiver wakeup signal, or as a mark or space parity bit. Resets have no effect on the M bit.

1 = 9-bit SCI characters
0 = 8-bit SCI characters

WAKE — Wakeup Method

This read/write bit determines which condition wakes up the SCI: a logic one (address mark) in the most significant bit (MSB) position of a received character or an idle condition on the PD0/RDI pin. Resets have no effect on the WAKE bit.

1 = Address mark wakeup
0 = Idle line wakeup

9.4.3 SCI Control Register 2 (SCCR2)

SCI control register 2, shown in Figure 9-6, has the following functions:

- Enables the SCI receiver and SCI receiver interrupts
- Enables the SCI transmitter and SCI transmitter interrupts
- Enables SCI receiver idle interrupts
- Enables SCI transmission complete interrupts
- Enables SCI wakeup
- Transmits SCI break characters

SCCR2 — SCI Control Register 2 **$000F**

	Bit 7	6	5	4	3	2	1	Bit 0
	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
RESET:	0	0	0	0	0	0	0	0

Figure 9-6. SCI Control Register 2 (SCCR2)

TIE — Transmit Interrupt Enable

This read/write bit enables SCI interrupt requests when the TDRE flag becomes set. Resets clear the TIE bit.

1 = TDRE interrupt requests enabled
0 = TDRE interrupt requests disabled

TCIE — Transmission Complete Interrupt Enable

This read/write bit enables SCI interrupt requests when the TC flag becomes set. Resets clear the TCIE bit.

1 = TC interrupt requests enabled
0 = TC interrupt requests disabled

RIE — Receive Interrupt Enable

This read/write bit enables SCI interrupt requests when the RDRF flag or the OR flag becomes set. Resets clear the RIE bit.

1 = RDRF interrupt requests enabled
0 = RDRF interrupt requests disabled

ILIE — Idle Line Interrupt Enable

This read/write bit enables SCI interrupt requests when the IDLE bit becomes set. Resets clear the ILIE bit.

1 = IDLE interrupt requests enabled
0 = IDLE interrupt requests disabled

TE — Transmit Enable

Setting this read/write bit begins the transmission by sending a preamble of 10 or 11 logic ones from the transmit shift register to the PD1/TDO pin. Resets clear the TE bit.

1 = Transmission enabled
0 = Transmission disabled

RE — Receive Enable

Setting this read/write bit enables the receiver. Clearing the RE bit disables the receiver and receiver interrupts but does not affect the receiver interrupt flags. Resets clear the RE bit.

1 = Receiver enabled
0 = Receiver disabled

RWU — Receiver Wakeup Enable

This read/write bit puts the receiver in a standby state. Typically, data transmitted to the receiver clears the RWU bit and returns the receiver to normal operation. The WAKE bit in SCCR1 determines whether an idle input or an address mark brings the receiver out of the standby state. Resets clear the RWU bit.

1 = Standby state
0 = Normal operation

SBK — Send Break

Setting this read/write bit continuously transmits break codes in the form of 10-bit or 11-bit groups of logic zeros. Clearing the SBK bit stops the break codes and transmits a logic one as a start bit. Resets clear the SBK bit.

1 = Break codes being transmitted
0 = No break codes being transmitted

9.4.4 SCI Status Register (SCSR)

The SCI status register, shown in Figure 9-7, contains flags to signal the following conditions:

- Transfer of SCDR data to transmit shift register complete
- Transmission complete
- Transfer of receive shift register data to SCDR complete
- Receiver input idle
- Receiver overrun
- Noisy data
- Framing error

SCSR — SCI Status Register (SCSR) **$0010**

	Bit 7	6	5	4	3	2	1	Bit 0
	TDRE	TC	RDRF	IDLE	OR	NF	FE	—
RESET:	1	1	0	0	0	0	0	—

Figure 9-7. SCI Status Register (SCSR)

TDRE — Transmit Data Register Empty

This clearable, read-only flag is set when the data in the SCDR transfers to the transmit shift register. TDRE generates an interrupt request if the TIE bit in SCCR2 is also set. Clear the TDRE bit by reading the SCSR with TDRE set, and then writing to the SCDR. Resets set the TDRE bit. Software must initialize the TDRE bit to logic zero to avoid an instant interrupt request when turning the transmitter on.

1 = SCDR data transferred to transmit shift register
0 = SCDR data not transferred to transmit shift register

TC — Transmission Complete

This clearable, read-only flag is set when the TDRE bit is set, and no data, preamble, or break character is being transmitted. TC generates an interrupt request if the TCIE bit in SCCR2 is also set. Clear the TC bit by reading the SCSR with TC set, and then writing to the SCDR. Resets set the TC bit. Software must initialize the TC bit to logic zero to avoid an instant interrupt request when turning the transmitter on.

1 = No transmission in progress
0 = Transmission in progress

RDRF — Receive Data Register Full

This clearable, read-only flag is set when the data in the receive shift register transfers to the SCI data register. RDRF generates an interrupt request if the RIE bit in SCCR2 is also set. Clear the RDRF bit by reading the SCSR with RDRF set, and then reading the SCDR.

1 = Received data available in SCDR
0 = Received data not available in SCDR

IDLE — Receiver Idle

This clearable, read-only flag is set when 10 or 11 consecutive logic ones appear on the receiver input. IDLE generates an interrupt request if the ILIE bit in SCCR2 is also set. Clear the IDLE bit by reading the SCSR with IDLE set, and then reading the SCDR.

1 = Receiver input idle
0 = Receiver input not idle

OR — Receiver Overrun

This clearable, read-only flag is set if the SCDR is not read before the receive shift register receives the next word. OR generates an interrupt request if the RIE bit in SCCR2 is also set. The data in the shift register is lost, but the data already in the SCDR is not affected. Clear the OR bit by reading the SCSR with OR set, and then reading the SCDR.

1 = Receive shift register full and RDRF = 1
0 = No receiver overrun

NF — Receiver Noise Flag

This clearable, read-only flag is set when noise is detected in data received in the SCI data register. Clear the NF bit by reading the SCSR, and then reading the SCDR.

1 = Noise detected in SCDR
0 = No noise detected in SCDR

FE — Receiver Framing Error

This clearable, read-only flag is set when there is a logic zero where a stop bit should be in the character shifted into the receive shift register. If the received word causes both a framing error and an overrun error, the OR flag is set and the FE flag is not set. Clear the FE bit by reading the SCSR, and then reading the SCDR.

1 = Framing error
0 = No framing error

9.4.5 Baud Rate Register (BAUD)

The baud rate register, shown in Figure 9-8, selects the baud rate for both the receiver and the transmitter.

BAUD — Baud Rate Register **$000D**

	Bit 7	6	5	4	3	2	1	Bit 0
	—	—	SCP1	SCP0	—	SCR2	SCR1	SCR0
RESET:	—	—	0	0	—	U	U	U

Figure 9-8. Baud Rate Register (BAUD)

SCP1 and SCP0 — SCI Prescaler Select Bits

These read/write bits control prescaling of the baud rate generator clock, as shown in Table 9-1. Resets clear both SCP1 and SCP0.

Table 9-1. Baud Rate Generator Clock Prescaling

SCP[1:0]	Baud Rate Generator Clock
00	Internal Clock ÷ 1
01	Internal Clock ÷ 3
10	Internal Clock ÷ 4
11	Internal Clock ÷ 13

SCR2–SCR0 — SCI Baud Rate Select Bits

These read/write bits select the SCI baud rate, as shown in Table 9-2. Resets have no effect on the SCR2–SCR0 bits.

Table 9-2. Baud Rate Selection

SCR[2:1:0]	SCI Baud Rate (Baud)
000	Prescaled Clock ÷ 1
001	Prescaled Clock ÷ 2
010	Prescaled Clock ÷ 4
011	Prescaled Clock ÷ 8
100	Prescaled Clock ÷ 16
101	Prescaled Clock ÷ 32
110	Prescaled Clock ÷ 64
111	Prescaled Clock ÷ 128

Table 9-3 shows all possible SCI baud rates derived from crystal frequencies of 2 MHz, 4 MHz, and 4.194304 MHz.

Table 9-3. Baud Rate Select Examples

SCP[1:0]	SCR[2:1:0]	SCI Baud Rate fosc = 2 MHz	 fosc = 4 Mhz	 fosc = 4.194304 MHz
00	000	62.50 kbaud	125 kbaud	131.1 kbaud
00	001	31.25 kbaud	62.50 kbaud	65.54 kbaud
00	010	15.63 kbaud	31.25 kbaud	32.77 kbaud
00	011	7813 Baud	15.63 kbaud	16.38 kbaud
00	100	3906 Baud	7813 Baud	8192 Baud
00	101	1953 Baud	3906 Baud	4096 Baud
00	110	976.6 Baud	1953 Baud	2048 Baud
00	111	488.3 Baud	976.6 Baud	1024 Baud
01	000	20.83 kbaud	41.67 kbaud	43.69 kbaud
01	001	10.42 kbaud	20.83 kbaud	21.85 kbaud
01	010	5208 Baud	10.42 kbaud	10.92 kbaud
01	011	2604 Baud	5208 Baud	5461 Baud
01	100	1302 Baud	2604 Baud	2731 Baud
01	101	651.0 Baud	1302 Baud	1365 Baud
01	110	325.5 Baud	651.0 Baud	682.7 Baud
01	111	162.8 Baud	325.5 Baud	341.3 Baud
10	000	15.63 kbaud	31.25 kbaud	32.77 kbaud
10	001	7813 Baud	15.63 kbaud	16.38 kbaud
10	010	3906 Baud	7813 Baud	8192 Baud
10	011	1953 Baud	3906 Baud	4096 Baud
10	100	976.6 Baud	1953 Baud	2048 Baud
10	101	488.3 Baud	976.6 Baud	1024 Baud
10	110	244.1 Baud	488.3 Baud	512.0 Baud
10	111	122.1 Baud	244.1 Baud	256.0 Baud
11	000	4808 Baud	9615 Baud	10.08 kbaud
11	001	2404 Baud	4808 Baud	5041 Baud
11	010	1202 Baud	2404 Baud	2521 Baud
11	011	601.0 Baud	1202 Baud	1260 Baud
11	100	300.5 Baud	601.0 Baud	630.2 Baud
11	101	150.2 Baud	300.5 Baud	315.1 Baud
11	110	75.12 Baud	150.2 Baud	157.5 Baud
11	111	37.56 Baud	75.12 Baud	78.77 Baud

SECTION 10
SERIAL PERIPHERAL INTERFACE (SPI)

This section describes the on-chip, synchronous, serial peripheral interface (SPI).

10.1 Features

Features of the SPI include the following:

- Full Duplex Operation
- Master and Slave Modes
- Four Programmable Master Mode Frequencies (1.05-MHz Maximum)
- 2.1-MHz Maximum Slave Mode Frequency
- Serial Clock with Programmable Polarity and Phase
- End of Transmission Interrupt Flag
- Write Collision Protection
- Bus Contention Protection

Figure 10-1 shows the structure of the SPI subsystem.

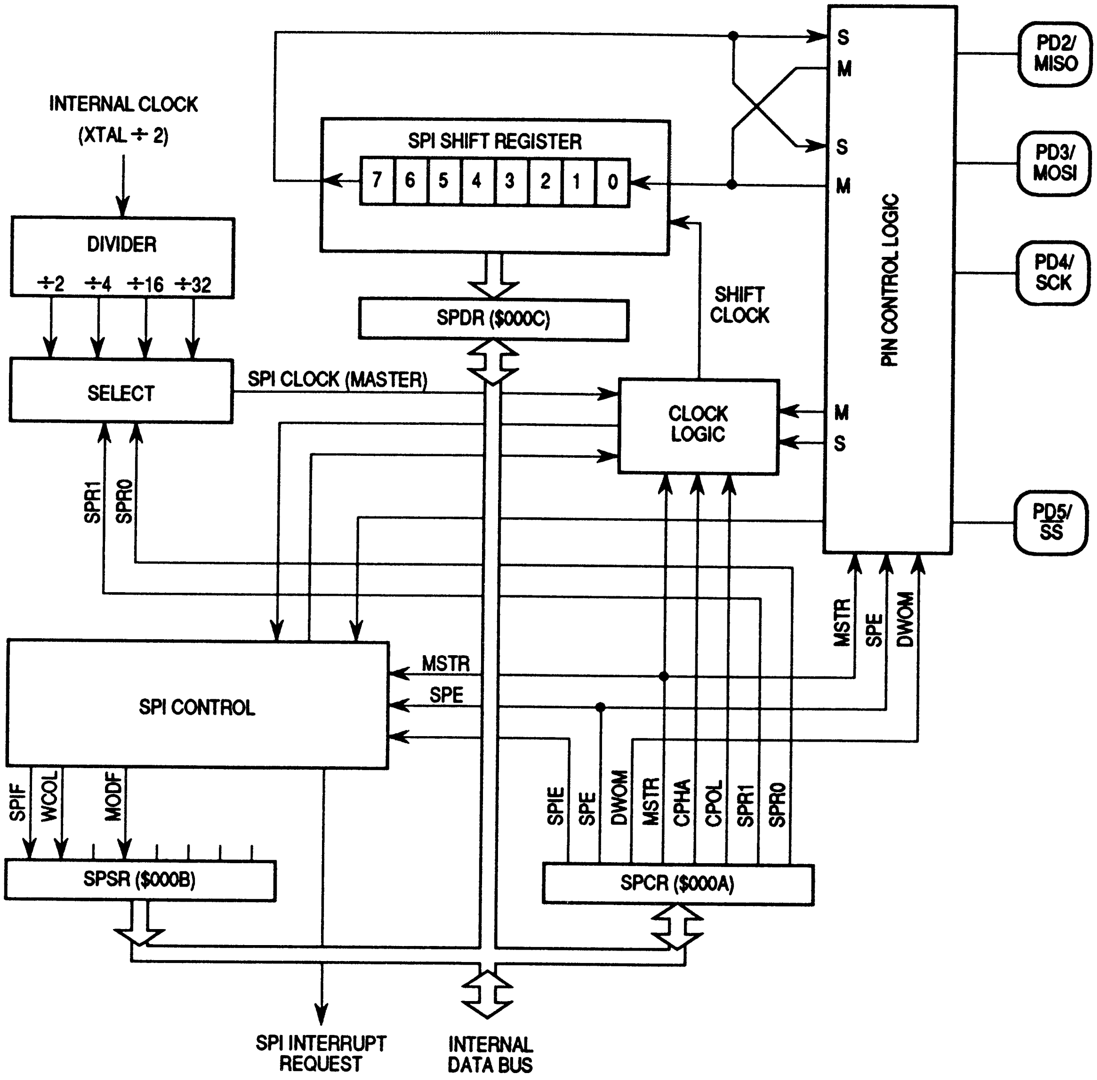


Figure 10-1. SPI Block Diagram

10.2 Operation

The master/slave SPI allows full-duplex, synchronous, serial communication between the MCU and peripheral devices, including other MCUs. As the 8-bit shift register of a master SPI transmits each byte to another device, a byte from the receiving device can enter the master SPI shift register. A clock signal from the master SPI synchronizes data transmission.

Only a master SPI can initiate transmissions. Software begins the transmission from a master SPI by writing to the SPI data register (SPDR). The SPDR does not buffer data being transmitted from the SPI. Data written to the SPDR goes directly into the shift register and begins the transmission immediately under the control of the serial clock. The transmission ends after eight cycles of the serial clock when the SPI flag (SPIF) in the SPI status register (SPSR) becomes set. At the same time that SPIF becomes set, the data shifted into the master SPI from the receiving device transfers to the SPDR. The SPDR buffers data being received by the SPI. Before the master SPI sends the next byte, software must clear the SPIF bit by reading the SPSR and then accessing the SPDR.

In a slave SPI, data enters the shift register under the control of the serial clock from the master SPI. After a byte enters the shift register of a slave SPI, it transfers to the SPDR. To prevent an overrun condition, slave software must then read the byte in the SPDR before another byte enters the shift register and is ready to transfer to the SPDR.

Figure 10-2 shows how a master SPI exchanges data with a slave SPI.

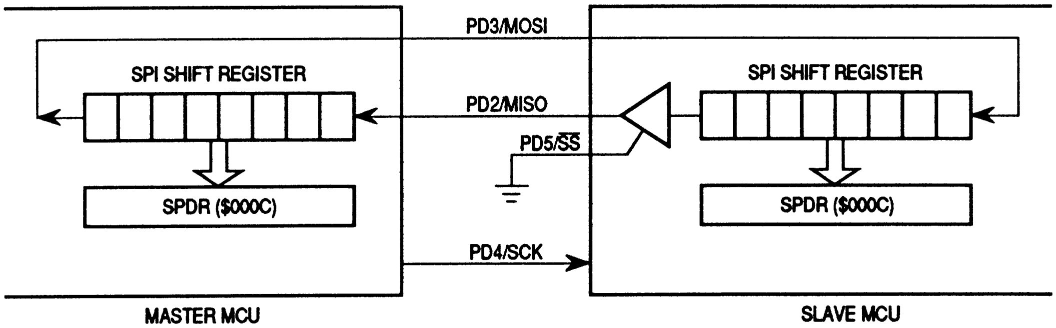


Figure 10-2. Master/Slave Connections

10.2.1 Pin Functions in Master Mode

Setting the MSTR bit in the SPI control register (SPCR) configures the SPI for operation in master mode. The following paragraphs describe the master-mode functions of the SPI pins.

10.2.1.1 PD4/SCK (Serial Clock)

In master mode, the PD4/SCK pin is the synchronizing clock output. Setting the DDR5 bit in data direction register D enables the PD4/SCK output driver.

10.2.1.2 PD3/MOSI (Master Output, Slave Input)

In master mode, the PD3/MOSI pin is the serial output. Setting the DDR3 bit in data direction register D enables the PD3/MOSI output driver.

10.2.1.3 PD2/MISO (Master Input, Slave Output)

In master mode, the PD2/MISO pin is configured as the serial input regardless of the state of the DDR2 bit in data direction register D.

10.2.1.4 PD5/$\overline{\text{SS}}$ (Slave Select)

In master mode, the PD5/$\overline{\text{SS}}$ pin can perform either of the following functions:

- Mode-fault detection — To protect against driver contention caused by the simultaneous operation of two SPIs in master mode, the master SPI monitors the PD5/$\overline{\text{SS}}$ pin for a logic zero. A logic zero on the PD5/$\overline{\text{SS}}$ pin of a master SPI disables the SPI, clears the MSTR bit, and sets the mode-fault flag (MODF). Clearing the DDR5 bit in data direction register D configures pin PD5/$\overline{\text{SS}}$ as an input and enables it as a mode-fault detection pin.

- General-purpose output — Setting the DDR5 bit in data direction register D configures pin PD5/$\overline{\text{SS}}$ as an output. As an output, the PD5/$\overline{\text{SS}}$ pin of a master SPI can function as a general-purpose output pin independent of SPI activity.

10.2.2 Pin Functions in Slave Mode

Clearing the MSTR bit in the SPCR configures the SPI for operation in slave mode. The following paragraphs describe the slave-mode functions of the SPI pins.

10.2.2.1 PD4/SCK (Serial Clock)

In slave mode, the PD4/SCK pin is the input for the synchronizing clock signal from the master SPI. The state of the DDR4 bit in data direction register D has no effect on the PD4/SCK pin of a slave SPI.

10.2.2.2 PD3/MOSI (Master Output, Slave Input)

In slave mode, the PD3/MOSI pin is the serial input regardless of the state of the DDR3 bit in data direction register D.

10.2.2.3 PD2/MISO (Master Input, Slave Output)

In slave mode, the PD2/MISO pin is the serial output. Setting the DDR2 bit in data direction register D enables the PD2/MISO output driver.

10.2.2.4 PD5/$\overline{\text{SS}}$ (Slave Select)

In slave mode, the PD5/$\overline{\text{SS}}$ pin enables the SPI for data and serial clock reception from a master SPI.

10.3 Multiple-SPI Systems

In a multiple-SPI system, all PD4/SCK pins are connected together, all PD3/MOSI pins are connected together, and all PD2/MISO pins are connected together.

Before a transmission, one SPI is configured as master and the rest are configured as slaves. Figure 10-3 is a block diagram showing a single master SPI and three slave SPIs.

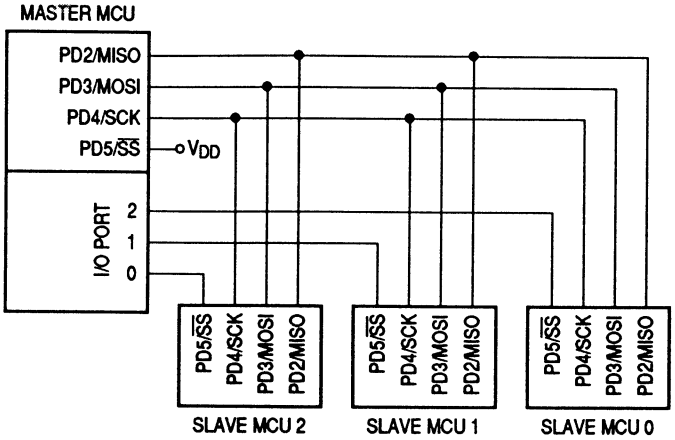


Figure 10-3. One Master and Three Slaves Block Diagram

Figure 10-4 is another block diagram with two master/slave SPIs and three slave SPIs.

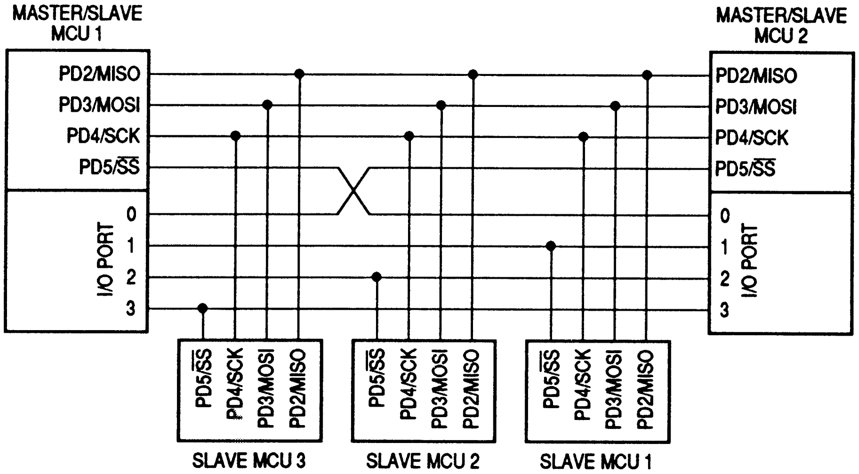


Figure 10-4. Two Master/Slaves and Three Slaves Block Diagram

10.4 Serial Clock Polarity and Phase

To accommodate the different serial communication requirements of peripheral devices, software can change the phase and polarity of the SPI serial clock. The clock polarity bit (CPOL) and the clock phase bit (CPHA), both in the SPCR, control the timing relationship between the serial clock and the transmitted data. Figure 10-5 shows how the CPOL and CPHA bits affect the clock/data timing.

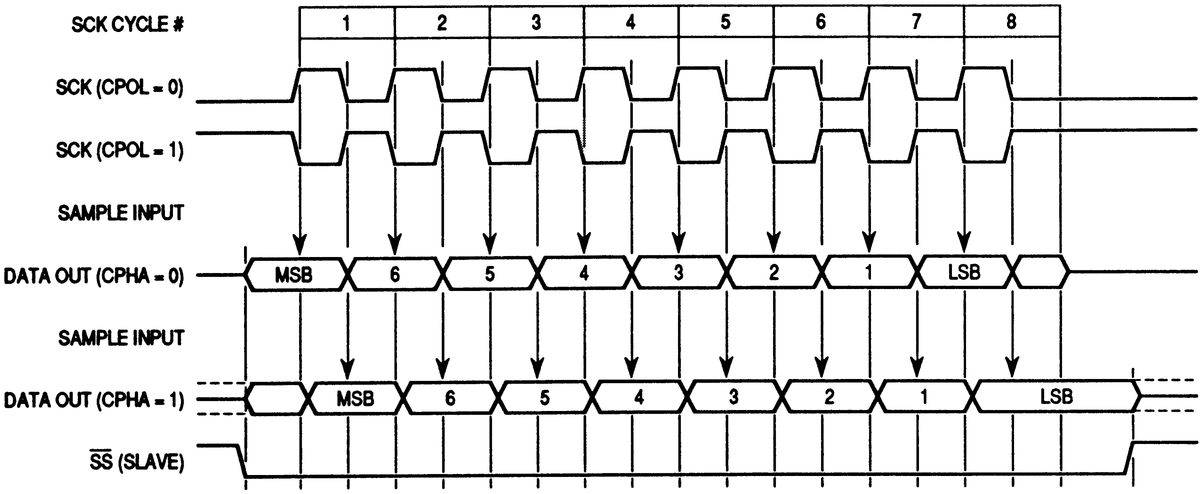


Figure 10-5. SPI Clock/Data Timing

10.5 Wired-OR Outputs

Setting the port D wired-OR mode bit (DWOM) in the SPCR configures all port D outputs, including the SPI pins, as open-drain outputs. The wired-OR mode option requires external pullup resistors on every port D output.

10.6 SPI Error Conditions

The following conditions produce SPI system error conditions:

- Bus contention caused by multiple master SPIs (mode-fault error)
- Writing to the SPDR during a transmission (write-collision error)
- Failing to read the SPDR before the next incoming byte sets the SPIF bit (overflow error)

10.6.1 Mode-Fault Error

A mode-fault error happens when a logic zero occurs on the PD5/$\overline{\text{SS}}$ pin of a master SPI. The MCU takes the following actions when a mode-fault error happens:

- Disables output drivers of SPI outputs by clearing the corresponding DDRD bits
- Puts the SPI in slave mode by clearing the MSTR bit
- Disables the SPI by clearing the SPE bit
- Sets the MODF bit

10.6.2 Write-Collision Error

Writing to the SPDR during a transmission causes a write-collision error and sets the WCOL bit in the SPSR. Either a master SPI or a slave SPI can generate a write-collision error.

- Master — A master SPI can cause a write-collision error by writing to the SPDR while the previously written byte is still being shifted out to the PD3/MOSI pin. The error does not affect the transmission of the previously written byte, but the byte that caused the error is lost.
- Slave — A slave SPI can cause a write-collision error in either of two ways, depending on the state of the CPHA bit:

  a) CPHA = 0 — A slave SPI can cause a write-collision error by writing to the SPDR while the PD5/$\overline{\text{SS}}$ pin is at logic zero. The error does not affect the byte in the SPDR, but the byte that caused the error is lost.

b) CPHA = 1 — A slave SPI can cause a write-collision error by writing to the SPDR while receiving a transmission, that is, between the first active SCK edge and the end of the eighth SCK cycle. The error does not affect the transmission from the master SPI, but the byte that caused the error is lost.

10.6.3 Overflow Error

Failing to read the byte in the SPDR before a subsequent byte enters the shift register causes an overflow condition. In an overflow condition, all incoming data is lost until software clears SPIF. There is no flag for the overflow condition.

10.7 SPI Interrupts

The SPIF bit in the SPSR indicates a byte has shifted into or out of the SPDR. The SPIF bit is a source of SPI interrupt requests. The SPI interrupt enable bit (SPIE) in the SPCR is the local mask for SPIF interrupts.

The MODF bit in the SPSR indicates a mode error and is a source of SPI interrupt requests. The MODF bit is set when a logic zero occurs on the PD5/$\overline{SS}$ pin while the MSTR bit is set and the DDRD5 bit is clear. The SPI interrupt enable bit (SPIE) in the SPCR is the local mask for MODF interrupts.

10.8 SPI I/O Registers

The following I/O registers control and monitor SPI operation:

- SPI Data Register (SPDR)
- SPI Control Register (SPCR)
- SPI Status Register (SPSR)

10.8.1 SPI Data Register (SPDR)

The SPDR, shown in Figure 10-6, is the read buffer for characters received by the SPI. Writing a byte to the SPDR places the byte directly into the SPI shift register.

SPDR — SPI Data Register **$000C**

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

Figure 10-6. SPI Data Register (SPDR)

10.8.2 SPI Control Register (SPCR)

The SPCR, shown in Figure 10-7, has the following functions:

- Enables SPI interrupt requests
- Enables the SPI
- Configures port D pins as open-drain outputs
- Configures the SPI as master or slave
- Selects serial clock polarity, phase, and frequency

SPCR — SPI Control Register **$000A**

	Bit 7	6	5	4	3	2	1	Bit 0
	SPIE	SPE	DWOM	MSTR	CPOL	CPHA	SPR1	SPR0
RESET:	0	0	0	0	U	U	U	U

U = UNAFFECTED

Figure 10-7. SPI Control Register (SPCR)

SPIE — SPI Interrupt Enable

This read/write bit enables SPI interrupts. Resets clear the SPIE bit.

1 = SPI interrupts enabled
0 = SPI interrupts disabled

SPE — SPI Enable

This read/write bit enables the SPI. Resets clear the SPE bit.

1 = SPI enabled
0 = SPI disabled

DWOM — Port D Wired-OR Mode

This read/write bit disables the pullup transistors on all port D outputs so that port D outputs become open-drain drivers.

1 = Wired-OR mode port D outputs
0 = Normal push-pull port D outputs

MSTR — Master

This read/write bit selects master mode operation or slave mode operation. Resets clear the MSTR bit.

1 = Master mode
0 = Slave mode

CPOL — Clock Polarity

This read/write bit determines the logic state of the PD4/SCK pin between transmissions. To transmit data between SPIs, the SPIs must have identical CPOL bits. Resets have no effect on the CPOL bit.

1 = PD4/SCK pin at logic one between transmissions
0 = PD4/SCK pin at logic zero between transmissions

CPHA — Clock Phase

This read/write bit controls the timing relationship between the serial clock and SPI data. To transmit data between SPIs, the SPIs must have identical CPHA bits. When CPHA = 0, the PD5/$\overline{\text{SS}}$ pin of the slave SPI must be set to logic one between bytes. See Figure 10-5. Reset has no effect on the CPHA bit.

1 = Edge following first active edge on PD4/SCK latches data
0 = First active edge on PD4/SCK latches data

SPR1 and SPR0 — SPI Clock Rate

These read/write bits select master mode serial clock rate, as shown in Table 10-1. The SPR1 and SPR0 bits of a slave SPI have no effect on the serial clock.

Table 10-1. SPI Clock Rate Selection

SPR[1:0]	SPI Clock Rate
00	Internal Clock ÷ 2
01	Internal Clock ÷ 4
10	Internal Clock ÷ 16
11	Internal Clock ÷ 32

10.8.3 SPI Status Register (SPSR)

The SPSR, shown in Figure 10-8, contains flags to signal the following conditions:

- SPI transmission complete
- Write collision
- Mode fault

SPSR — SPI Status Register **$000B**

	Bit 7	6	5	4	3	2	1	Bit 0
	SPIF	WCOL	—	MODF	—	—	—	—
RESET:	0	0	0	0	—	—	—	—

Figure 10-8. SPI Status Register (SPSR)

SPIF — SPI Flag

This clearable, read-only flag is set each time a byte shifts out of or into the shift register. SPIF generates an interrupt request if the SPIE bit in the SPCR is also set. Clear the SPIF bit by reading the SPSR with SPIF set, and then reading or writing the SPDR. Resets clear the SPIF bit.

1 = Transmission complete
0 = Transmission not complete

WCOL — Write Collision

This clearable, read-only flag is set when software writes to the SPDR while a transmission is in progress. Clear the WCOL flag by reading the SPSR with WCOL set, and then reading or writing the SPDR. Resets clear the WCOL flag.

1 = Invalid write to SPDR
0 = No invalid write to SPDR

MODF — Mode Fault

This clearable, read-only flag is set when a logic zero occurs on the PD5/$\overline{SS}$ pin while the MSTR bit is set and bit DDRD5 is clear. MODF generates an interrupt request if the SPIE bit is also set. Clear the MODF flag by reading the SPSR with MODF set, and then writing the SPCR. Resets clear the MODF flag.

1 = PD5/$\overline{SS}$ pulled low while MSTR bit set
0 = PD5/$\overline{SS}$ not pulled low while MSTR bit set

SECTION 11
SELF-CHECK MODE

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

11.1 Self Check

The self-check function determines if the MCU is functioning properly. The self-check circuit is shown in Figure 11-1. If 9 Vdc is applied to the $\overline{\text{IRQ}}$ pin, and a logic one is applied to the TCAP pin, the MCU enters the self-check mode on reset. Port C pins PC3–PC0 are monitored for the self-check results. After a reset in self-check mode, the following seven self-check tests are performed automatically:

1. I/O — Functional test of ports A, B, and C
2. RAM — Counter test for each RAM byte
3. Timer — Test of counter register and OCF bit
4. SCI — Transmission test: checks for RDRF, TDRE, TC, and FE flags
5. ROM — Exclusive OR with odd ones parity result
6. SPI — Transmission test: checks for SPIF and WCOL
7. Interrupts — Tests external, timer, SCI, and SPI interrupts

Table 11-1 shows the light-emitting diode codes for self-check results.

Table 11-1. Self-Check Results

PC3	PC2	PC1	PC0	Remarks
1	0	0	1	Bad I/O
1	0	1	0	Bad RAM
1	0	1	1	Bad Timer
1	1	0	0	Bad SCI
1	1	0	1	Bad ROM
1	1	1	0	Bad SPI
1	1	1	1	Bad Interrupts or $\overline{\text{IRQ}}$ Request
Flashing				Good Device
All Others				Bad Device, Bad Port C, etc.

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

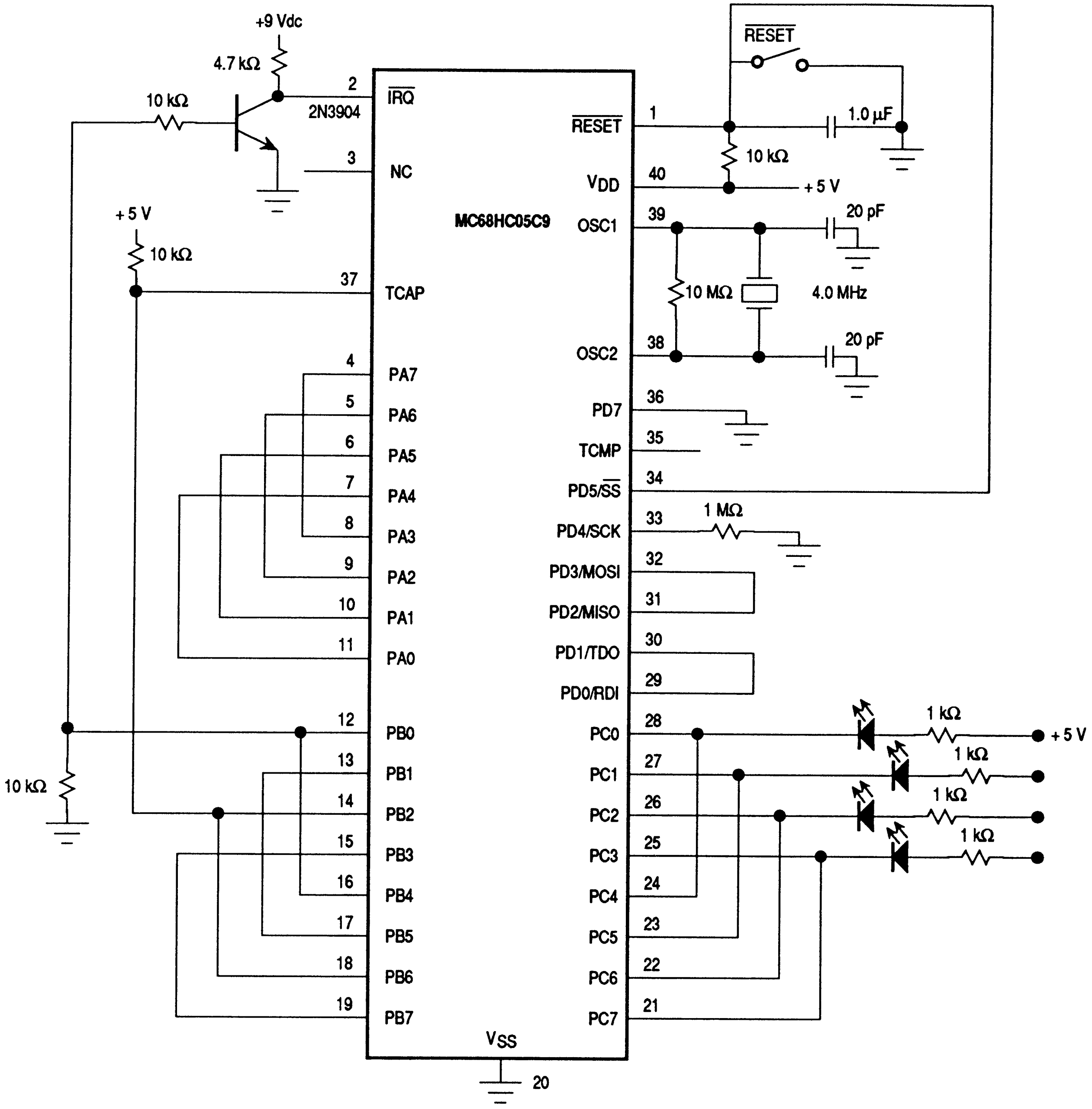


Figure 11-1. Self-Check Circuit Schematic

SECTION 12
INSTRUCTION SET

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

12.1 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

12.1.1 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 12-1 lists the instructions that use inherent addressing.

Table 12-1. 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

12.1.2 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 12-2 lists the instructions that use immediate addressing.

Table 12-2. 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

12.1.3 Direct

Direct instructions can access any of the first 256 memory addresses with only 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 12-3 lists the instructions that use direct addressing.

Table 12-3. 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

12.1.4 Extended

Extended instructions can access any address in memory with only three bytes. 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 12-4 lists the instructions that use the extended addressing mode.

Table 12-4. 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

12.1.5 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 12-5 lists the instructions that use indexed, no offset addressing.

12.1.6 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 12-5 lists the instructions that use indexed, 8-bit offset addressing.

12.1.7 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 12-5 lists the instructions that use indexed, 16-bit offset addressing.

Table 12-5. 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	√	√	

12.1.8 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 12-6 lists the instructions that use relative addressing.

Table 12-6. 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

12.2 Instruction Set

The MCU instructions fall into the following five categories:

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

12.2.1 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. Most register/memory instructions use the following addressing modes:

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

Table 12-7 lists the register/memory instructions.

Table 12-7. 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

12.2.2 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 12-8 lists the read-modify-write instructions.

Table 12-8. 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

12.2.3 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 12-9 lists the jump and branch instructions.

Table 12-9. 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

12.2.4 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 12-10 lists these instructions.

Table 12-10. Bit Manipulation Instructions

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

12.2.5 Control Instructions

These register reference instructions control CPU operation during program execution. Control instructions, listed in Table 12-11, use inherent addressing.

Table 12-11. 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

12.2.6 Instruction Set Summary and Opcode Map

Table 12-12 shows all MC68HC05C9 instructions in all possible addressing modes.

Table 12-13 is an opcode map of the M68HC05 instruction set.

For each instruction, the operand construction and the execution time in internal clock cycles (t_{CYC}) are shown. One internal clock cycle equals two oscillator input cycles. The following legend summarizes the symbols and abbreviations used in Table 12-12.

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 mode
dd	Address of operand in direct add ressing	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 mode	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 mode	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 mode	←	Is loaded with
INH	Inherent addressing mode	:	Concatenated with
IX	Indexed, no offset addressing mode	×	Multiplication
IX1	Indexed, 8-bit offset addressing mode	–( )	Negation (two's complement)
IX2	Indexed, 16-bit offset addressing mode	+	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 12-12. Instruction Set (Sheet 1 of 4)

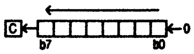
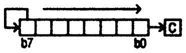
Source Form(s)	Operation	Description	Addressing Mode for Operand	Machine Coding (hexadecimal)		Cycles	Condition Code				
				Opcode	Operand		H	I	N	Z	C
ADC opr	Add with carry	A ← (A) + (M) + C	IMM DIR EXT IX2 IX1 IX	A9 B9 C9 D9 E9 F9	ii dd hh ll ee ff ff	2 3 4 5 4 3	↕	–	↕	↕	↕
ADD opr	Add without carry	A ← (A) + (M)	IMM DIR EXT IX2 IX1 IX	AB BB CB DB EB FB	ii dd hh ll ee ff ff	2 3 4 5 4 3	↕	–	↕	↕	↕
AND opr	Logical AND	A ← (A) • (M)	IMM DIR EXT IX2 IX1 IX	A4 B4 C4 D4 E4 F4	ii dd hh ll ee ff ff	2 3 4 5 4 3	–	–	↕	↕	–
ASL opr ASLA ASLX ASL opr ASL opr	Arithmetic shift left		DIR INH INH IX1 IX	38 48 58 68 78	dd ff	5 3 3 6 5	–	–	↕	↕	↕
ASR opr ASRA ASRX ASR opr ASR opr	Arithmetic shift right		DIR INH INH IX1 IX	37 47 57 67 77	dd ff	5 3 3 6 5	–	–	↕	↕	↕
BCC rel	Branch if carry bit clear	? C = 0	REL	24	rr	3	–	–	–	–	–
BCLR n opr	Clear bit n	Mn ← 0	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	11 13 15 17 19 1B 1D 1F	dd dd dd dd dd dd dd dd	5 5 5 5 5 5 5 5	–	–	–	–	–
BCS rel	Branch if carry bit set	? 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 $\overline{\text{IRQ}}$ pin high	? $\overline{\text{IRQ}}$ = 1	REL	2F	rr	3	–	–	–	–	–
BIL rel	Branch if $\overline{\text{IRQ}}$ pin low	? $\overline{\text{IRQ}}$ = 0	REL	2E	rr	3	–	–	–	–	–
BIT rel	Bit test accumulator contents with memory contents	(A) • (M)	IMM DIR EXT IX2 IX1 IX	A5 B5 C5 D5 E5 F5	ii dd hh ll ee ff ff	2 3 4 5 4 3	–	–	↕	↕	–
BLO rel	Branch if lower	? 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 12-12. 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	$M \leftarrow \overline{M} = \$FF - (M)$	DIR	33	dd	5	–	–	↕	↕	1
COMA		$A \leftarrow \overline{A} = \$FF - (A)$	INH	43		3					
COMX	(one's complement)	$X \leftarrow \overline{X} = \$FF - (X)$	INH	53		3					
COM opr		$M \leftarrow \overline{M} = \$FF - (M)$	IX1	63	ff	6					
COM opr		$M \leftarrow \overline{M} = \$FF - (M)$	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 12-12. Instruction Set (Sheet 3 of 4)

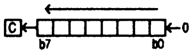
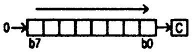
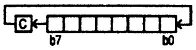
Source Form(s)	Operation	Description	Addressing Mode for Operand	Machine Coding (hexadecimal)		Cycles	Condition Code				
				Opcode	Operand		H	I	N	Z	C
EOR opr	Exclusive OR accumulator contents with memory contents	A ← (A) ⊕ (M)	IMM	A8	ii	2	–	–	↕	↕	–
			DIR	B8	dd	3					
			EXT	C8	hh ll	4					
			IX2	D8	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 (two's 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 12-12. Instruction Set (Sheet 4 of 4)

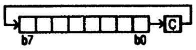
Source Form(s)	Operation	Description	Addressing Mode for Operand	Machine Coding (hexadecimal)		Cycles	Condition Code				
				Opcode	Operand		H	I	N	Z	C
ROR opr	Rotate right through carry		DIR	36	dd	5	–	–	↕	↕	↕
RORA			INH	46		3					
RORX			INH	56		3					
ROR opr			IX1	66	ff	6					
ROR opr			IX	76		5					
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	A2	ii	2	–	–	↕	↕	↕
			DIR	B2	dd	3					
			EXT	C2	hh ll	4					
			IX2	D2	ee ff	5					
			IX1	E2	ff	4					
			IX	F2		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	B7	dd	4	–	–	↕	↕	–
			EXT	C7	hh ll	5					
			IX2	D7	ee ff	6					
			IX1	E7	ff	5					
			IX	F7		4					
STOP	Enable $\overline{IRQ}$; stop oscillator		INH	8E		2	–	0	–	–	–
STX opr	Store index register contents in memory	M ← (X)	DIR	BF	dd	4	–	–	↕	↕	–
			EXT	CF	hh ll	5					
			IX2	DF	ee ff	6					
			IX1	EF	ff	5					
			IX	FF		4					
SUB opr	Subtract memory contents from accumulator contents	A ← (A) – (M)	IMM	A0	ii	2	–	–	↕	↕	↕
			DIR	B0	dd	3					
			EXT	C0	hh ll	4					
			IX2	D0	ee ff	5					
			IX1	E0	ff	4					
			IX	F0		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	Test memory, accumulator, or index register contents for negative or zero	(M) – $00	DIR	3D	dd	4	–	–	↕	↕	–
TSTA			INH	4D		3					
TSTX			INH	5D		3					
TST opr			IX1	6D	ff	5					
TST opr			IX	7D		4					
TXA	Transfer index register contents to accumulator	A ← (X)	INH	9F		2	–	–	–	–	–
WAIT	Enable interrupts; halt CPU		INH	8F		2	–	0	–	–	–

Table 12-13. 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 13
ELECTRICAL SPECIFICATIONS

This section contains parametric and timing information.

13.1 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 13-1. Keep V_{in} and V_{out} within the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{DD}$. Connect unused inputs to the appropriate logical voltage level, either V_{SS} or V_{DD}.

Table 13-1. Maximum Ratings

Rating	Symbol	Value	Unit
Supply Voltage	V_{DD}	–0.3 to +7.0	V
Input Voltage All Pins in Normal Operation	V_{in}	V_{SS} – 0.3 to V_{DD} + 0.3	V
Self-Check Mode ($\overline{IRQ}$ Pin Only)		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 MC68HC05C9P, FN MC68HC05C9CP, CFN, CB, CFB MC68HC05C9B, FB	T_A	T_L to T_H 0 to +70 –40 to +85 0 to +70	°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°)
4. B = Shrink dual-in-line package (SDIP)
5. FB = Quad flat pack (QFP)

13.2 Thermal Characteristics

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic Dual-In-Line Package (PDIP) Plastic Leaded Chip Carrier (PLCC) Quad Flat Pack (QFP) Plastic Shrink DIP (SDIP)	θ_{JA}	 60 70 95 60	°C/W

13.3 Power Considerations

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

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

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.

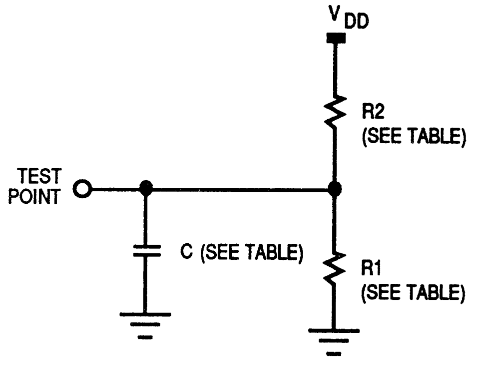
The following is an approximate relationship between P_D and T_J (neglecting $P_{I/O}$):

$$P_D = K \div (T_J + 273\ °C) \tag{2}$$

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

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

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



V_{DD} = 4.5 V

Pins	R1	R2	C
PA7–PA0 PB7–PB0 PC7–PC0 PD5–PD0, PD7	3.26 kΩ	2.38 kΩ	50 pF

V_{DD} = 3.0 V

Pins	R1	R2	C
PA7–PA0 PB7–PB0 PC7–PC0 PD5–PD0, PD7	10.91 kΩ	6.32 kΩ	50 pF

Figure 13-1. Test Load

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

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

Table 13-2. DC Electrical Characteristics (V_{DD} = 5.0 Vdc)

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage, $I_{LOAD} \leq 10.0\ \mu A$	V_{OL} V_{OH}	— $V_{DD} - 0.1$	— —	0.1 —	V
Output High Voltage (Iload = 0.8 mA) PA7–PA0, PB7–PB0, PC7–PC0, PD7, PD5–PD0, TCMP	V_{OH}	$V_{DD} - 0.8$	— —	— —	V
Output Low Voltage (I_{LOAD} = 1.6 mA) PA7–PA0, PB7–PB0, PC7–PC0, PD7, PD5–PD0, TCMP	V_{OL}	—	—	0.4	V
Input High Voltage PA7–PA0, PB7–PB0, PC7–PC0, PD7, PD5–PD0, TCAP, $\overline{\text{IRQ}}$, $\overline{\text{RESET}}$, OSC1	V_{IH}	$0.7 \times V_{DD}$	—	V_{DD}	V
Input Low Voltage PA7–PA0, PB7–PB0, PC7–PC0, PD7, PD5–PD0, TCAP, $\overline{\text{IRQ}}$, $\overline{\text{RESET}}$, OSC1	V_{IL}	V_{SS}	—	$0.2 \times V_{DD}$	V
Data Retention Mode (0° to 70° C)	V_{RM}	2.0	—	—	V
Supply Current (Refer to NOTES)	I_{DD}				
Run		—	5	10.0	mA
Wait		—	1.6	4.0	mA
Stop					
25 °C		—	2.0	50	µA
0°C to +70 °C (Standard)		—	—	140	µA
–40° to +85 °C		—		180	µA
I/O Ports High-Z Leakage Current PA7–PA0, PB7–PB0, PC7–PC0, PD7, PD5–PD0, $\overline{\text{RESET}}$	I_{IL}	—	—	±10	µA
Input Current $\overline{\text{IRQ}}$, TCAP, OSC1	I_{in}	—	—	±1	µA
Capacitance Ports (as Input or Output) $\overline{\text{RESET}}$, $\overline{\text{IRQ}}$, TCAP, PD7, PD5–PD0	C_{OUT} C_{IN}	— —	— —	12 8	pF

NOTES:

1. All values shown reflect average measurements.
2. Typical values at midpoint of voltage range, 25 °C only.
3. Wait I_{DD}: only timer system active (SPE = TE = RE = 0). If SPI, SCI active (SPE = TE = RE = 1) add 10% current draw.
4. Run (operating) I_{DD} and Wait I_{DD} measured using external square wave clock source (f_{OSC} = 4.2 MHz); all inputs 0.2 V from rail; no dc loads; less than 50 pF on all outputs; C_L = 20 pF on OSC2.
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. Stop I_{DD} measured with OSC1 = V_{SS}.
7. Standard temperature range is 0 °C to +70 °C. Extended temperature (–40 °C to +85 °C) is available.
8. Wait I_{DD} is affected linearly by the OSC2 capacitance.

13.5 DC Electrical Characteristics (V_{DD} = 3.3 Vdc)

(V_{DD} = 3.3 Vdc ± 0.3 Vdc, V_{SS} = 0 Vdc , $T_A = T_L$ to T_H, unless otherwise noted.)

Table 13-3. DC Electrical Characteristics (V_{DD} = 3.3 Vdc)

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage, $I_{LOAD} \leq 10.0$ µA	V_{OL} V_{OH}	— $V_{DD} - 0.1$	— —	0.1 —	V
Output High Voltage (Iload = 0.2 mA) PA7–PA0, PB7–PB0, PC7–PC0, PD7, PD0–PD5, TCMP	V_{OH}	$V_{DD} - 0.3$	— —	— —	V
Output Low Voltage (I_{LOAD} = 1.6 mA) PA7–PA0, PB7–PB0, PC7–PC0, PD7, PD5–PD0, TCMP	V_{OL}	—	—	0.3	V
Input High Voltage PA7–PA0, PB7–PB0, PC7–PC0, PD7, PD5–PD0, TCAP, $\overline{\text{IRQ}}$, $\overline{\text{RESET}}$, OSC1	V_{IH}	$0.7 \times V_{DD}$	—	V_{DD}	V
Input Low Voltage PA7–PA0, PB7–PB0, PC7–PC0, PD7, PD5–PD0, TCAP, $\overline{\text{IRQ}}$, $\overline{\text{RESET}}$, OSC1	V_{IL}	V_{SS}	—	$0.2 \times V_{DD}$	V
Data Retention Mode (0° to 70° C)	V_{RM}	2.0	—	—	V
Supply Current (Refer to NOTES)	I_{DD}				
Run		—	2	4.0	mA
Wait		—	0.5	1.4	mA
Stop					
25 °C		—	1.0	30	µA
0 °C to +70 °C (Standard)		—	—	80	µA
–40 °C to +85 °C		—		120	µA
I/O Ports High-Z Leakage Current PA7–PA0, PB7–PB0, PC7–PC0, PD7, PD5–PD0, $\overline{\text{RESET}}$	I_{IL}	—	—	±10	µA
Input Current $\overline{\text{IRQ}}$, TCAP, OSC1	I_{in}	—	—	±1	µA

NOTES:

1. All values shown reflect average measurements.
2. Typical values at midpoint of voltage range, 25 °C only.
3. Wait I_{DD}: only timer system active (SPE = TE = RE = 0). If SPI, SCI active (SPE = TE = RE = 1) add 10% current draw.
4. Run (operating) I_{DD} and Wait I_{DD} measured using external square wave clock source (f_{OSC} = 4.2 MHz); all inputs 0.2 V from rail; no dc loads; less than 50 pF on all outputs; C_L = 20 pF on OSC2.
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. Stop I_{DD} measured with OSC1 = V_{SS}.
7. Standard temperature range is 0 °C to +70 °C. Extended temperature (–40 °C to +85 °C) is available.
8. Wait I_{DD} is affected linearly by the OSC2 capacitance.

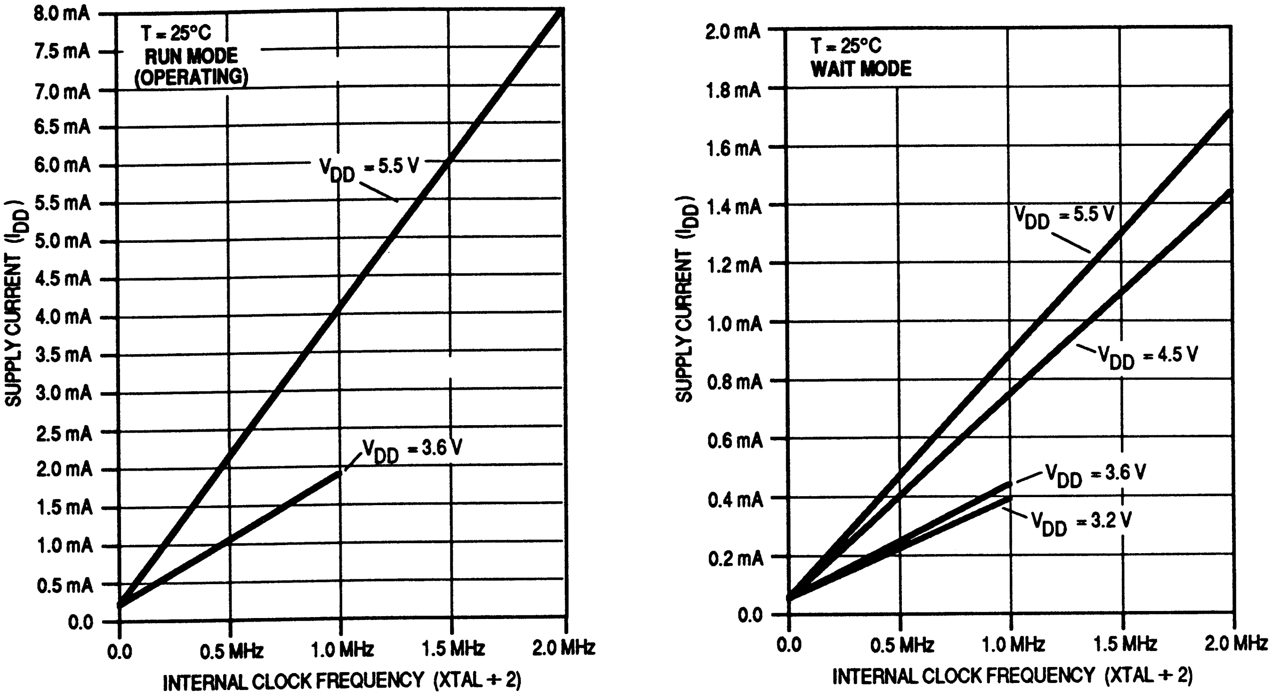


Figure 13-2. Typical Supply Current vs Internal Clock Frequency

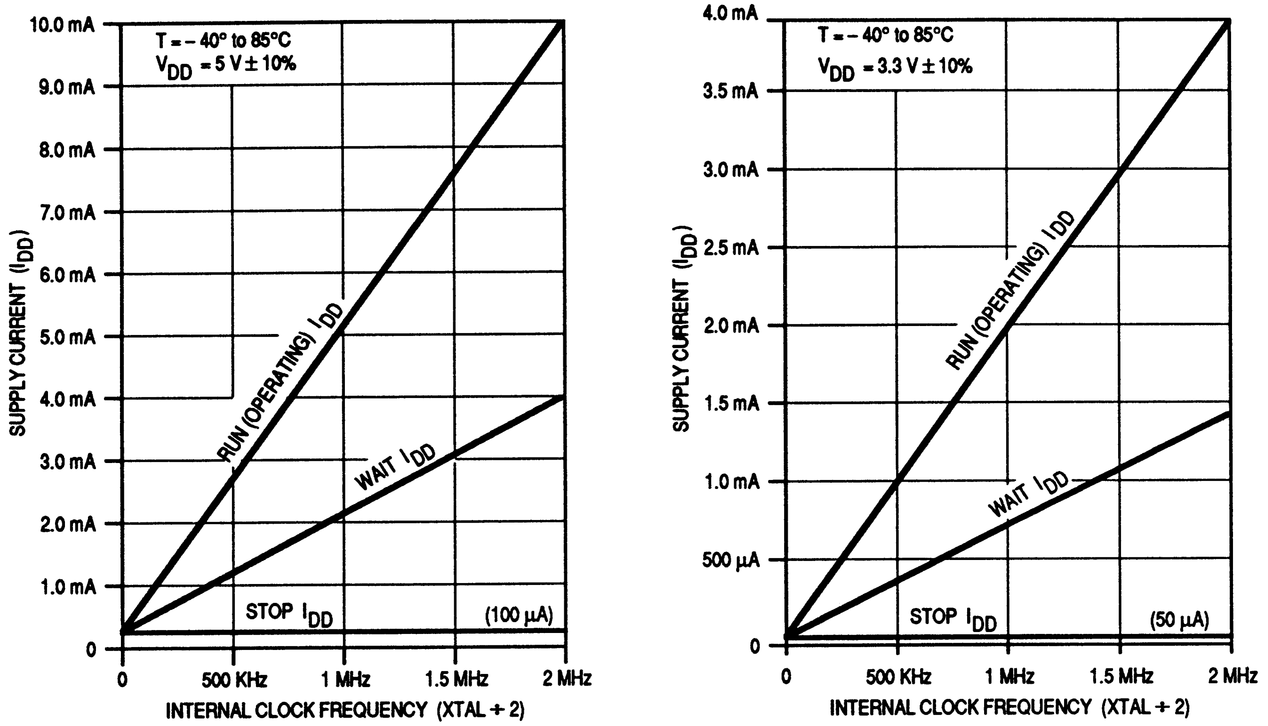


Figure 13-3. Maximum Supply Current vs Clock Frequency

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

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

Table 13-4. Control Timing (V_{DD} = 5.0 Vdc)

Characteristic	Symbol	Min	Max	Unit
Oscillator Frequency Crystal Option External Clock Option	f_{OSC}	— dc	4.2 4.2	MHz
Internal Operating Frequency Crystal ($f_{OSC} \div 2$) External Clock ($f_{OSC} \div 2$)	f_{OP}	— dc	2.1 2.1	MHz
Internal Clock Cycle Time	t_{CYC}	480	—	ns
$\overline{\text{RESET}}$ Pulse Width	t_{RL}	1.5	—	t_{CYC}
Capture/Compare Timer Resolution (NOTE 1) Input Capture Pulse Width Input Capture Pulse Period	 t_{RESL} t_{TH}, t_{TL} t_{TLTL}	 4.0 125 (NOTE 2)	 — — —	 t_{CYC} ns 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

NOTES:

1. Because a 2-bit prescaler in the capture/compare timer must count four internal cycles (t_{CYC}), this is the limiting minimum factor in determining the timer resolution.
2. The minimum period t_{TLTL} should not be less than the number of cycle times it takes to execute the capture interrupt service routine plus 24 t_{CYC}.
3. The minimum period t_{TLTL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus 21 t_{CYC}.

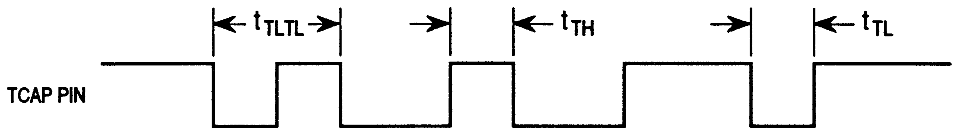


Figure 13-4. TCAP Timing Relationships

13.7 Control Timing (V_{DD} = 3.3 Vdc)

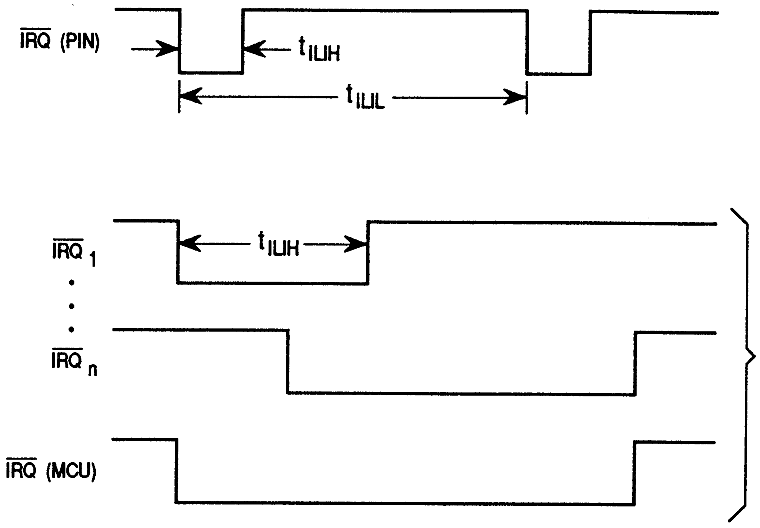
(V_{DD} = 3.3 Vdc ± 10%, V_{SS} = 0 Vdc , $T_A = T_L$ to T_H)

Table 13-5. Control Timing (V_{DD} = 3.3 Vdc)

Characteristic	Symbol	Min	Max	Unit
Oscillator Frequency Crystal Option External Clock Option	f_{OSC}	— dc	2.0 2.0	MHz
Internal Operating Frequency Crystal (f_{OSC} ÷ 2) External Clock (f_{OSC} ÷ 2)	f_{OP}	— dc	1.0 1.0	MHz
Internal Clock Cycle Time	t_{CYC}	1000	—	ns
$\overline{\text{RESET}}$ Pulse Width, Excluding Power-Up	t_{RL}	1.5	—	t_{CYC}
Capture/Compare Timer Resolution (NOTE 1) Input Capture Pulse Width Input Capture Pulse Period	 t_{RESL} t_{TH}, t_{TL} t_{TLTL}	 4.0 250 (NOTE 2)	 — — —	 t_{CYC} ns t_{CYC}
Interrupt Pulse Width Low (Edge-Triggered)	t_{ILIH}	250	—	ns
Interrupt Pulse Period	t_{ILIL}	(NOTE 3)	—	t_{CYC}
OSC1 Pulse Width	t_{OH}, t_{OL}	200	—	ns

NOTES:

1. Because a 2-bit prescaler in the capture/compare timer must count four internal cycles (t_{CYC}), this is the limiting minimum factor in determining the timer resolution.
2. The minimum period t_{TLTL} should not be less than the number of cycle times it takes to execute the capture interrupt service routine plus 24 t_{CYC}.
3. The minimum period t_{TLTL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus 21 t_{CYC}.

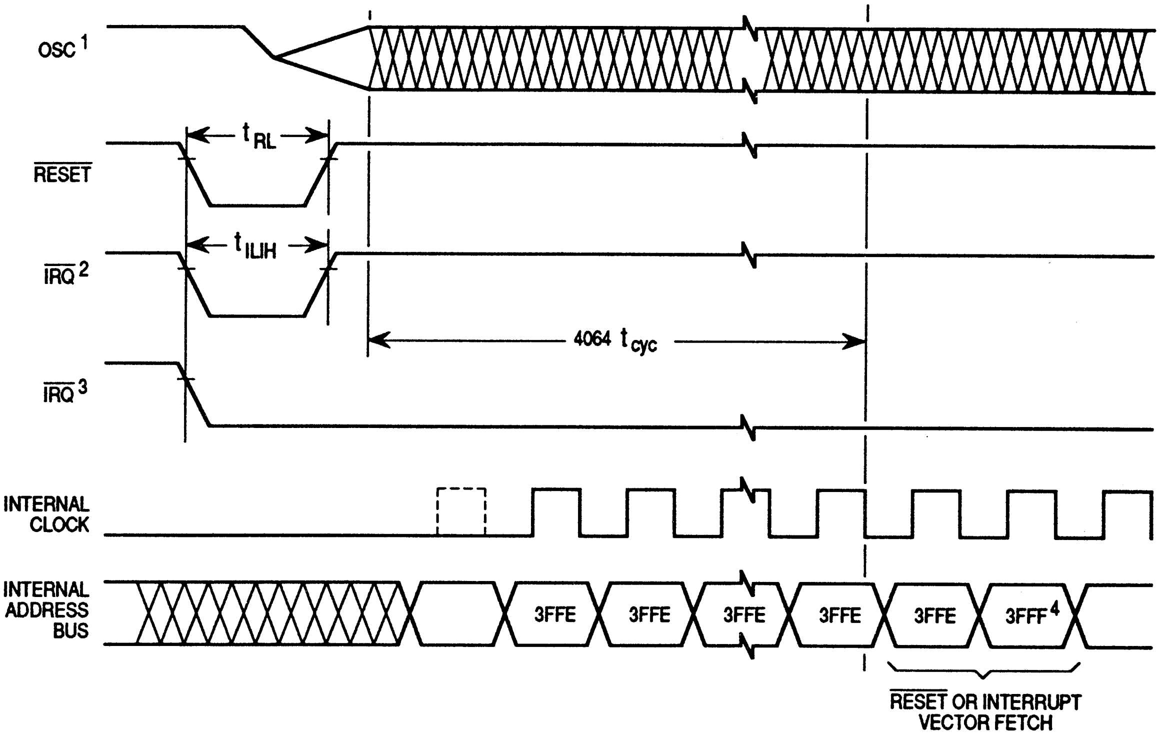


Edge-Sensitive Trigger Condition
The minimum pulse width (t_{ILIH}) is either 125 ns (V_{DD}= 5 V) or 250 ns (V_{DD} = 3 V). The period t_{ILIL} should not be less than the number of t_{CYC} cycles it takes to execute the interrupt service routine plus 19 t_{CYC} cycles.

Level-Sensitive Trigger Condition
If after servicing an interrupt the IRQ remains low, then the next interrupt is recognized.

NORMALLY USED WITH WIRED–OR CONNECTION

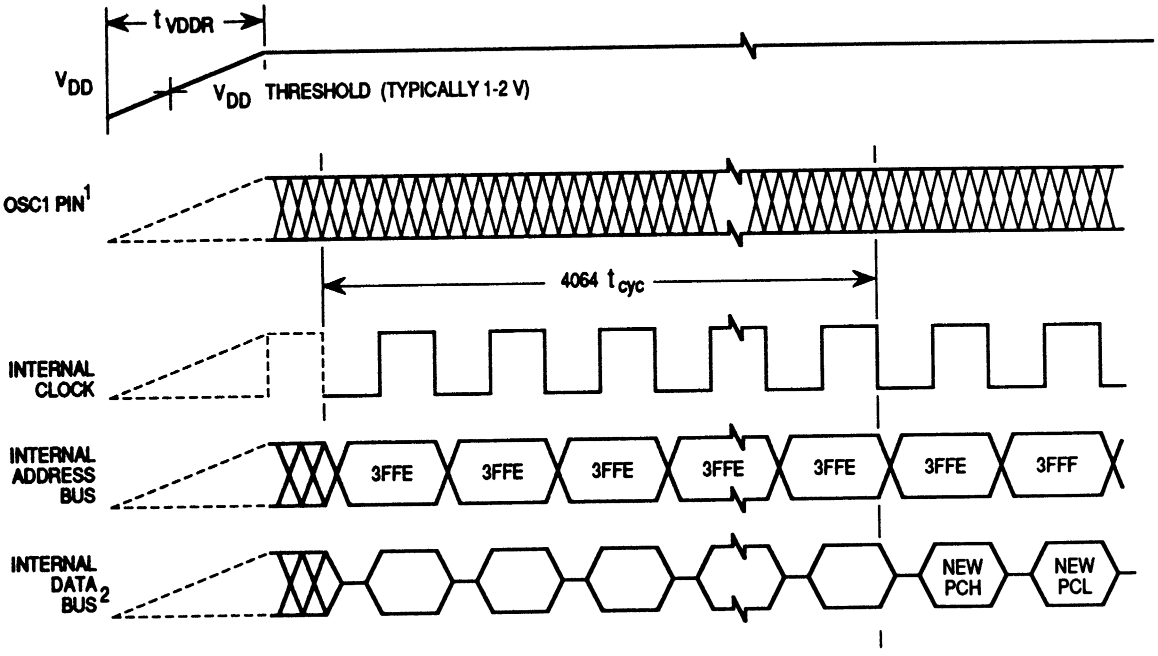
Figure 13-5. External Interrupt Timing



NOTES:

1. Represents the internal clocking of the OSC1 pin
2. IRQ pin edge–sensitive
3. IRQ pin level– and edge–sensitive
4. RESET vector address shown for timing example

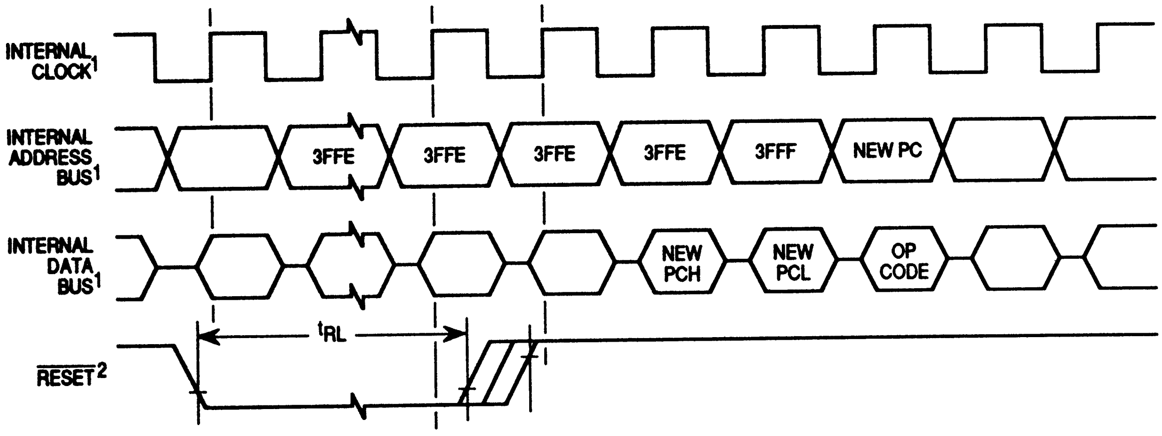
Figure 13-6. STOP Recovery Timing Diagram



NOTES:

1. OSC1 line is meant to represent time only, not frequency.
2. Internal clock, internal address bus, and internal data bus signals are not available externally.

Figure 13-7. 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 13-8. External Reset Timing

13.8 Serial Peripheral Interface (SPI) Timing

Table 13-6. Serial Peripheral Interface Timing (5.0 Vdc)

V_{DD} = 5.0 Vdc ± 10%; V_{SS} = 0 Vdc, $T_A = T_L$ to T_H (Refer to Figures 13-11 and 13-12.)

Num	Characteristic	Symbol	Min	Max	Unit
	Operating Frequency Master Slave	$f_{OP(M)}$ $f_{OP(S)}$	dc dc	0.5 2.1	f_{OP} MHz
1	Cycle Time Master Slave	$t_{CYC(M)}$ $t_{CYC(S)}$	2.0 480	— —	t_{CYC} ns
2	Enable Lead Time Master Slave	$t_{LEAD\ (M)}$ $t_{LEAD\ (S)}$	* 240	— —	ns ns
3	Enable Lag Time Master Slave	$t_{LAG\ (M)}$ $t_{LAG\ (S)}$	* 720	— —	ns ns
4	Clock (SCK) High Time Master Slave	$t_{W(SCKH)M}$ $t_{W(SCKH)S}$	340 190	— —	ns ns
5	Clock (SCK) Low Time Master Slave	$t_{W(SCKL)M}$ $t_{W(SCKL)S}$	340 190	— —	ns ns
6	Data Setup Time (Inputs) Master Slave	$t_{SU(M)}$ $t_{SU(S)}$	100 100	— —	ns ns
7	Data Hold Time (Inputs) Master Slave	$t_{H(M)}$ $t_{H(S)}$	100 100	— —	ns ns
8	Slave Access Time (Time to Data Active from High-Impedance State)	t_A	0	120	ns
9	Slave Disable Time (Hold Time to High-Impedance State)	t_{DIS}	—	240	ns
10	Data Valid Master (Before Capture Edge) Slave (After Enable Edge)**	$t_{V(M)}$ $t_{V(S)}$	0.25 —	— 240	$t_{CYC(M)}$ ns
11	Data Hold Time (Outputs) Master (After Capture Edge) Slave (After Enable Edge)	$t_{HO\ (M)}$ $t_{HO\ (S)}$	0.25 0	— —	$t_{CYC(M)}$ ns
12	Rise Time (20% V_{DD} to 70% V_{DD}, C_L = 200 pF) SPI Outputs (SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and $\overline{\text{SS}}$)	t_{RM} t_{RS}	— —	100 2.0	ns µs
13	Fall Time (20% V_{DD} to 20% V_{DD}, C_L = 200 pF) SPI Outputs (SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and $\overline{\text{SS}}$)	t_{FM} t_{FS}	— —	100 2.0	ns µs

*Signal production depends on software.

**Assumes 200 pF load on all SPI pins.

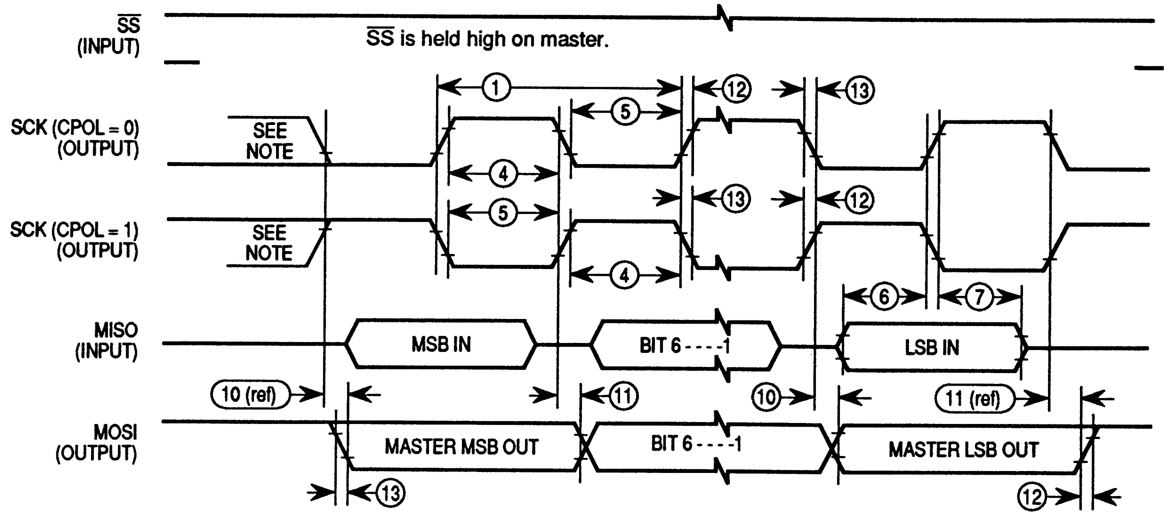
Table 13-7. Serial Peripheral Interface Timing (3.3 Vdc)

V_{DD} = 3.3 Vdc ± 10%; V_{SS} = 0 Vdc, $T_A = T_L$ to T_H (Refer Figures 13-11 and 13-12.)

Num	Characteristic	Symbol	Min	Max	Unit
	Operating Frequency Master Slave	 $f_{OP(M)}$ $f_{OP(S)}$	 dc dc	 0.5 1.0	 f_{OP} MHz
1	Cycle Time Master Slave	 $t_{CYC(M)}$ $t_{CYC(S)}$	 2.0 1.0	 — —	 t_{CYC} µs
2	Enable Lead Time Master Slave	 $t_{LEAD\ (M)}$ $t_{LEAD\ (S)}$	 * 500	 — —	 ns ns
3	Enable Lag Time Master Slave	 $t_{LAG\ (M)}$ $t_{LAG\ (S)}$	 * 1.5	 — —	 ns µs
4	Clock (SCK) High Time Master Slave	 $t_{W(SCKH)M}$ $t_{W(SCKH)S}$	 720 400	 — —	 µs ns
5	Clock (SCK) Low Time Master Slave	 $t_{W(SCKL)M}$ $t_{W(SCKL)S}$	 720 400	 — —	 µs ns
6	Data Setup Time (Inputs) Master Slave	 $t_{SU(M)}$ $t_{SU(S)}$	 200 200	 — —	 ns ns
7	Data Hold Time (Inputs) Master Slave	 $t_{H(M)}$ $t_{H(S)}$	 200 200	 — —	 ns ns
8	Slave Access Time (Time to Data Active from High-Impedance State)	t_A	0	250	ns
9	Slave Disable Time (Hold Time to High-Impedance State)	t_{DIS}	—	500	ns
10	Data Valid Master (Before Capture Edge) Slave (After Enable Edge)**	 $t_{V(M)}$ $t_{V(S)}$	 0.25 —	 — 500	 $t_{CYC(M)}$ ns
11	Data Hold Time (Outputs) Master (After Capture Edge) Slave (After Enable Edge)	 $t_{HO\ (M)}$ $t_{HO\ (S)}$	 0.25 0	 — —	 $t_{CYC(M)}$ ns
12	Rise Time (20% V_{DD} to 70% V_{DD}, C_L = 200 pF) SPI Outputs (SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and $\overline{\text{SS}}$)	 t_{RM} t_{RS}	 — —	 200 2.0	 ns µs
13	Fall Time (20% V_{DD} to 20% V_{DD}, C_L = 200 pF) SPI Outputs (SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and $\overline{\text{SS}}$)	 t_{FM} t_{FS}	 — —	 200 2.0	 ns µs

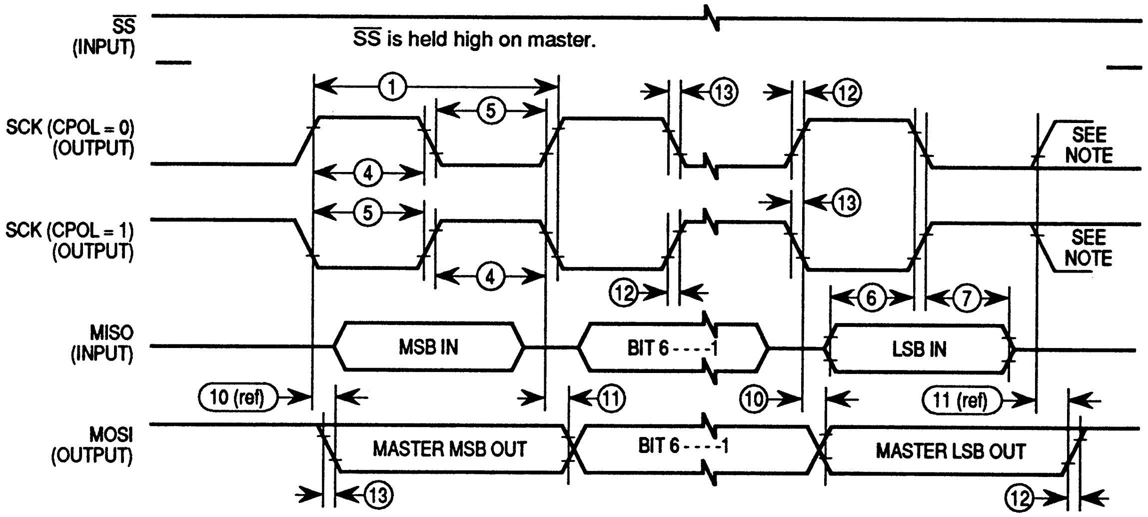
*Signal production depends on software.

**Assumes 200 pF load on all SPI pins.



NOTE: This first clock edge is generated internally but is not seen at the SCK pin.

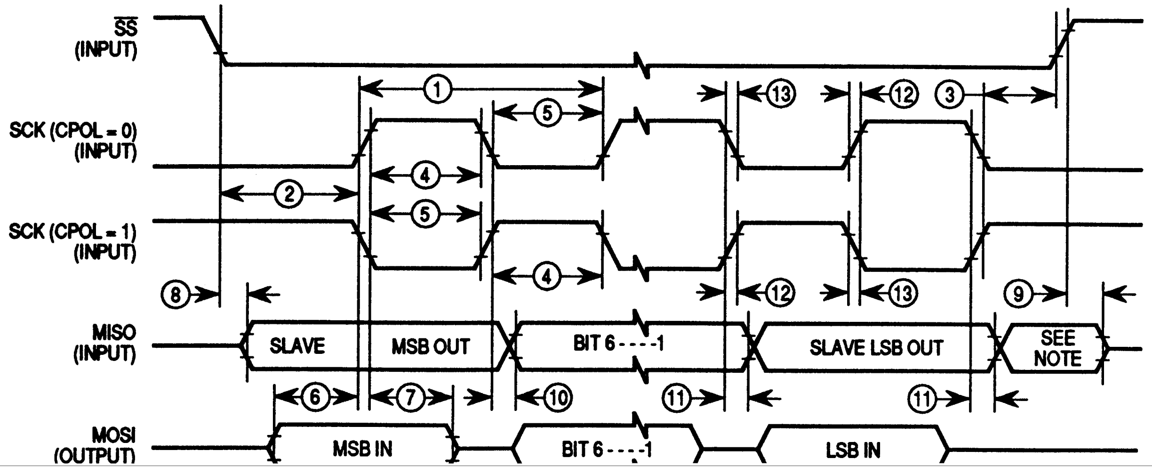
a) SPI Master Timing (CPHA = 0)



NOTE: This last clock edge is generated internally but is not seen at the SCK pin.

b) SPI Master Timing (CPHA = 1)

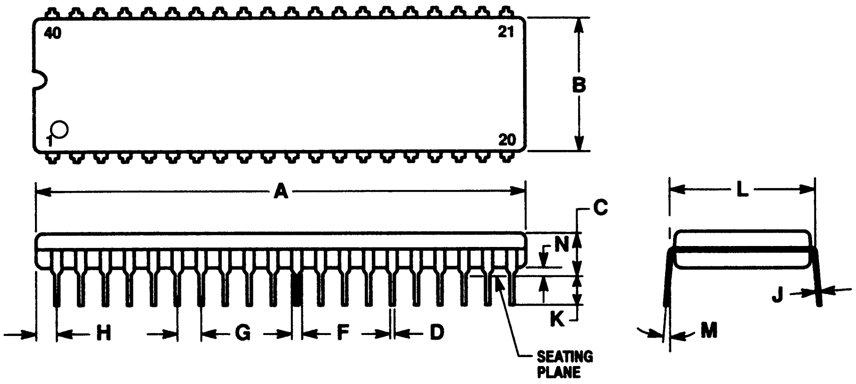
Figure 13-9. SPI Master Timing Diagram


SS (INPUT)
SCK (CPOL = 0) (INPUT)
SCK (CPOL = 1) (INPUT)
MISO (INPUT)
MOSI (OUTPUT)
①
②
③
④
⑤
⑥
⑦
⑧
⑨
⑩
⑪
⑫
⑬
SLAVE
MSB OUT
BIT 6 - - - - 1
SLAVE LSB OUT
SEE NOTE
MSB IN
BIT 6 - - - - 1
LSB IN

SECTION 14
MECHANICAL SPECIFICATIONS

This section describes the dimensions of the plastic dual-in-line package (PDIP), plastic-leaded chip carrier (PLCC) package, quad flat pack (QFP) package, and plastic shrink DIP (SDIP) package.

14.1 Plastic Dual-in-Line Package (PDIP)

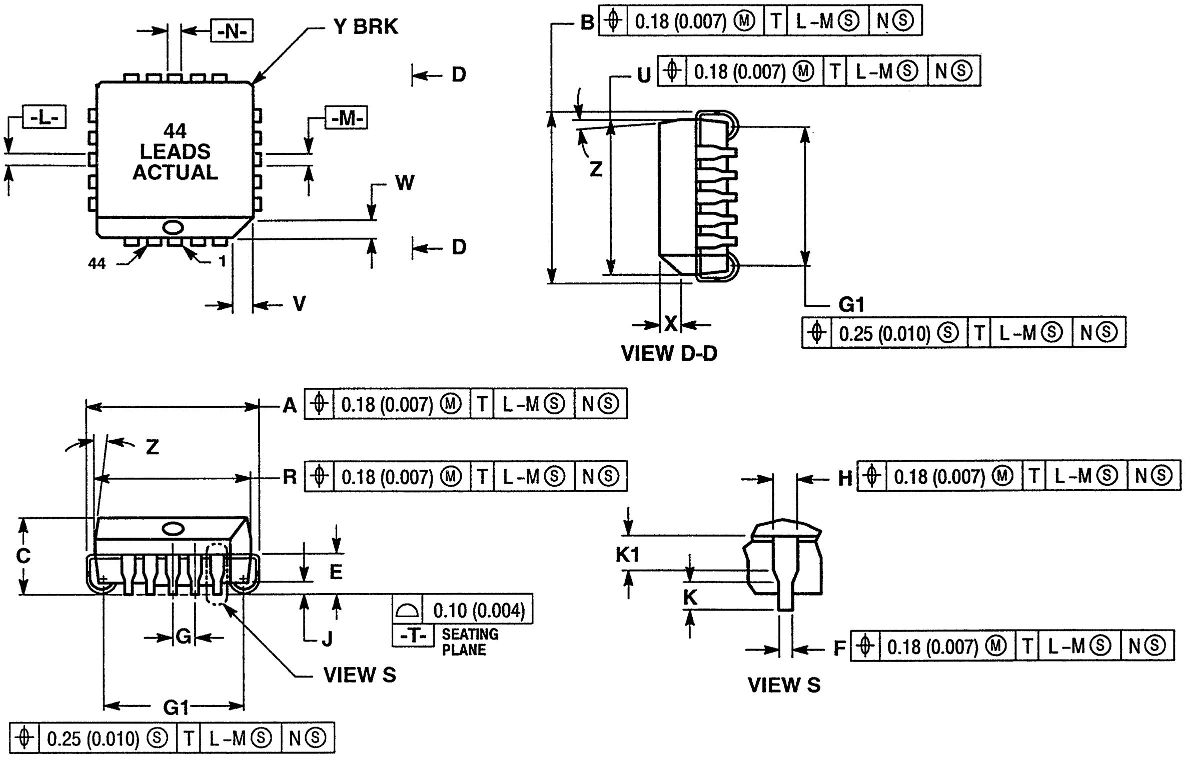


NOTES:
1. POSITIONAL TOLERANCE OF LEADS (D), SHALL BE WITHIN 0.25 mm (0.010) AT MAXIMUM MATERIAL CONDITION, IN RELATION TO SEATING PLANE AND EACH OTHER.
2. DIMENSION L TO CENTER OF LEADS WHEN FORMED PARALLEL.
3. DIMENSION B DOES NOT INCLUDE MOLD FLASH.

	MILLIMETERS		INCHES	
DIM	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

Figure 14-1. MC68HC05C9P (Case # 711-03)

14.2 Plastic-Leaded Chip Carrier (PLCC)



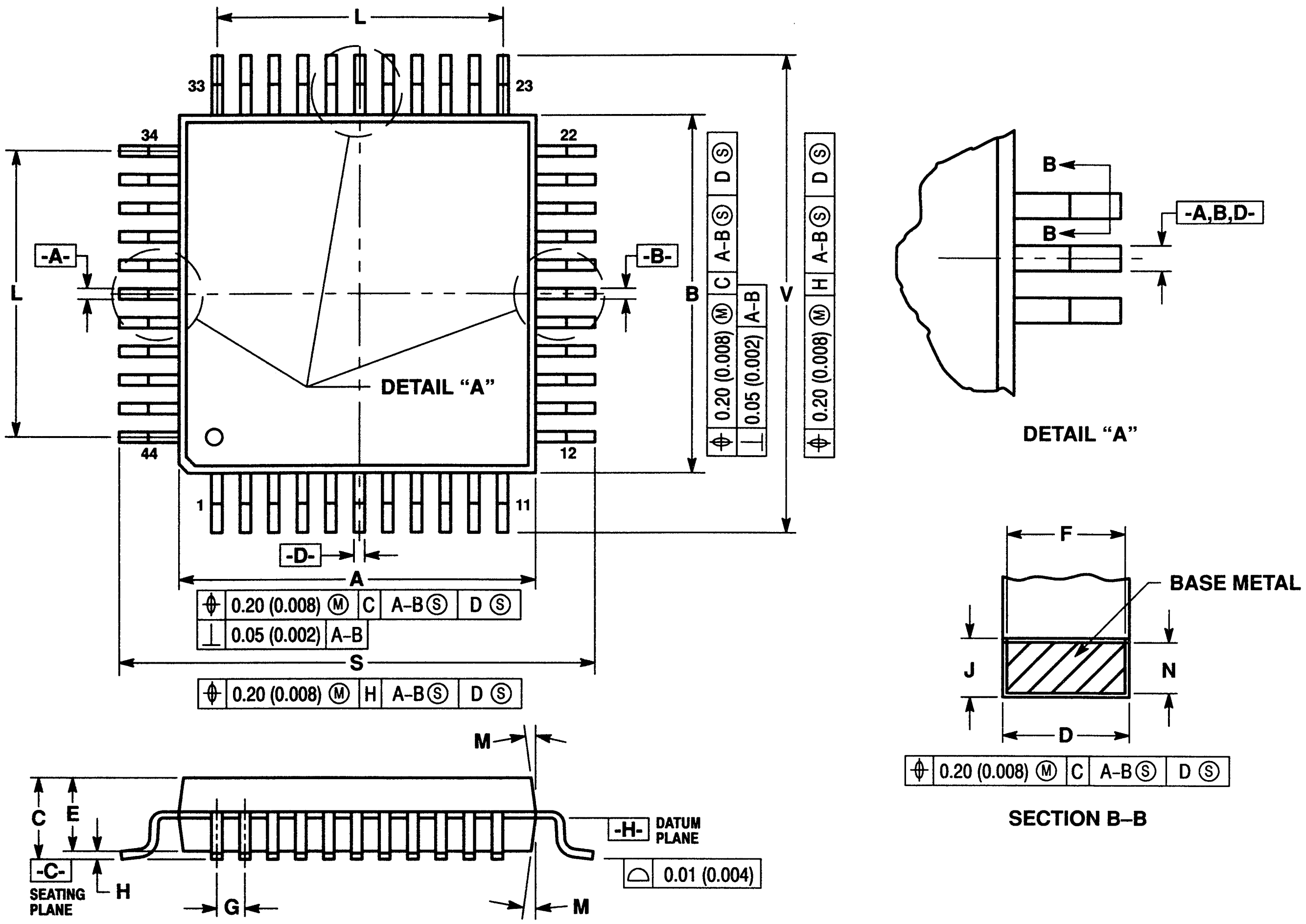
NOTES:

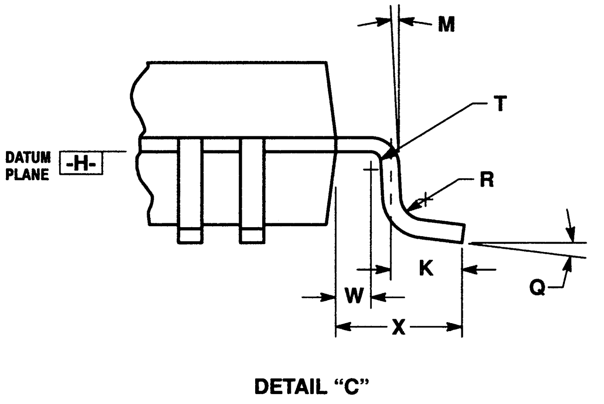
1. **DUE TO SPACE LIMITATION, CASE 777-02 SHALL BE REPRESENTED BY A GENERAL (SMALLER) CASE OUTLINE DRAWING RATHER THAN SHOWING ALL 44 LEADS.**
2. DATUMS -L-, -M-, AND -N- DETERMINED WHERE TOP OF LEAD SHOULDER EXITS PLASTIC BODY AT MOLD PARTING LINE.
3. DIM G1, TRUE POSITION TO BE MEASURED AT DATUM -T-, SEATING PLANE.
4. DIM R AND U DO NOT INCLUDE MOLD FLASH. ALLOWABLE MOLD FLASH IS 0.25 (0.010) PER SIDE.
5. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
6. CONTROLLING DIMENSION: INCH.
7. THE PACKAGE TOP MAY BE SMALLER THAN THE PACKAGE BOTTOM BY UP TO .012 (.300). DIMENSIONS R AND U ARE DETERMINED AT THE OUTERMOST EXTREMES OF THE PLASTIC BODY EXCLUSIVE OF MOLD FLASH, TIE BAR BURRS, GATE BURRS AND INTERLEAD FLASH, BUT INCLUDING ANY MISMATCH BETWEEN THE TOP AND BOTTOM OF THE PLASTIC BODY.
8. DIMENSION H DOES NOT INCLUDE DAMBAR PROTRUSION OR INTRUSION. THE DAMBAR PROTRUSION(S) SHALL NOT CAUSE THE H DIMENSION TO BE GREATER THAN .037 (.940). THE DAMBAR INTRUSION(S) SHALL NOT CAUSE THE H DIMENSION TO BE SMALLER THAN .025 (.635).
9. 777-01 IS OBSOLETE, NEW STANDARD 777-02.

	MILLIMETERS		INCHES	
DIM	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	—

Figure 14-2. MC68HC05C9FN (Case # 777-02)

14.3 Quad Flat Pack (QFP)



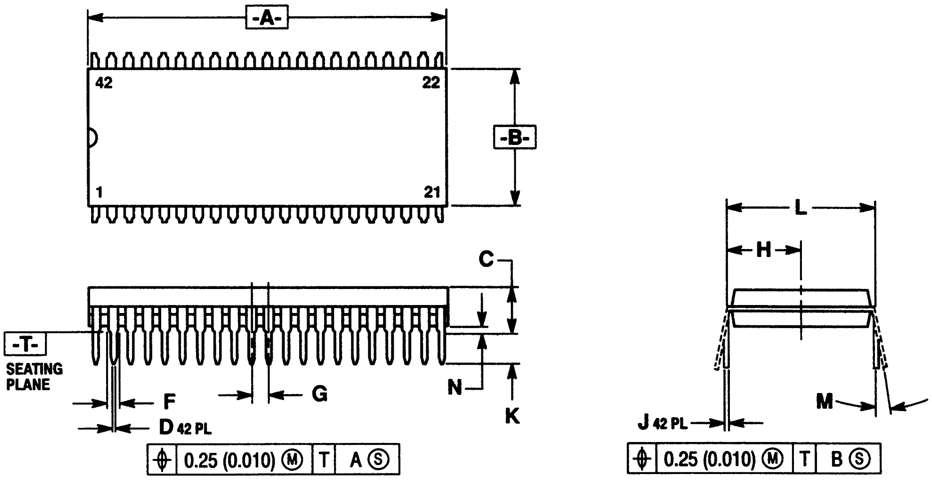


NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: MILLIMETER.
3. DATUM PLANE -H- IS LOCATED AT BOTTOM OF LEAD AND IS COINCIDENT WITH THE LEAD WHERE THE LEAD EXITS THE PLASTIC BODY AT THE BOTTOM OF THE PARTING LINE.
4. DATUMS A–B AND -D- TO BE DETERMINED AT DATUM PLANE -H-.
5. DIMENSIONS S AND V TO BE DETERMINED AT SEATING PLANE -C-.
6. DIMENSIONS A AND B DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.25 (0.010) PER SIDE. DIMENSIONS A AND B DO INCLUDE MOLD MISMATCH AND ARE DETERMINED AT DATUM PLANE -H-.
7. DIMENSION D DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL BE 0.08 (0.003) TOTAL IN EXCESS OF THE D DIMENSION AT MAXIMUM MATERIAL CONDITION. DAMBAR CANNOT BE LOCATED ON THE LOWER RADIUS OR THE FOOT.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	9.90	10.10	0.390	0.398
B	9.90	10.10	0.390	0.398
C	2.10	2.45	0.083	0.096
D	0.30	0.45	0.012	0.018
E	2.00	2.10	0.079	0.083
F	0.30	0.40	0.012	0.016
G	0.80 BSC		0.031 BSC	
H	—	0.25	—	0.010
J	0.13	0.23	0.005	0.009
K	0.65	0.95	0.026	0.037
L	8.00 REF		0.315 REF	
M	5°	10°	5°	10°
N	0.13	0.17	0.005	0.007
Q	0°	7°	0°	7°
R	0.13	0.30	0.005	0.012
S	12.95	13.45	0.510	0.530
T	0.13	—	0.005	—
U	0°	—	0°	—
V	12.95	13.45	0.510	0.530
W	0.40	—	0.016	—
X	1.6 REF		0.063 REF	

Figure 14-3. MC68HC05C9FB (Case # 824A-01)

14.4 Plastic Shrink DIP (SDIP)



NOTES:

1. DIMENSIONS AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: INCH.
3. DIMENSION L TO CENTER OF LEAD WHEN FORMED PARALLEL.
4. DIMENSIONS A AND B DO NOT INCLUDE MOLD FLASH. MAXIMUM MOLD FLASH 0.25 (0.010).

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	36.45	37.21	1.435	1.465
B	13.72	14.22	0.540	0.560
C	3.94	5.08	0.155	0.200
D	0.36	0.56	0.014	0.022
F	0.81	1.17	0.032	0.046
G	1.778 BSC		0.070 BSC	
H	7.62 BSC		0.300 BSC	
J	0.20	0.38	0.008	0.015
K	2.92	3.43	0.115	0.135
L	15.24 BSC		0.600 BSC	
M	0°	15°	0°	15°
N	0.51	1.02	0.020	0.040

Figure 14-4. MC68HC05C9B (Case # 858-01)

SECTION 15
ORDERING INFORMATION

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

15.1 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 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 **15.2 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.

15.2 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.

15.2.1 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**. Refer to 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.

15.2.2 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**. Refer to 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.

15.3 ROM Program Verification

The primary use for the on-chip ROM is to contain the customer's application program. The customer develops and debugs the application program and then submits the 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.

15.4 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.

Home Page:
www.freescale.com
email:
support@freescale.com
USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
(800) 521-6274
480-768-2130
support@freescale.com
Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com
Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku
Tokyo 153-0064, Japan
0120 191014
+81 2666 8080
support.japan@freescale.com
Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate,
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com
For Literature Requests Only:
Freescale Semiconductor
Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
(800) 441-2447
303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor
@hibbertgroup.com

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

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

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

Freescale Semiconductor, Inc.