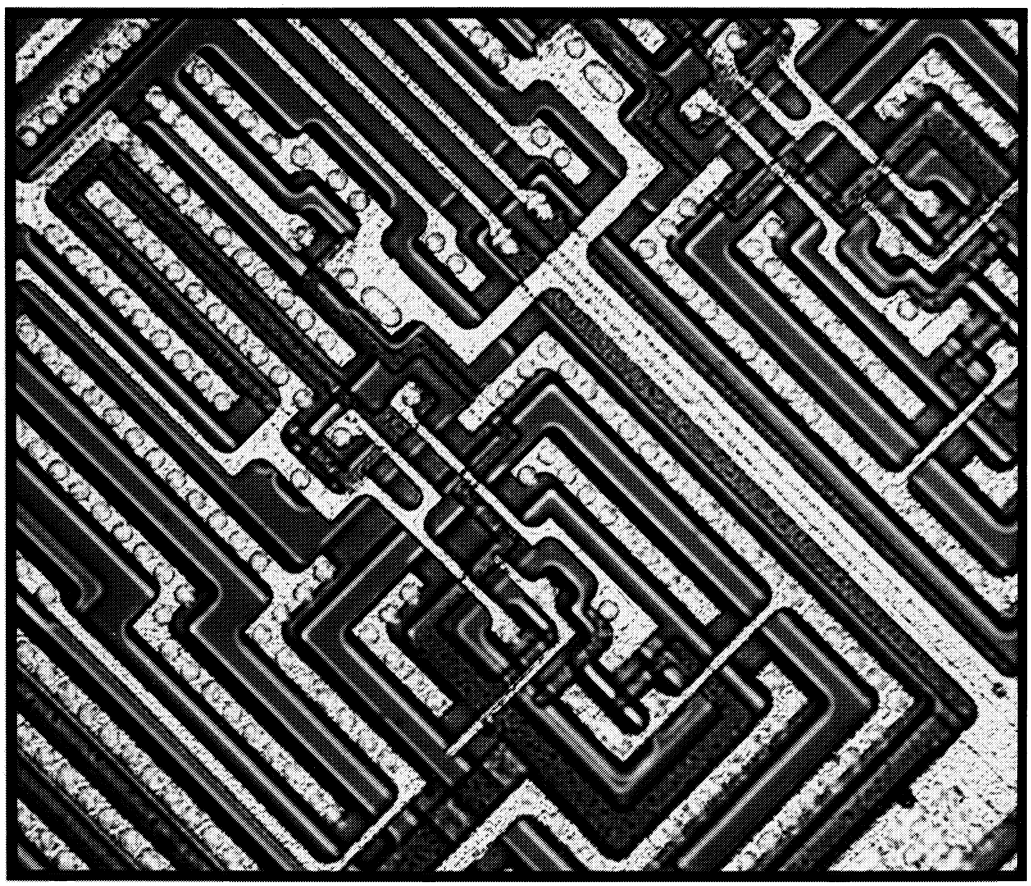
MC68HC05L6

Advance Information

8-Bit Microcomputer with Liquid Crystal Driver Circuitry



This document contains information on a new product. Specifications and information herein are subject to change without notice.

freescale™ semiconductor

ADI1254

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Concluded)

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS
(Continued)

LIST OF TABLES

SECTION 1
INTRODUCTION

1.1 GENERAL

The MC68HC05L6 HCMOS microcomputer is a member of the M68HC05 Family of low-cost single-chip microcomputers. This 8-bit microcomputer contains an on-chip oscillator, CPU, RAM, ROM, I/O, a serial peripheral interface, timer, liquid crystal display driver circuitry and a tone generator.

1.2 FEATURES

The following are some of the hardware and software highlights of the MC68HC05L6 microcomputer.

HARDWARE FEATURES

- HCMOS Technology
- 8-Bit Architecture
- Power Saving Stop and Wait Modes
- RAM and CPU Register Contents Valid at V_{CC} = 2.0 Volts (CPU Halted)
- Independent Power Supplies (CPU - LCD) 3 to 6 Volt Operation
- 176 Bytes of On-Chip RAM
- 6208 Bytes of On-Chip ROM
- 24 Bidirectional I/O Lines
- 4.0 MHz Internal Operating Frequency at 5 Volts
- Internal 16-Bit Timer Similar to MC6801 Timer
- Serial Peripheral Interface System
- Self-Check Mode
- External, Timer, and Serial Peripheral Interface Interrupts
- Master Reset and Power-On Reset
- On-Chip Oscillator with RC or Crystal/Ceramic Resonator Mask Options
- LCD 2/3, 1/3 V_{LL} Divider Circuitry Included
- Tone Generator
- 68-Pin Quad Package

SOFTWARE FEATURES

- Similar to MC6800
- 8×8 Unsigned Multiply Instruction
- Efficient Use of Program Space
- Versatile Interrupt Handling
- True Bit Manipulation

SOFTWARE FEATURES (Continued)

- Indexed Addressing for Tables
- Efficient Instruction Set
- Memory Mapped I/O
- Programmable LCD Driver Circuitry
  - 96 Pixels in ×4 Mode
  - 72 Pixels in ×3 Mode
- Low Cost Development Support with Evaluation Module

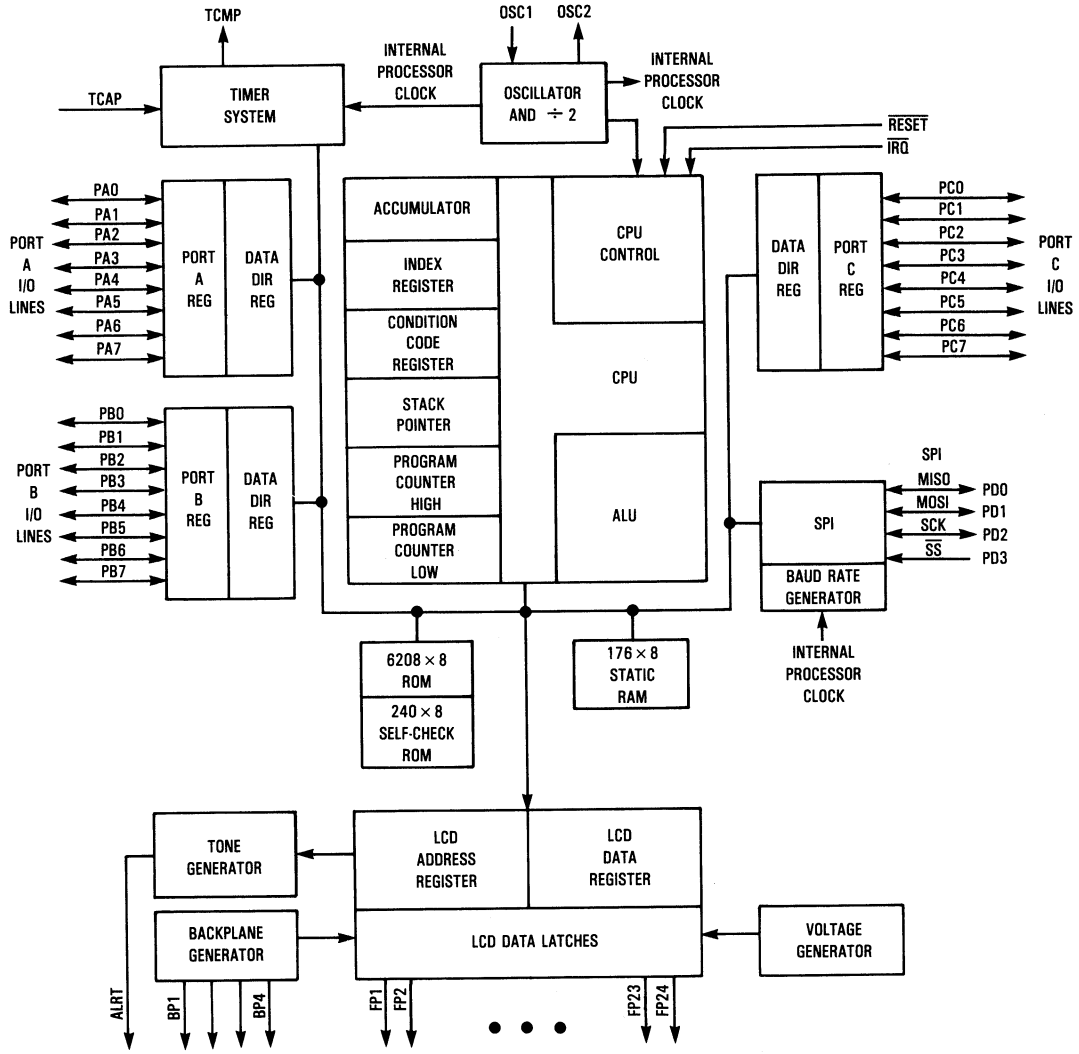


Figure 1-1. MC68HC05L6 Microcomputer Block Diagram

SECTION 2
SIGNAL DESCRIPTION, INPUT/OUTPUT PROGRAMMING, MEMORY, CPU REGISTERS, AND SELF-CHECK

2.1 INTRODUCTION

This section provides a description of the input/output signals, input/output programming, memory, CPU registers, and self-check.

2.2 SIGNAL DESCRIPTION

The following paragraphs provide a description of the input/output signals.

2.2.1 V_{DD}, V_{LL}, and V_{SS}

Power is supplied to the microcomputer using these pins. V_{DD} is power for the processor, RAM, ROM, etc., V_{LL} is power for the LCD driver circuits, and V_{SS} is ground.

2.2.2 $\overline{IRQ}$ (External Interrupt Request)

IRQ is a mask-programmable option which provides two different choices of interrupt triggering sensitivity: 1) negative edge-sensitive triggering only, or 2) both negative edge-sensitive and level-sensitive triggering.

The microcomputer completes the current instruction before it responds to the interrupt request. When the $\overline{IRQ}$ pin goes low for at least one t_{ILIH}, a logic one is latched internally to signify the interrupt request. When the microcomputer completes its current instruction, the interrupt latch is tested. If an interrupt is pending and the interrupt mask bit (I bit) in the condition code register is clear, the interrupt sequence begins.

If level-sensitive triggering has been implemented, the $\overline{IRQ}$ input requires an external resistor to V_{DD} for "wire-OR" operation. See **INTERRUPTS** in Section 3 for more detail concerning interrupts.

2.2.3 $\overline{RESET}$

The $\overline{RESET}$ input is not required for startup but can be used to reset the microcomputer internal state and provide an orderly software startup procedure. Refer to **RESETS** in Section 3 for a detailed description.

2.2.4 TCAP

The TCAP input controls the input capture feature for the on-chip programmable timer system. Refer to **INPUT CAPTURE REGISTER** in Section 4 for additional information.

2.2.5 TCMP

The TCMP pin provides an output for the output compare feature of the on-chip timer system. Refer to **OUTPUT COMPARE REGISTER** in Section 4 for additional information.

2.2.6 OSC1, OSC2

The MC68HC05L6 can be configured by mask option to accept either a crystal/ceramic resonator input or an RC network to control the internal oscillator. The internal clocks are derived by a divide-by-two of the internal oscillator frequency (f_{osc}).

2.2.6.1 CRYSTAL. The circuit shown in in Figure 2-1(b) is recommended when using a crystal. The internal oscillator is designed to interface with an AT-cut parallel resonant quartz crystal resonator in the frequency range specified for f_{osc} (refer to paragraph 8.7 or 8.8 Control Timing). Use of an external CMOS oscillator is recommended when crystals outside the specified ranges are to be used. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time. Refer to paragraphs 8.5 or 8.6 for V_{DD} specifications.

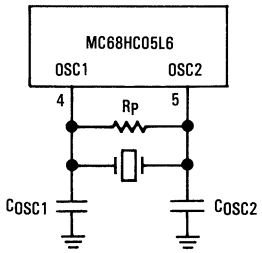
CRYSTAL

	2 MHz	4 MHz	UNITS
R_{SMAX}	400	75	Ω
C_0	5	7	pF
C_1	0.008	0.012	μF
C_{OSC1}	15–40	15–30	pF
C_{OSC2}	15–30	15–25	pF
R_P	10	10	MΩ
Q	30	40	K

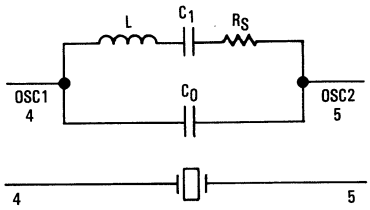
CERAMIC RESONATOR

	2–4 MHz	UNITS
R_S (TYPICAL)	10	Ω
C_0	40	pF
C_1	4.3	pF
C_{OSC1}	30	pF
C_{OSC2}	30	pF
R_P	1–10	MΩ
Q	1250	–

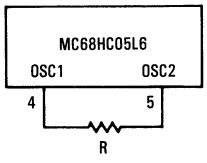
(a) CRYSTAL/CERAMIC RESONATOR PARAMETERS



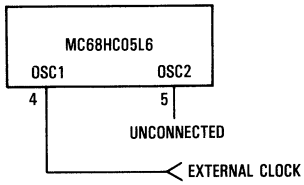
(b) CRYSTAL/CERAMIC RESONATOR OSCILLATOR CONNECTIONS



(c) EQUIVALENT CRYSTAL CIRCUIT



(d) RC OSCILLATOR CONNECTIONS



(e) EXTERNAL CLOCK SOURCE CONNECTIONS (EITHER CRYSTAL OR RC MASK OPTIONS)

Figure 2-1. Oscillator Connections

2.2.6.2 CERAMIC RESONATOR. A ceramic resonator may be used in place of the crystal in cost-sensitive applications. The circuit in Figure 2-1(b) is recommended when using a ceramic resonator. Figure 2-1(a) lists the recommended capacitance and feedback resistance values. The manufacturer of the particular ceramic resonator being considered should be consulted for specific information.

2.2.6.3 RC. If the RC oscillator is selected, then a resistor is connected to the oscillator pins as shown in Figure 2-1(d). The relationship between R and f_{osc} is shown in Figure 2-2.

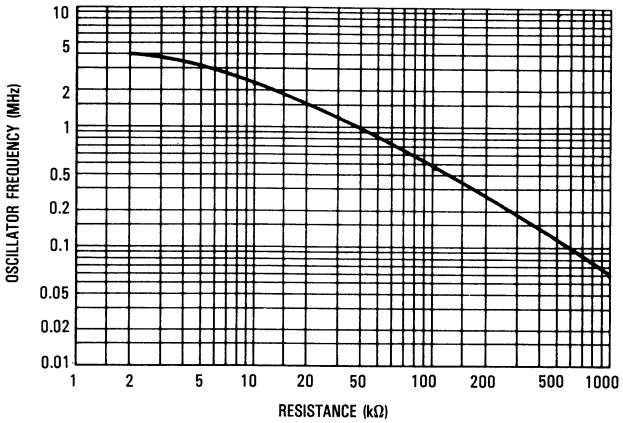


Figure 2-2. Typical Frequency vs Resistance for RC Oscillator Option Only (V_{DD} = 5.0 V)

2.2.6.4 EXTERNAL CLOCK. An external clock should be applied to the OSC1 input with the OSC2 input not connected, as shown in Figure 2-1(e). An external clock may be used with either the RC or crystal oscillator option.

2.2.7 PA0–PA7

These eight I/O lines comprise port A. The state of any pin is software programmable and all port A lines are configured as input during power-on or reset. Refer to the **INPUT/OUTPUT PROGRAMMING** paragraph for a detailed description of I/O programming.

2.2.8 PB0–PB7

These eight lines comprise port B. The state of any pin is software programmable and all port B lines are configured as input during power-on or reset. Refer to the **INPUT/OUTPUT PROGRAMMING** paragraph for a detailed description of I/O programming.

2.2.9 PC0–PC7

These eight lines comprise port C. The state of any pin is software programmable and all port C lines are configured as input during power-on or reset. Refer to the **INPUT/OUTPUT PROGRAMMING** paragraph for a detailed description of I/O programming.

2.2.10 PD0–PD3

These four lines comprise port D. This port performs a dual function for the MC68HC05L6: a 4-bit input-only port or the SPI input/output port.

When the serial peripheral interface is disabled, port D can be used to input four bits of data (least significant nibble) to the port D data register at location $0003. The most significant nibble will always read as zeros.

When the serial peripheral interface is enabled, these four lines function as the interface lines between the SPI-type microprocessors and/or peripherals. Refer to **SECTION 5 SERIAL PERIPHERAL INTERFACE** for additional information

2.2.11 BP1–BP4

These four output lines provide the backplane drive signals to the liquid crystal display unit.

2.2.12 FP1–FP24

These 24 output lines provide the frontplane drive signals to the liquid crystal display unit.

2.2.13 ALRT

This pin provides the tone generator output signal.

2.3 INPUT/OUTPUT PROGRAMMING

Ports A, B, and C may be programmed as an input or an output under software control. The direction of the pins is determined by the state of the corresponding bit in the port data direction register (DDR). Each 8-bit port has an associated 8-bit data register. Any port A, port B, or port C pin is configured as an output if its corresponding DDR bit is set to a logic one and as an input if the DDR bit is set to a logic zero.

At power-on reset, all DDRs are cleared, which configures all port A, B, and C pins as inputs. The data direction registers are capable of being written to or read by the processor. Refer to Figure 2-3 and Table 2-1. When a port is programmed as an output, a read of the data register actually reads the value of the output data latch and not the I/O pin.

Table 2-1. I/O Pin Functions

R/$\overline{W}$*	DDR	I/O Pin Function
0	0	The I/O pin is in input mode. Data is written into the output data latch.
0	1	Data is written into the output data latch and output to the I/O pin.
1	0	The state of the I/O pin is read.
1	1	The I/O pin is in an output mode. The output data latch is read.

*R/$\overline{W}$ is an internal signal.

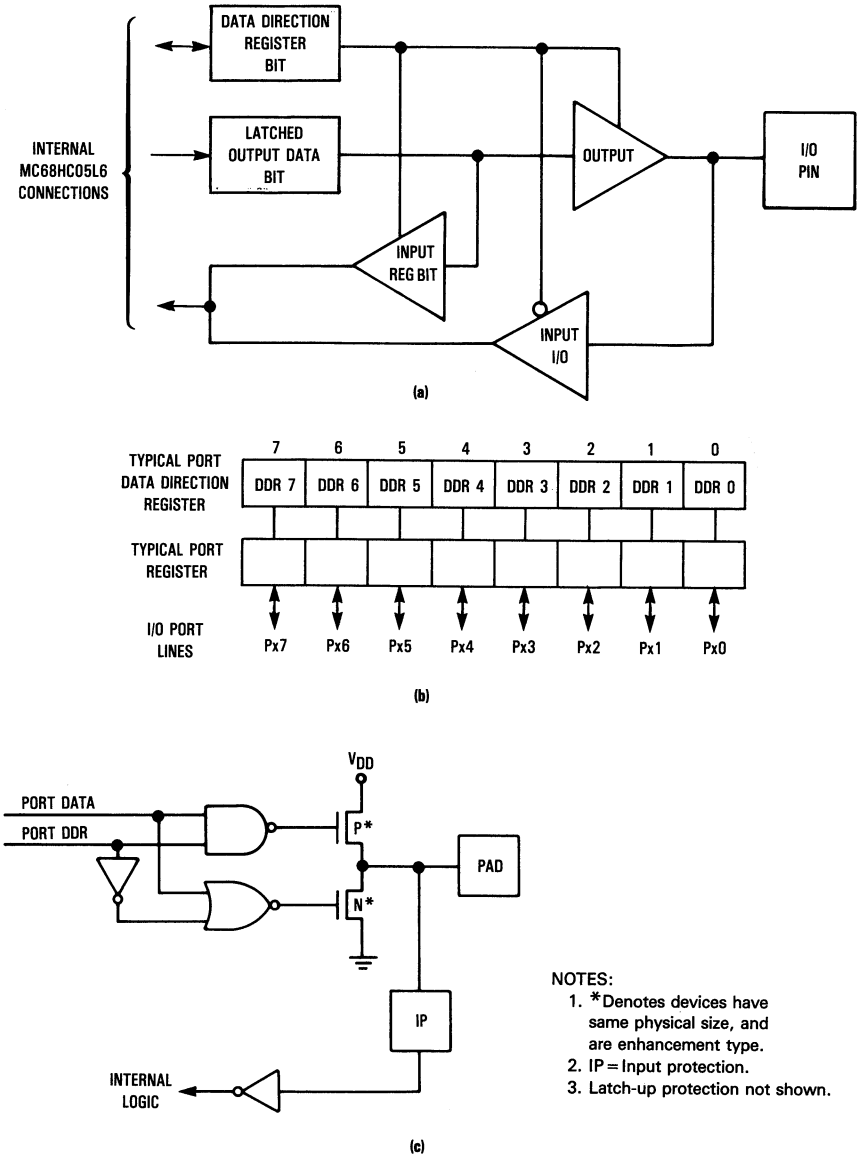


Figure 2-3. Typical Parallel Port I/O Circuitry

2.4 MEMORY

A memory map configuration for the MC68HC05L6 is shown in Figure 2-4. The first 256 bytes of memory (page zero) include 22 bytes used as registers for input/output features such as the data ports, data direction, timer, serial peripheral interface, and the liquid crystal display drivers along with 40 bytes of user ROM and 176 bytes of user RAM. The 176 bytes of user RAM includes up to 64 bytes for the stack.

User ROM consists of a 48 byte segment within page zero and a 6144 byte segment starting at location $0100 along with a 16 byte segment located at the top of available memory to be used for vectors to interrupt or special subroutines. Self check (244 bytes) and self-check vector (16 bytes) memory locations are located just below the user vectors. The segment of ROM starting at $1900 through $1EEB is not implemented and not available to the user at this time.

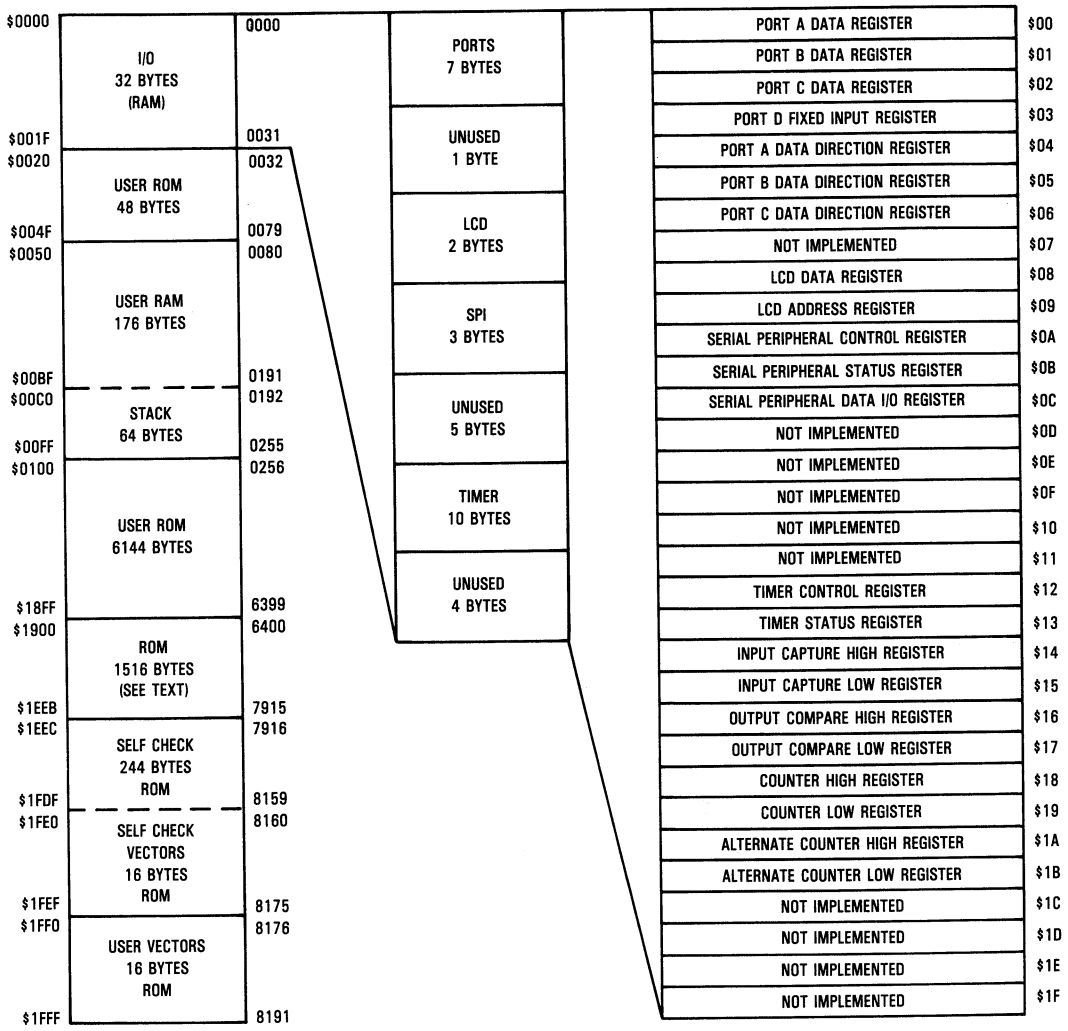


Figure 2-4. Memory Map

2.5 CPU REGISTERS

The MC68HC05L6 CPU contains five registers, as shown in the programming model of Figure 2-5. The interrupt stacking order is shown in Figure 2-6.

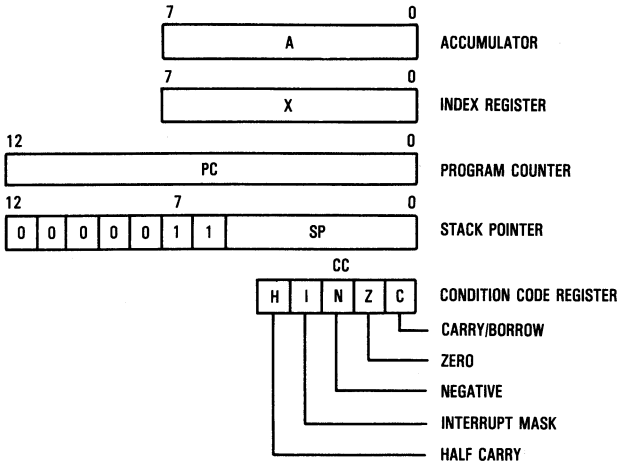
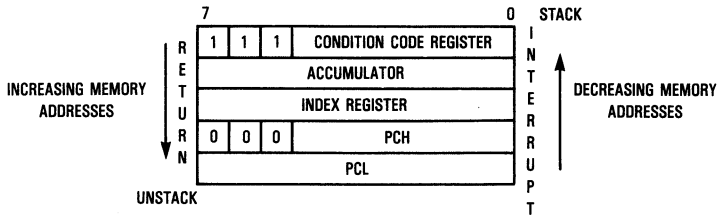


Figure 2-5. Programming Model



NOTE: Since the Stack Pointer decrements during pushes, the PCL is stacked first, followed by PCH, etc. Pulling from the stack is in the reverse order.

Figure 2-6. Stacking Order

2.5.1 Accumulator (A)

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

2.5.2 Index Register (X)

The X register is an 8-bit register which is used during the indexed modes of addressing. It provides an 8-bit value which is used to create an effective address. The index register is also used for data manipulations with the read-modify-write type of instructions and as a temporary storage register when not performing addressing operations.

2.5.3 Program Counter (PC)

The program counter is a 13-bit register that contains the address of the next instruction to be executed by the processor.

2.5.4 Stack Pointer (SP)

The stack pointer is a 13-bit register containing the address of the next free locations on the push-down/pop-up stack. When accessing memory, the seven most significant bits are permanently configured to 0000011. These seven bits are appended to the six least significant register bits to produce an address within the range of $00FF to $00C0. The stack area of RAM is used to store the return address on subroutine calls and the machine state during interrupts. During external or power-on reset, and during a reset stack pointer (RSP) instruction, the stack pointer is set to its upper limit ($00FF). Nested interrupt and/or subroutines may use up to 64 (decimal) locations. When the 64 locations are exceeded, the stack pointer wraps around and points to its upper limit ($00FF), thus, losing the previously stored information. A subroutine call occupies two RAM bytes on the stack, while an interrupt uses five RAM bytes.

2.5.5 Condition Code Register (CC)

The condition code register is a 5-bit register which indicates the results of the instruction just executed as well as the state of the processor. These bits can be individually tested by a program and specified action taken as a result of their state. Each bit is explained in the following paragraphs.

2.5.5.1 HALF CARRY BIT (H). The H bit is set to a one when a carry occurs between bits three and four of the ALU during an ADD or ADC instruction. The H bit is useful in binary coded decimal subroutines.

2.5.5.2 INTERRUPT MASK BIT (I). When the I bit is set, all interrupts are disabled. Clearing this bit enables the interrupts. If an external interrupt occurs while the I bit is set, the interrupt is latched and is processed after the I bit is next cleared; therefore, no interrupts are lost because of the I bit being set. An internal interrupt can be lost if it is cleared while the I bit is set (refer to **SECTION 4 PROGRAMMABLE TIMER**, and **SECTION 5 SERIAL PERIPHERAL INTERFACE** for more information).

2.5.5.3 NEGATIVE (N). When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation is negative (bit 7 in the result is a logic one).

2.5.5.4 ZERO (Z). When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation is zero.

2.5.5.5 CARRY/BORROW (C). Indicates that a carry or borrow out of the arithmetic logic unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions, shifts, and rotates.

2.6 SELF-CHECK

The self-check capability of the MC68HC05L6 MCU provides an internal check to determine if the device is functional. Self-check is performed using the circuit shown in the schematic diagram of Figure 2-7. As shown in the diagram, port C pins PC0-PC3 are monitored (light emitting diodes are shown but other devices could be used) for the self-check results. The self-check mode is entered by applying a 9 Vdc input (through a 4.7 kilohm resistor) to the $\overline{\text{IRQ}}$ pin and 5 Vdc input (through a 4.7 kilohm resistor) to the TCAP pin and then depressing the reset switch to execute a reset. After reset, the following seven tests are performed automatically.

I/O—Functionally exercises ports A, B, and C
RAM—Counter test for each RAM byte
Timer—Tracks counter register and checks OCF flag
LCD—Displays the SC number
ROM—Exclusive OR with odd ones parity result
SPI—Transmission test with check for SPIF, WCOL, and MODF flags
INTERRUPTS—Tests external timer and SPI interrupts

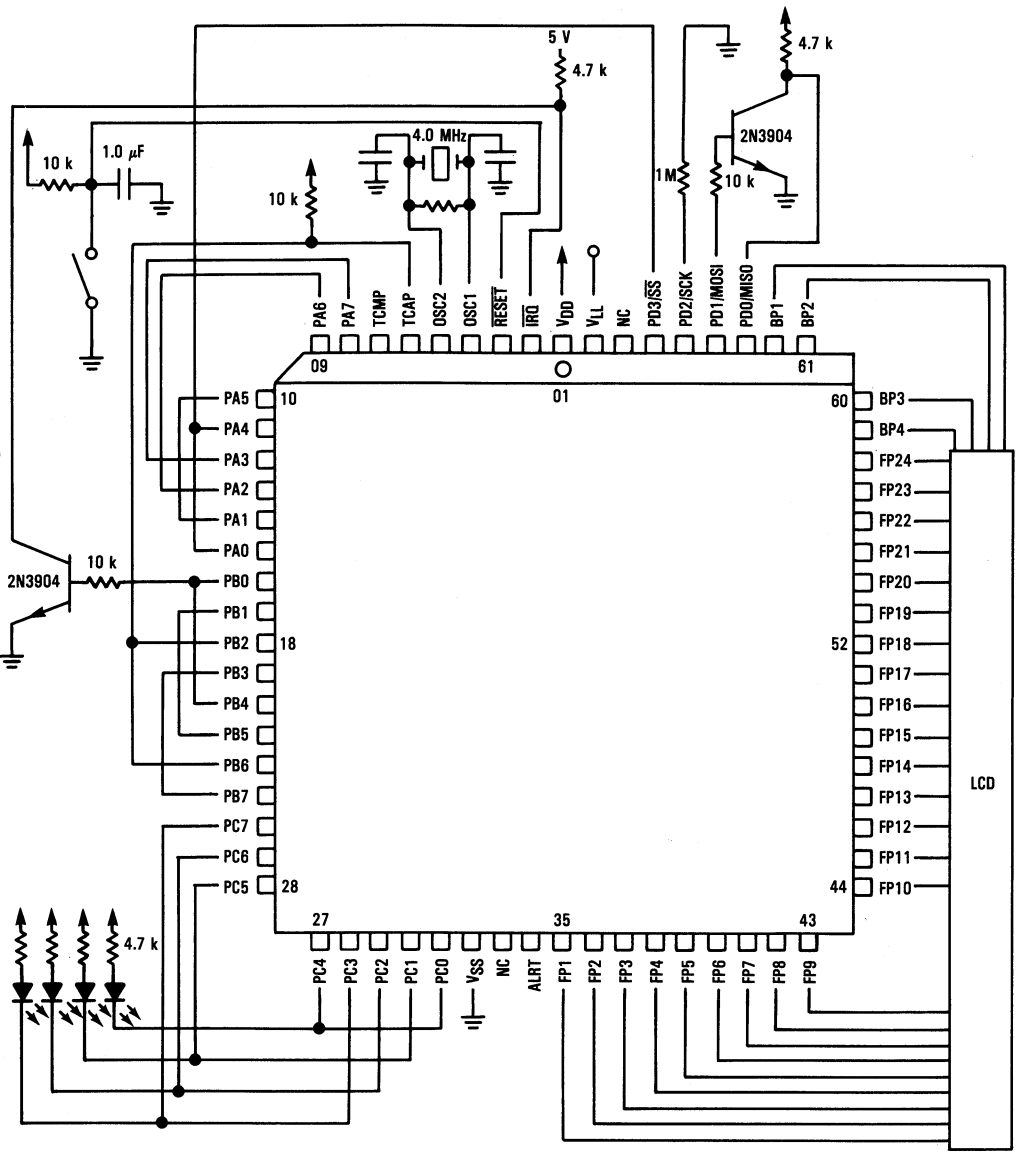


Figure 2-7. Self-Check Circuit Schematic Diagram

Self-check results (using the LEDs as monitors) are shown in Table 2-2. The following subroutines are available to user programs and do not require any external hardware.

Table 2-2. 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 ROM
1	1	0	1	Bad SPI
1	1	1	0	Bad Interrupts or $\overline{IRQ}$ Request
Flashing				Good Device
All Others				Bad Device, Bad Port C, etc.

0 Indicates LED On; 1 Indicates LED is OFF.

2.7 TIMER TEST SUBROUTINE

This subroutine returns with the Z bit cleared if any error is detected; otherwise, the Z bit is set.

This subroutine is called at location $1F0E. The output compare register is first set to the current timer state. Because the timer is free running and has only a divide-by-four prescaler, each timer count cannot be tested. The test reads the timer once every 10 counts (40 cycles) and checks for correct counting. The test tracks the counter until the timer wraps around, triggering the output compare flag in the timer status register. RAM locations $0050 and $0051 are overwritten. Upon return to the user's program, X = 40. If the test passed, A = 0.

2.8 ROM CHECKSUM ROUTINE

This subroutine returns with the Z bit cleared if any error is detected; otherwise, the Z bit is set.

This subroutine is called at locations $1F7D with RAM locations $0053 equal to $01 and A = 0. A short routine is set up and executed in RAM to compute a checksum of the entire ROM pattern. Upon return to the user's program, X = 0. If the test passed, A = 0, RAM locations $0050 through $0053 are overwritten.

SECTION 3
RESET MODES, INTERRUPTS, LOW POWER MODES

3.1 INTRODUCTION

This section contains a description of the reset modes and the different interrupt capabilities of the MC68HC05L6. The two low power consumption modes (WAIT and STOP) are also discussed.

3.2 RESET MODES

The MC68HC05L6 has two reset modes: an active low external reset pin ($\overline{\text{RESET}}$) and a power-on reset function; refer to Figure 3-1.

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

The $\overline{\text{RESET}}$ input pin is used to reset the MCU to provide an orderly software startup procedure. When using the external reset mode, the $\overline{\text{RESET}}$ pin must stay low for a minimum of one and one half cycle times. The $\overline{\text{RESET}}$ pin has Schmitt Trigger characteristics to improve noise immunity.

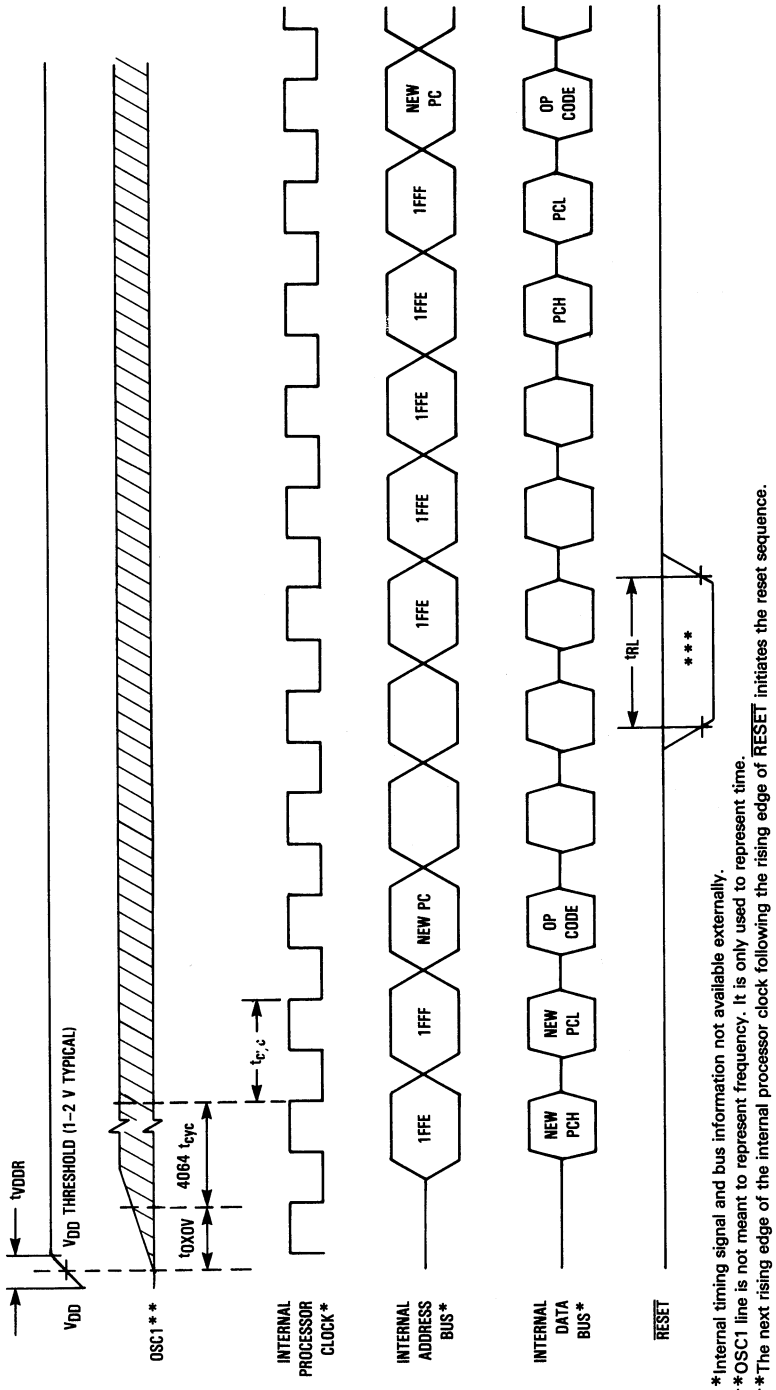
3.2.2 Power-On Reset

The power-on reset occurs when a positive transition is detected on power supply input pin. The power-on reset is used strictly for power turn-on conditions and should not be used to detect any drops in the power supply voltage. There is no provision for a power-down reset. The power-on circuitry provides a 4064 cycle time delay from the time that the oscillator becomes active. If the $\overline{\text{RESET}}$ pin is low at the end of the 4064 cycle time time out, the processor remains in the reset condition until the $\overline{\text{RESET}}$ input pin goes high. The user must ensure that input power has risen to a point where the MCU can operate properly prior to the time the 4064 cycle time delay has elapsed. If there is any doubt, the $\overline{\text{RESET}}$ pin should remain low until such time that input power has risen to the minimum operating voltage required.

Table 3-1 shows the actions of the two resets on internal circuits, but not necessarily in order of occurrence (X indicates that the condition occurs for the particular reset).

3.3 INTERRUPTS

Systems often require that normal processing be interrupted so that some external event may be serviced. The MC68HC05L6 may be interrupted by one of four different methods; either one of three maskable hardware interrupts ($\overline{\text{IRQ}}$, SPI, or Timer) and one non-maskable software interrupt (SWI). Interrupts such as Timer and SPI have several flags which will cause the interrupt. Generally, interrupt flags are located in read-only status registers, whereas their equivalent enable bits are located in associated control registers. The interrupt flags and enable bits are never contained in the same register. If the enable bit is a logic zero it blocks the interrupt from occurring but does not inhibit the flag from being set. Reset clears all enable bits to preclude interrupts during the reset procedure.



*Internal timing signal and bus information not available externally.
**OSC1 line is not meant to represent frequency. It is only used to represent time.
***The next rising edge of the internal processor clock following the rising edge of $\overline{\text{RESET}}$ initiates the reset sequence.

Figure 3-1. Power-On Reset and $\overline{\text{RESET}}$

Table 3-1. Reset Action on Internal Circuit

Condition	$\overline{\text{RESET}}$ Pin	Power-On Reset
Timer Prescaler Reset to Zero State	X	X
Timer Counter Configured to $FFFC	X	X
Timer Output Compare (TCMP) Bit Reset to Zero	X	X
All Timer Interrupt Enable Bits Cleared (ICIE, OCIE, and TOIE) to Disable Timer Interrupts The OLVL Timer is Also Cleared by Reset	X	X
All Data Direction Registers Cleared to Zero (Input)	X	X
Configure Stack Pointer to $00FF	X	X
Force Internal Address Bus to Restart Vector ($1FFE-$1FFF)	X	X
Set I Bit in Condition Code Register to a Logic One	X	X
Clear STOP Latch	X*	X
Clear External Interrupt Latch	X	X
Clear WAIT Latch	X	X
Disable SPI (Serial Output Enable Control Bit SPE=0)—Other SPI Bits Cleared by Reset Include: SPIE, MSTR, SPIF, WCOL, and MODF	X	X
Clear Serial Interrupt Enable Bit (SPIE)	X	X
Place SPI System in Slave Mode (MSTR=0)	X	X

*Indicates that timeout still occurs.

The general sequence for clearing an interrupt is a software sequence of first accessing the status register while the interrupt flag is set, followed by a read or write of an associated register. When any of these interrupts occur, and if the enable bit is a logic one, normal processing is suspended at the end of the current instruction execution. Interrupts cause the processor registers to be saved on the stack (see Figure 2-6) and the interrupt mask (I bit) is set to prevent additional interrupts. The appropriate interrupt vector then points to the starting address of the interrupt service routine (refer to Figure 2-4 for vector location). Upon completion of the interrupt service routine, The RTI instruction (which is normally a part of the service routine) causes the register contents to be recovered from the stack followed by a return to normal processing. The stack order is shown in Figure 2-6.

NOTE

The interrupt mask bit (I bit) will be cleared if and only if the corresponding bit stored in the stack is zero.

A discussion of interrupts, plus a table listing vector addresses for all interrupts including reset is provided in Table 3-2.

Table 3-2. Vector Address for Interrupts and Reset

Register	Flag Name	Interrupts	CPU Interrupt	Vector Address
N/A	N/A	Reset	$\overline{\text{RESET}}$	$1FFE-$1FFF
N/A	N/A	Software	SWI	$1FFC-$1FFD
N/A	N/A	External Interrupt	$\overline{\text{IRQ}}$	$1FFA-$1FFB
Timer Status	ICF OCF TOF	Input Capture Output Compare Timer Overflow	TIMER	$1FF8-$1FF9
SPI Status	SPIF MODF	Transfer Complete Mode Fault	SPI	$1FF4-$1FF5

ware Controlled Interrupt Sequence

The following three functions ($\overline{\text{RESET}}$, STOP, and WAIT) are not in the strictest sense an interrupt; however, they are acted upon in a similar manner. Flowcharts for hardware interrupts are shown in Figure 3-2, and for STOP and WAIT are provided in Figure 3-3. A discussion is provided below.

(a) A low input on the $\overline{\text{RESET}}$ input pin causes the program to vector to its starting address which is specified by the contents of memory locations $1FFE and $1FFF. The I bit in the condition code register is also set.

(b) WAIT—the WAIT instruction causes all processor clocks to stop, but leaves the timer, SPI, and LCD clocks running. This rest state of the coprocessor can be cleared by reset, an external interrupt ($\overline{\text{IRQ}}$), Timer interrupt, or SPI interrupt. There are no special wait vectors for these individual interrupts.

(c) STOP—The STOP instruction causes the oscillator to be turned off and the processor to "sleep" until an external interrupt ($\overline{\text{IRQ}}$) or reset occurs.

3.3.2 Software Interrupt (SWI)

The software interrupt is an executable instruction. The action of the SWI instruction is similar to the hardware interrupts. The SWI is executed regardless of the state of the interrupt mask (I bit) in the condition code register. The interrupt service routine address is specified by the contents of memory location $1FFC and $1FFD.

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

If the interrupt mask (I bit) of the condition code register has been cleared and the external interrupt pin ($\overline{\text{IRQ}}$) has gone low, then the external interrupt is recognized. When the interrupt is recognized, the current state of the CPU is pushed onto the stack and the I bit is set. This masks further interrupts until the present one is serviced. The interrupt service routine address is specified by the contents of memory location $1FFA and $1FFB. Either a level-sensitive and negative edge-sensitive trigger, or a negative edge-sensitive only trigger are available as a mask option. Figure 3-4 shows both a functional and mode timing diagram for the interrupt line. The timing diagram shows two different treatments of the interrupt line ($\overline{\text{IRQ}}$) to the processor. The first method shows single pulses on the interrupt line spaced far enough apart to be serviced. The minimum time between pulses is a function of the number of cycles required to execute the interrupt service routine plus 21 cycles. Once a pulse occurs, the next pulse should not occur until the MCU software has exited the routine (an RTI occurs). The second configuration shows several interrupt lines "wire-ORed" to form the interrupts at the processor. Thus, if after servicing one interrupt the interrupt line remains low, then the next interrupt is recognized.

NOTE

The internal interrupt latch is cleared in the first part of the service routine; therefore, one (and only one) external interrupt pulse could be latched during t_{ILIL} and serviced as soon as the I bit is cleared.

3.3.4 Timer Interrupt

There are three different timer interrupt flags that will cause a timer interrupt whenever they are set and enabled. These three interrupt flags are found in the three most significant bits of the timer status register (TSR, location $13) and all three will vector to the same interrupt service routine ($1FF8–$1FF9).

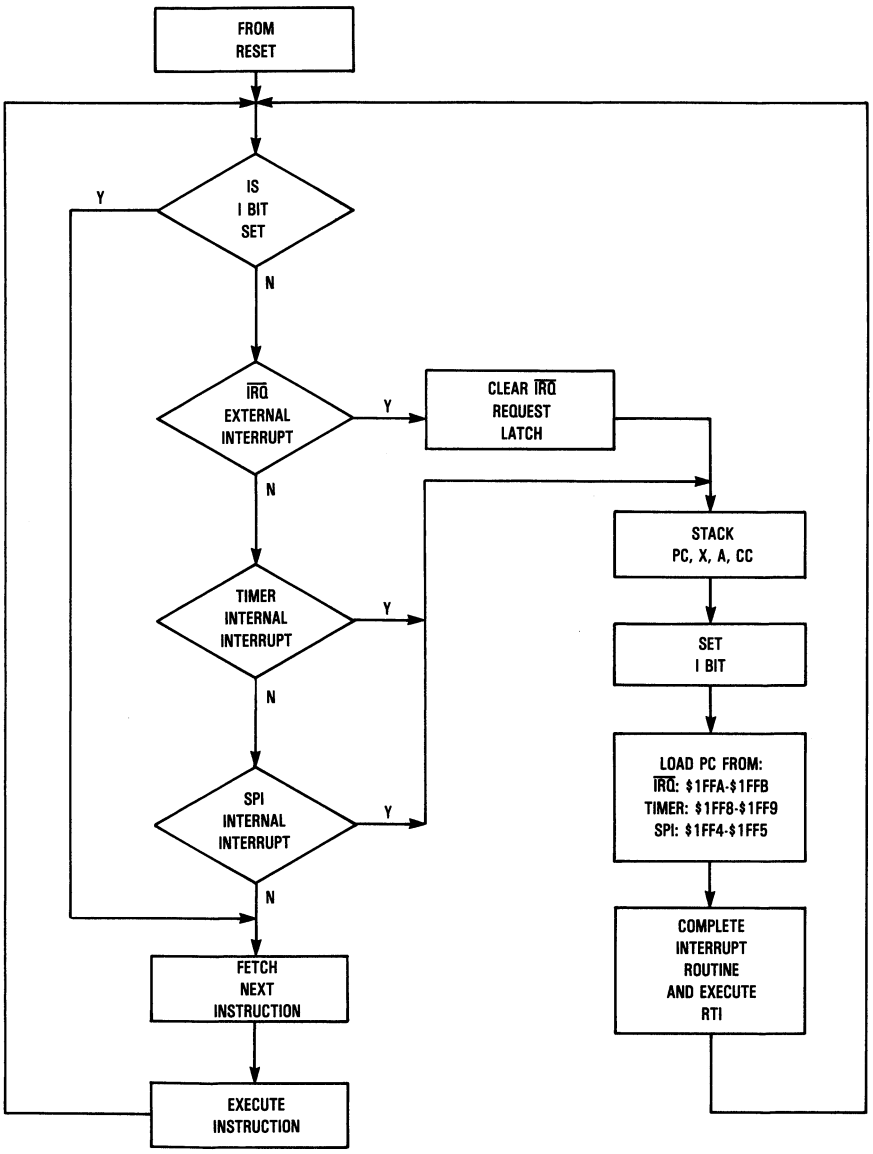


Figure 3-2. Hardware Interrupt Flowchart

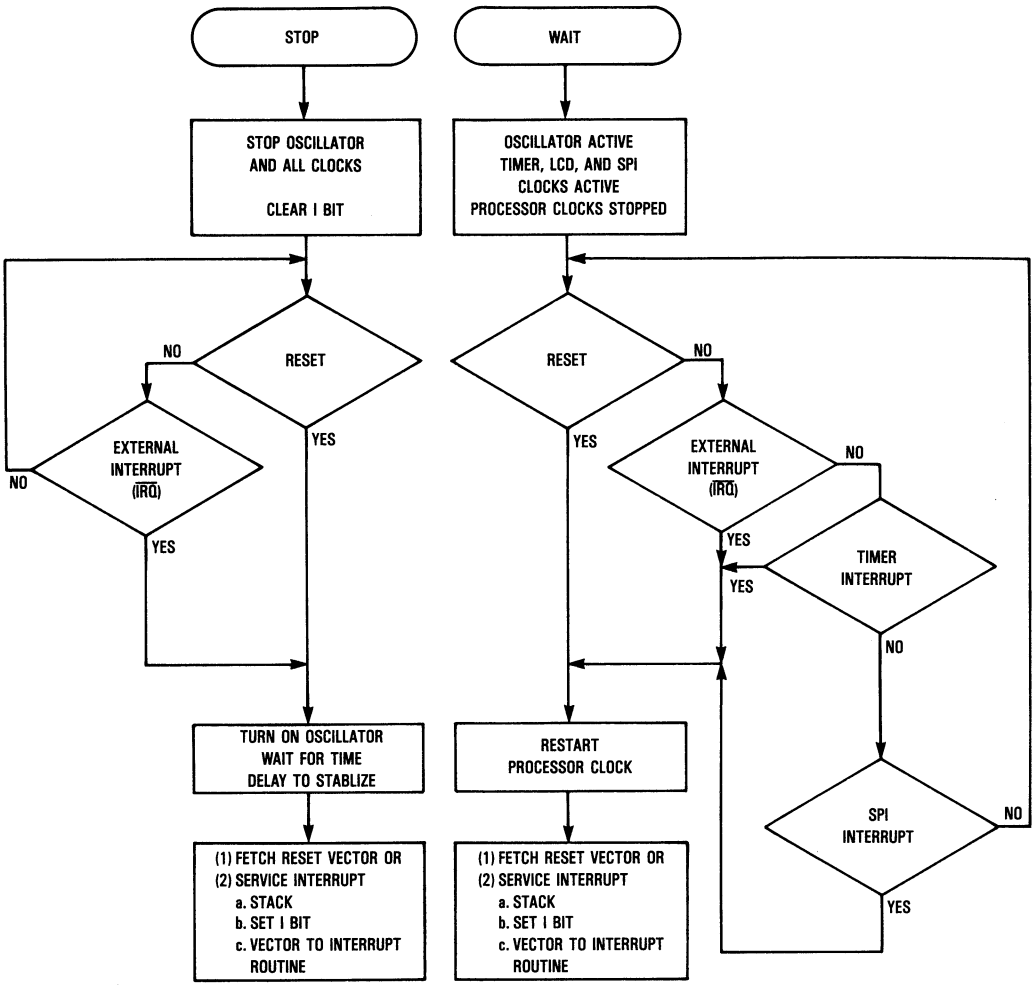
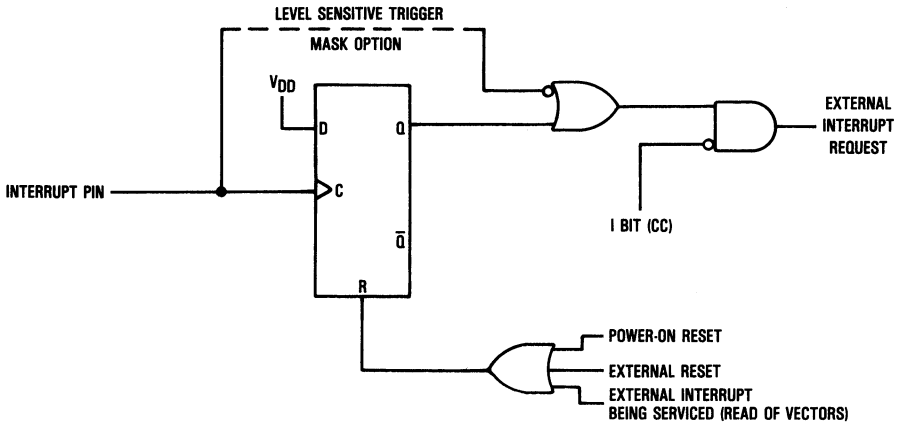
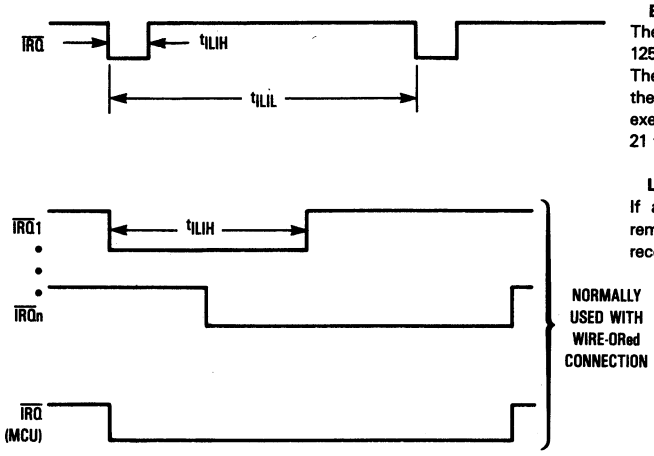


Figure 3-3. STOP/WAIT Flowcharts



(a) INTERRUPT FUNCTION DIAGRAM



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 21 t_{cyc} cycles.

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

(b) INTERRUPT MODE DIAGRAM

Figure 3-4. External Interrupt

t flags have corresponding enable bits (ICIE, OCIE, and TOIE) in the timer control register (TCR, location $12). Reset clears all enable bits, thus preventing an interrupt from occurring during the reset time period. The actual processor interrupt is generated only if the I bit in the condition code register is also cleared. When the interrupt is recognized, the current machine state is pushed onto the stack and the I bit is set. This masks further interrupts until the present one is serviced. The interrupt service routine address is specified by the contents of memory location $1FF8 and $1FF9. The general sequence for clearing an interrupt is a software sequence of accessing the status register while the flag is set, followed by a read or write of an associated register. Refer to **SECTION 4 PROGRAMMABLE TIMER** for additional information about the timer circuitry.

3.3.5 Serial Peripheral Interface (SPI) Interrupts

An interrupt in the serial peripheral interface (SPI) occurs when one of the interrupt flag bits in the serial peripheral status register (location $0B) is set, provided the I bit in the condition code register is clear and the enable bit in the serial peripheral control register (location $0A) is enabled. When the interrupt is recognized, the current state of the machine is pushed onto the stack and the I bit in the condition code register is set. This masks further interrupts until the present one is serviced. The SPI interrupt causes the program counter to vector to memory location $1FF4 and $1FF5 which contains the starting address of the interrupt service routine. Software in the serial peripheral interrupt service routine must determine the priority and cause of the SPI interrupt by examining the interrupt flag bits located in the SPI status register. The general sequence for clearing an interrupt is a software sequence of accessing the status register while the flag is set, followed by a read or write of an associated register. Refer to **SECTION 5 SERIAL PERIPHERAL INTERFACE** for a description of the SPI system and its interrupts.

3.4 LOW POWER MODES

Two low power consumption modes of operation are available: wait and stop. These operating modes are initiated by using the WAIT and STOP instructions.

3.4.1 WAIT Instruction

The WAIT instruction places the microcomputer in a low power consumption mode. In the WAIT mode, the internal clock remains active, and all CPU processing is stopped; however, the programmable timer and serial peripheral interface remain active. Refer to Figure 3-3. During the WAIT mode, the I bit in the condition code register is cleared to enable all interrupts. All other registers and memory remain unaltered and all parallel input/output lines remain unchanged. This operating mode will continue until any interrupt or reset is sensed. At this time the program counter vectors to the memory location ($1FFA through $1FFF) which contains the starting address of the interrupt or reset service routine.

3.4.2 STOP Instruction

The STOP instruction places the microcomputer in its lowest power consumption mode. The internal oscillator is turned off, causing all internal processing to be halted; refer to Figure 3-3. During the STOP mode, the I bit in the condition code register is cleared to enable external interrupts. All other registers and memory remain unaltered and all input/output lines remain unchanged. This continues until an external interrupt ($\overline{IRQ}$) or reset is sensed at which time the internal oscillator is turned on. The external interrupt or reset causes the program counter to vector to memory location $1FFA and $1FFB or $1FFE and $1FFF which contains the starting address of the interrupt or reset service routine respectively.

SECTION 4
PROGRAMMABLE TIMER

4.1 INTRODUCTION

The programmable timer, which is preceded by a fixed divide-by-four prescaler, can be used for many purposes, including input waveform measurements while simultaneously generating an output waveform. Pulse widths can vary from several microseconds to many seconds. A block diagram of the timer is shown in Figure 4-1 and timing diagrams are shown in Figures 4-2 through 4-5.

Because the timer has a 16-bit architecture, each specific functional segment (capability) is represented by two registers. These registers contain the high and low byte of that functional segment. Generally, accessing the low byte of a specific timer function allows full control of that function; however, an access of the high byte inhibits that specific timer function until the low byte is also accessed.

NOTE

The I bit in the condition code register should be set while manipulating both the high and low byte register of a specific timer function to ensure that an interrupt does not occur. This prevents interrupts from occurring between the time that the high and low bytes are accessed.

The programmable timer capabilities are provided by using the following ten addressable 8-bit registers (note that the high and low represent the significance of the byte).

Timer Control Register (TCR)	location $12,
Timer Status Register (TSR)	location $13,
Input Capture High Register	location $14,
Input Capture Low Register	location $15,
Output Compare High Register	location $16,
Output Compare Low Register	location $17,
Counter High Register	location $18,
Counter Low Register	location $19,
Alternate Counter High Register	location $1A,
Alternate Counter Low Register	location $1B.

A description of each register is provided in the following paragraphs.

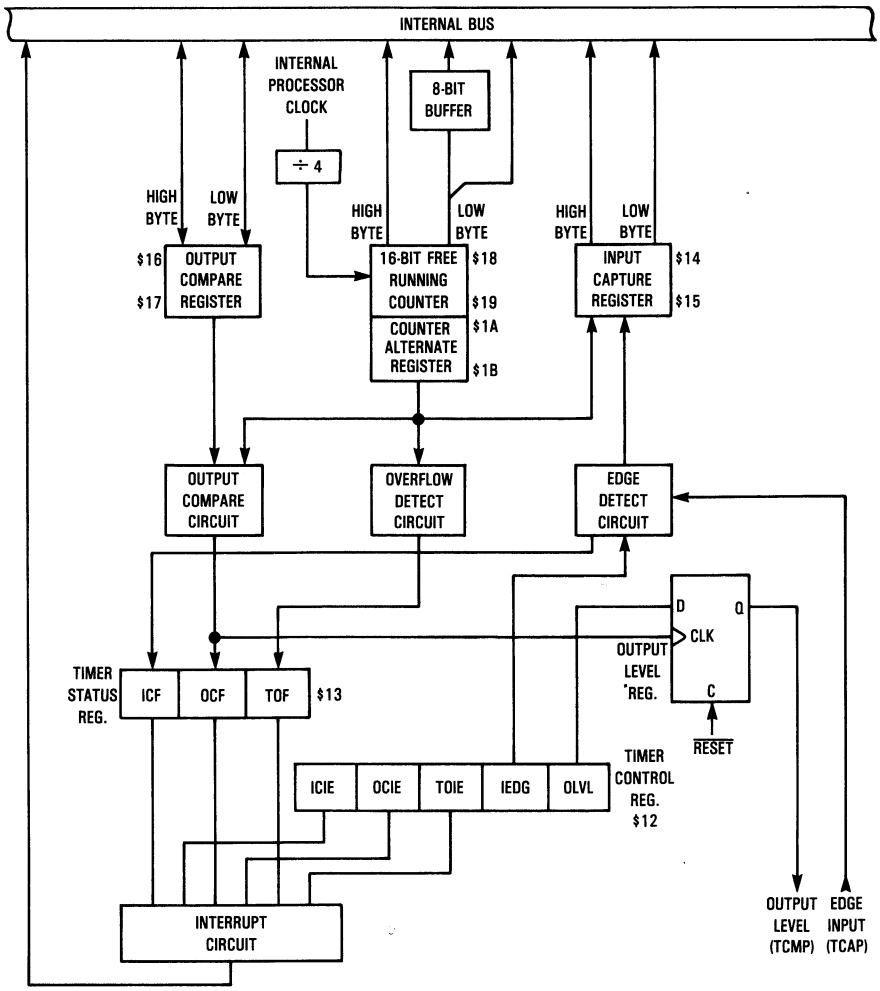
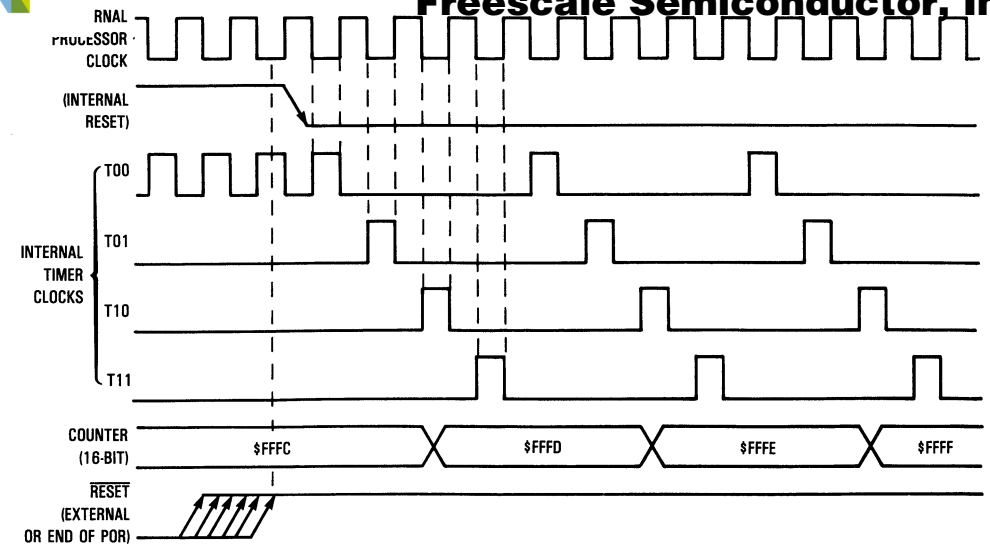
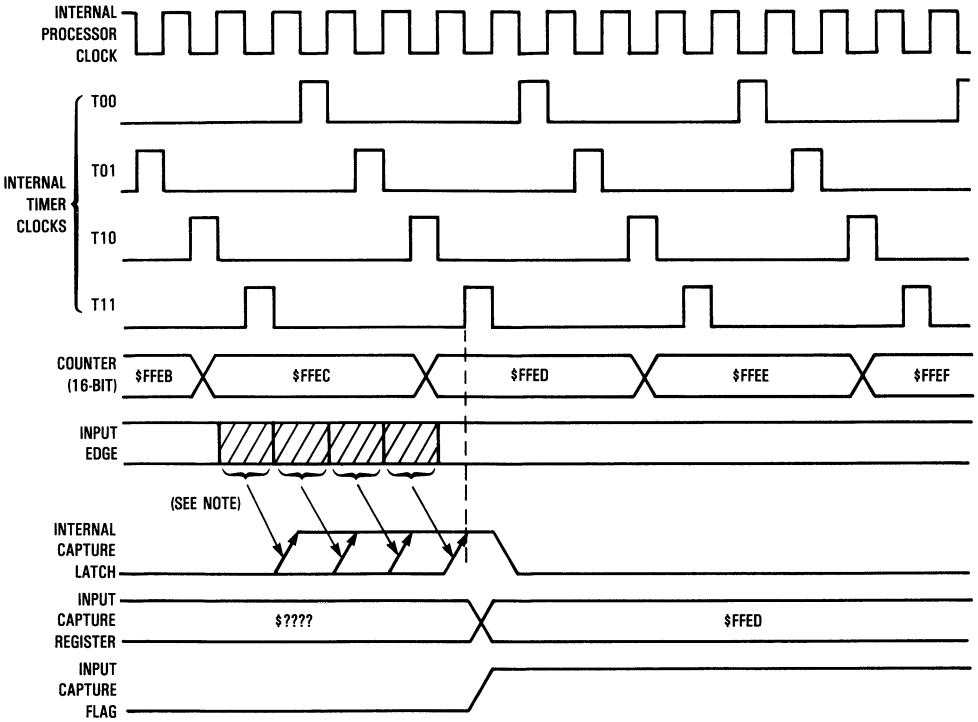


Figure 4-1. Programmable Timer Block Diagram



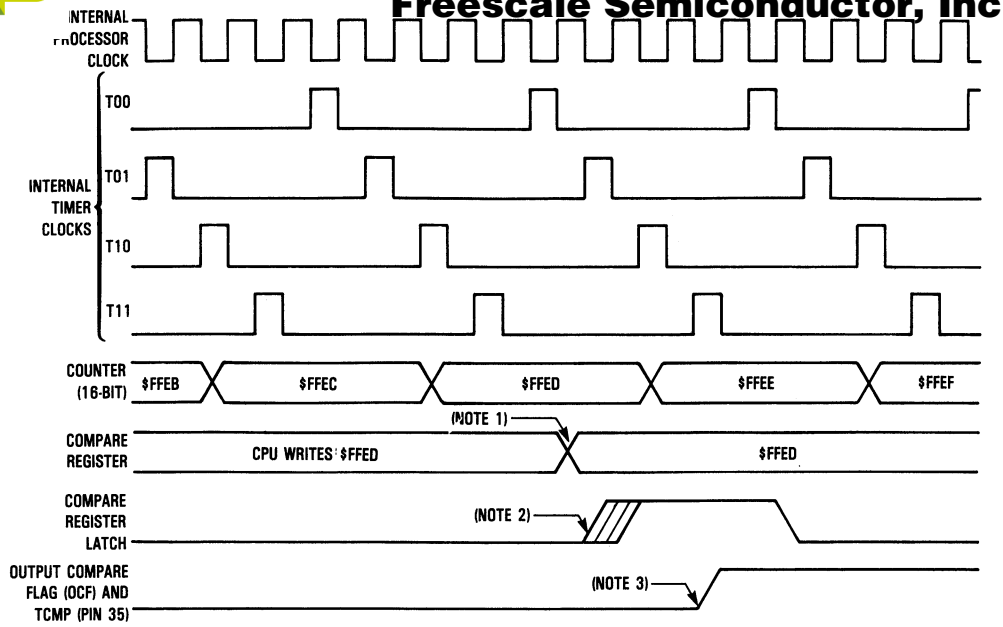
NOTE: The Counter Register and Timer Control Register are the only ones affected by $\overline{\text{RESET}}$.

Figure 4-2. Timer State Timing Diagram for Reset



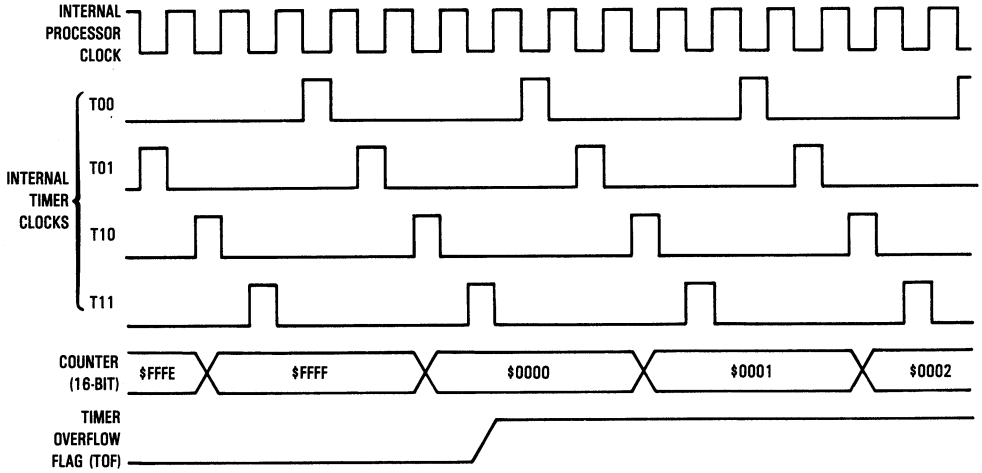
NOTE: If the input edge occurs in the shaded area from one timer state T10 to the other timer state T10 the input capture flag is set during the next state T11.

Figure 4-3. Timer State Timing Diagram for Input Capture



NOTES:
1. The CPU write to the compare register may take place at any time, but a compare only occurs at timer state T01. Thus, a 4-cycle difference may exist between the write to the compare register and the actual compare.
2. Internal compare takes place during timer state T01.
3. OCF is set at the timer state T11 which follows the comparison match ($FFED in this example).

Figure 4-4. Timer State Timing Diagram for Output Compare



NOTE: The TOF bit is set at timer state T11 (transition of counter from $FFFF to $0000). It is cleared by a read of the timer status register during the internal processor clock high time followed by a read of the counter low register.

Figure 4-5. Timer State Diagram for Timer Overflow

CONTROL REGISTER (TCR)

The timer control register is an 8-bit read/write register which contains five control bits. Three of these bits control interrupts associated with each of the three flag bits found in the timer status register (discussed below). The other two bits control: 1) which edge is significant to the input capture edge detector (i.e., negative or positive), and 2) the next value to be clocked to the output level register in response to a successful output compare. The timer control register and the free running counter are the only sections of the timer affected by reset. The TCMP pin is forced low during external reset and stays low until a valid compare changes it to a high. The timer control register is illustrated below followed by a definition of each bit.

	7	6	5	4	3	2	1	0
$0012	ICIE	OCIE	TOIE	0	0	0	IEDG	OLVL

ICIE—Input Capture Interrupt Enable

If the input capture interrupt enable bit is set, a timer interrupt is enabled when the ICF status flag (in the timer status register) is set. If the ICIE bit is clear, the interrupt is inhibited. The ICIE bit is cleared by reset.

OCIE—Input Capture Interrupt Enable

If the output compare interrupt enable bit is set, a timer interrupt is enabled whenever the OCF status flag is set. If the OCIE bit is clear, the interrupt is inhibited. The OCIE bit is cleared by reset.

TOIE—Timer Overflow Interrupt Enable

If the timer overflow interrupt enable bit is set, a timer interrupt is enabled whenever the TOF status flag (in the timer status register) is set. If the TOIE bit is clear, the interrupt is inhibited. The TOIE bit is cleared by reset.

IEDG—Input Edge

The value of the input edge bit determines which level transition will trigger a free running counter transfer to the input capture register. Reset does not affect the IEDG bit.

0 = Negative Edge
1 = Positive Edge

OLVL—Output Level

The value of the output level bit is clocked into the output level register by the next successful output compare. This bit and the output level register are cleared by reset.

0 = Low Output
1 = High Output

4.3 TIMER STATUS REGISTER (TSR)

The timer status register is an 8-bit register of which the three most significant bits contain read-only status information. These three bits indicate the following:

1. A proper transition has taken place with an accompanying transfer of the free running counter contents to the input capture register,
2. A match has been found between the free running counter and the output compare register, and
3. A free running counter transition from $FFFF to $0000 has been sensed (timer overflow).

The timer status register is illustrated below followed by a definition of each bit. Refer to timing diagrams shown in Figures 4-2, 4-3, and 4-4 for timing relationship to the timer status register bits.

	7	6	5	4	3	2	1	0
$0013	ICF	OCF	TOF	0	0	0	0	0

ICF—Input Capture Flag

The input capture flag bit is set when a proper edge has been sensed by the input capture edge detector. It is cleared by a processor access of the timer status register (with ICF set) followed by accessing the low byte ($15) of the input capture register. Reset does not affect the input compare flag.

OCF—Output Capture Flag

The output compare flag bit is set when the output compare register contents matches the contents of the free running counter. The OCF is cleared by accessing the timer status register (with OCF set) and then accessing the low byte ($17) of the output compare register. Reset does not affect the output compare flag.

TOF—Timer Overflow Flag

The timer overflow flag bit is set by a transition of the free running counter from $FFFF to $0000. It is cleared by accessing the timer status register (with TOF set) followed by an access of the free running counter least significant byte ($19). Reset does not affect the TOF bit.

Accessing the timer status register satisfies the first condition required to clear any status bits which happen to be set during the access. The only remaining step is to provide an access of the register which is associated with the status bit. Typically, this presents no problem for the input capture and output compare functions.

A problem can occur when using the timer overflow function and reading the free running counter at random times to measure an elapsed time. Without incorporating the proper precautions into software, the timer overflow flag could unintentionally be cleared if: 1) the timer status register is read or written when TOF is set, and 2) the least significant byte of the free running counter is read but not for the purpose of servicing the flag. The counter alternate register at address $1A and $1B contains the same value as the free running counter (at address $18 and $19); therefore, this alternate register can be read at any time without affecting the timer overflow flag in the timer status register.

4.4 INPUT CAPTURE REGISTER

The two 8-bit registers which make up the 16-bit input capture register are read-only and are used to latch the value of the free running counter after a defined transition is sensed by the corresponding input capture edge detector. The level transition which triggers the counter transfer is defined by the corresponding input edge bit (IEDG). Reset does not affect the contents of the input capture register.

The result obtained by an input capture will be one more than the value of the free running counter on the rising edge of the internal processor clock preceding the external transition (refer to timing diagram shown in Figure 4-3). This delay is required for internal synchronization. Resolution is affected by the prescaler allowing the timer to only increment every four internal processor clock cycles.

The free running counter contents are transferred to the input capture register on each proper signal transition regardless of whether the input capture flag (ICF) is set or clear. The input capture register always contains the free running counter value which corresponds to the most recent input capture.

After a read of the most significant byte of the input capture register ($14), counter transfer is inhibited until the least significant byte ($15) of the input capture register is also read. This characteristic forces the minimum pulse period attainable to be determined by the time used in the capture software routine and its interaction with the main program. The free running counter increments every four internal processor clock cycles due to the prescaler.

least significant byte ($15) of the input capture register does not inhibit the free running counter transfer. Again, minimum pulse periods are ones which allow software to read the least significant byte ($15) and perform needed operations. There is no conflict between the read of the input capture register and the free running counter transfer since they occur on opposite edges of the internal processor clock.

4.5 OUTPUT COMPARE REGISTER

The output compare register is a 16-bit register, which is made up of two 8-bit registers at locations $16 (most significant byte) and $17 (least significant byte). The output compare register can be used for several purposes such as, controlling an output waveform or indicating when a period of time has elapsed. The output compare register is unique in that all bits are readable and writable and are not altered by the timer hardware. Reset does not affect the contents of this register and if the compare function is not used, the two bytes of the output compare register can be used as storage locations.

The contents of the output compare register are compared with the contents of the free running counter once during every four internal processor clocks. If a match is found, the corresponding output compare flag (OCF) bit is set and the corresponding output level (OLVL) bit is clocked (by the output compare circuit pulse) to an output level register. The values in the output compare register and the output level bit should be changed after each successful comparison in order to control an output waveform or establish a new elapsed timeout. An interrupt can also accompany a successful output compare provided the corresponding interrupt enable bit, OCIE, is set.

After a processor write cycle to the output compare register containing the most significant byte ($16), the output compare function is inhibited until the least significant byte ($17) is also written. The user must write both bytes (locations) if the most significant byte is written first. A write made only to the least significant byte ($17) will not inhibit the compare function. The free running counter is updated every four internal processor clock cycles due to the internal prescaler. The minimum time required to update the output compare register is a function of the software program rather than the internal hardware.

A processor write may be made to either byte of the output compare register without affecting the other byte. the output level (OLVL) bit is clocked to the output level register regardless of whether the output compare flag (OCF) is set or clear.

Because neither the output compare flag (OCF bit) or output compare register is affected by reset, care must be exercised when initializing the output compare function with software. The following procedure is recommended.

(1) Write the high byte of the output compare register to inhibit further compares until the low byte is written.
(2) Read the timer status register to arm the OCF if it is already set.
(3) Write the output compare register low byte to enable the output compare function with the flag clear.

The advantage of this procedure is to prevent the OCF bit from being set between the time it is read and the write to the output compare register. A software example is shown below.

```
B7   16   STA   OCMPHI   INHIBIT OUTPUT COMPARE
B6   13   LDA   TSTAT    ARM OCF BIT IF SET
BF   17   STX   OCMPLD   READY FOR NEXT COMPARE
```

The key element in the programmable timer is a 16-bit free running counter, or counter register, preceded by a prescaler which divides the internal processor clock by four. The prescaler gives the timer a resolution of 2.0 microseconds if the internal processor clock is 2.0 MHz. The counter is clocked to increasing values during the low portion of the internal processor clock. Software can read the counter at any time without affecting its value.

The double byte free running counter can be read from either of two locations $18-$19 (called counter register at this location), or $1A-$1B (counter alternate register at this location). A read sequence containing only a read of the least significant byte of the free running counter ($19, $1B) will receive the count value at the time of the read. If a read of the free running counter or counter alternate register first addresses the most significant byte ($18, $1A) it causes the least significant byte ($19,$1B) to be transferred to a buffer. This buffer value remains fixed after the first most significant byte "read" even if the user reads the most significant byte several times. This buffer is accessed when reading the free running counter or counter alternate register least significant byte ($19 or $1B), and thus completes a read sequence of the total counter value. Note that in reading either the free running counter or counter alternate register, if the most significant byte is read, the least significant byte must also be read in order to complete the sequence.

The free running counter is configured to $FFFC during reset and is always a read-only register. During a power-on reset, the counter is also configured to $FFFC and begins running after the oscillator startup delay. Because the free running counter is 16 bits preceded by a fixed divide-by-four prescaler, the value in the free running counter repeats every 262,144 processor internal processor clock cycles. When the counter rolls over from $FFFF to $0000, the timer overflow flag (TOF) bit is set. An interrupt can also be enabled when counter rollover occurs by setting its interrupt enable bit (TOIE).

4.7 TIMER DURING WAIT MODE

When the MCU enters the wait mode, the CPU clock is halted. All CPU action is suspended; however, the timer remains active. In fact an interrupt from the timer (in addition to a logic low on the $\overline{IRQ}$ or $\overline{RESET}$ pins) causes the processor to exit the wait mode.

4.8 TIMER DURING STOP MODE

When the MCU enters the stop mode, the timer counter stops counting (the internal processor clock is stopped) and remains at that particular count value until the stop mode is exited by an interrupt (if exited by reset the counter is forced to $FFFC). If the stop mode is exited by an external low on the $\overline{IRQ}$ pin, then the counter resumes from its stopped value as if nothing had happened.

Another feature of the programmable timer, in the stop mode, is that if at least one valid input capture edge occurs at the TCAP pin, the input capture detect circuitry is armed. This action does not set any timer flags or "wake up" the MCU, but when the MCU does "wake up" there will be an active input capture flag (and data) from that first valid edge which occurred during the stop mode.

If the stop mode is exited by an external reset (logic low on $\overline{RESET}$ pin), then no such input capture flag or data action takes place even if there was a valid input capture edge (at the TCAP pin) during the MCU stop mode.

SECTION 5
SERIAL PERIPHERAL INTERFACE (SPI)

5.1 INTRODUCTION

The serial peripheral interface (SPI) is an interface built into the MC68HC05L6 microcomputer which allows several SPI microcomputers, or SPI-type peripherals to be interconnected within a single "black box" or on the same printed circuit board. In a serial peripheral interface, separate wires (signals) are required for data and clock. In the SPI format, the clock is not included in the data stream and must be furnished as a separate signal. An SPI system may be configured as one containing one master MCU and several slave MCUs, or as a system in which an MCU is capable of being either a master or a slave.

Figure 5-1 illustrates two different system configurations. Figure 5-1(a) represents a system of five different microcomputers in which there is one master and four slaves (0, 1, 2, 3). In this system four basic lines (signals) are required for the MOSI (master out slave in), MISO (master in slave out), SCK (serial clock), and $\overline{SS}$ (slave select) lines. Figure 5-1(b) represents a system of five microcomputers in which three can be either master or slave and two are slave only.

Features of the SPI include:

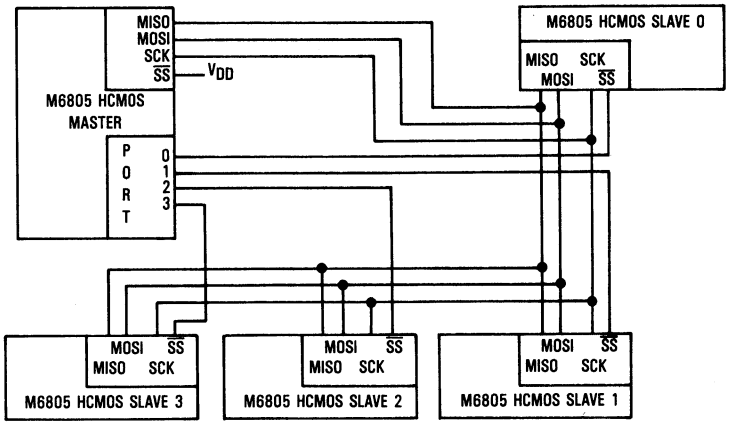
- Full Duplex, Three-Wire Synchronous Transfers
- Master or Slave Operation
- 1.05 MHz (Maximum) Master Bit Frequency
- 2.1 MHz (Maximum) Slave Bit Frequency
- Four Programmable Master Bit Rates
- Programmable Clock Polarity and Phase
- End of Transmission Interrupt Flag
- Write Collision Flag Protection
- Master-Master Mode Fault Protection Capability

5.2 SPI SIGNAL DESCRIPTION

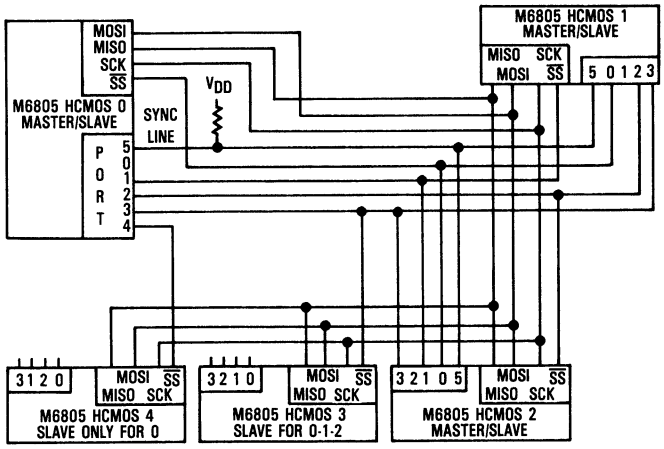
The four basic SPI signals (MOSI, MISO, SCK, and $\overline{SS}$) are described in the following paragraphs. Each signal is described for both the master and slave mode.

5.2.1 Master Out Slave In (MOSI)

The MOSI pin is configured as an output in a master (mode) device and as an input in a slave (mode) device. Data is transferred serially from a master to a slave on this line, most significant bit first. The timing diagram of Figure 5-2 shows the relationship between data and serial clock (SCK). As shown in Figure 5-2, four possible timing relationships may be chosen by using control bits CPOL and CPHA. The master device always allows data to be applied on the MOSI line a half-cycle before the serial clock edge (SCK) in order for the slave device to latch the data.



(a) SINGLE MASTER, FOUR SLAVES



(b) THREE MASTER/SLAVE, TWO SLAVES

Figure 5-1. Master-Slave System Configuration

NOTE

Both the slave device(s) and a master device must be programmed to similar timing modes for proper data transfer.

When the master device transmits data to a slave device via the MOSI line, the slave device responds by sending data to the master device via the MISO line. This implies full duplex transmission with both data out and data in synchronized with the same clock signal (one which is provided by the master device). Thus, the byte transmitted is replaced by the byte received and eliminates the need for separate transmit-empty and receiver-full status bits. A single status bit (SPIF) in the serial peripheral status register (SPSR, location $0B) is used to signify that the I/O operation is complete.

Configuration of the MOSI pin is a function of the MSTR bit in the serial peripheral control register (SPCR, location $0A). When a device is operating as a master, the MOSI pin is an output because the program in firmware sets the MSTR bit to a logic one.

5.2.2 Master In Slave Out (MISO)

The MISO pin is configured as an input in a master (mode) device and as an output in a slave (mode) device. Data is transferred serially from a slave to a master on this line, most significant bit first. The MISO pin of a slave device is placed in the high-impedance state if it is not selected by the master; i.e., its $\overline{SS}$ pin is a logic one. The timing diagram in Figure 5-2 shows the relationship between data and the serial clock (SCK). As shown in Figure 5-2, four possible timing relationships may be chosen by using control bits CPOL and CPHA. The master device always allows data to be applied on the MISO line a half-cycle before the serial clock edge (SCK) in order for the slave device to latch the data.

NOTE

Both the slave device(s) and a master device must be programmed to similar timing modes for proper data transfer.

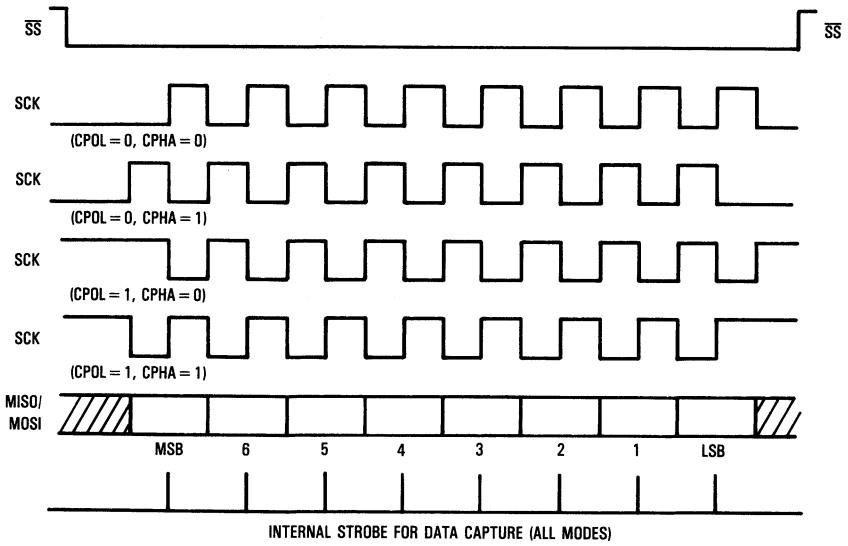


Figure 5-2. Data Clock Timing Diagram

laster device transmits data to a slave device via the MOSI line, the slave device responds by sending data to the master device via the MISO line. This implies full duplex transmission with both data out and data in synchronized with the same clock signal (one which is provided by the master device). Thus, the byte transmitted is replaced by the byte received and eliminates the need for separate transmit-empty and receiver-full status bits. A single status bit (SPIF) in the serial peripheral status register (SPSR, location $0B) is used to signify that the I/O operation is complete.

In the master device, the MSTR control bit in the serial peripheral control register (SPCR,location $0A) is set to a logic one (by the program) to allow the master device to receive data on its MISO pin. In the slave device, its MISO pin is enabled by the logic level of the $\overline{SS}$ pin; i.e., if $\overline{SS}=1$ then the MISO pin is placed in the high-impedance state, whereas, if $\overline{SS}=0$ the MISO pin is an output for the slave device.

5.2.3 Slave Select ($\overline{SS}$)

The slave select ($\overline{SS}$) pin is a fixed input which receives an active low signal that is generated by the master device to enable slave device(s) to accept data. To ensure that data will be accepted by a slave device, the $\overline{SS}$ signal line must be a logic low prior to occurrence of serial SCK and must remain low until after the last (eighth) SCK cycle. Figure 5-2 illustrates the relationship between SCK and the data for two different level combinations of CPHA, when $\overline{SS}$ is pulled low. These are: 1) with CPHA = 1 or 0, the first bit or data is applied to the MISO line for transfer, and 2) when CPHA = 0 the slave device is prevented from writing to its data register. Refer to the WCOL status flag in the serial peripheral status register (location $0B) description for further information on the effects that the $\overline{SS}$ input and CPHA control bit have on the I/O data register. A high level $\overline{SS}$ signal forces the MISO line to the high-impedance state. Also, SCK and the MOSI line are ignored by a slave device when its $\overline{SS}$ signal is high.

When a device is a master, it constantly monitors its $\overline{SS}$ signal input for a logic low. The master device will become a slave device any time its $\overline{SS}$ signal input is detected low. This ensures that there is only one master controlling the $\overline{SS}$ line for a particular system. When the $\overline{SS}$ line is detected low, it clears the MSTR control bit (serial peripheral control register, location $0A). Also, control bit SPE in the serial peripheral control register is cleared and causes the serial peripheral interface to be disabled. The MODF flag bit in the serial peripheral status register (location $0B) is also set to indicate to the master device that another device is attempting to become a master. Two devices attempting to be outputs are normally the result of a software error; however, a system could be configured which would contain a default master which would automatically "take-over" and restart the system. Refer to **SERIAL PERIPHERAL INTERFACE SYSTEM CONSIDERATIONS**.

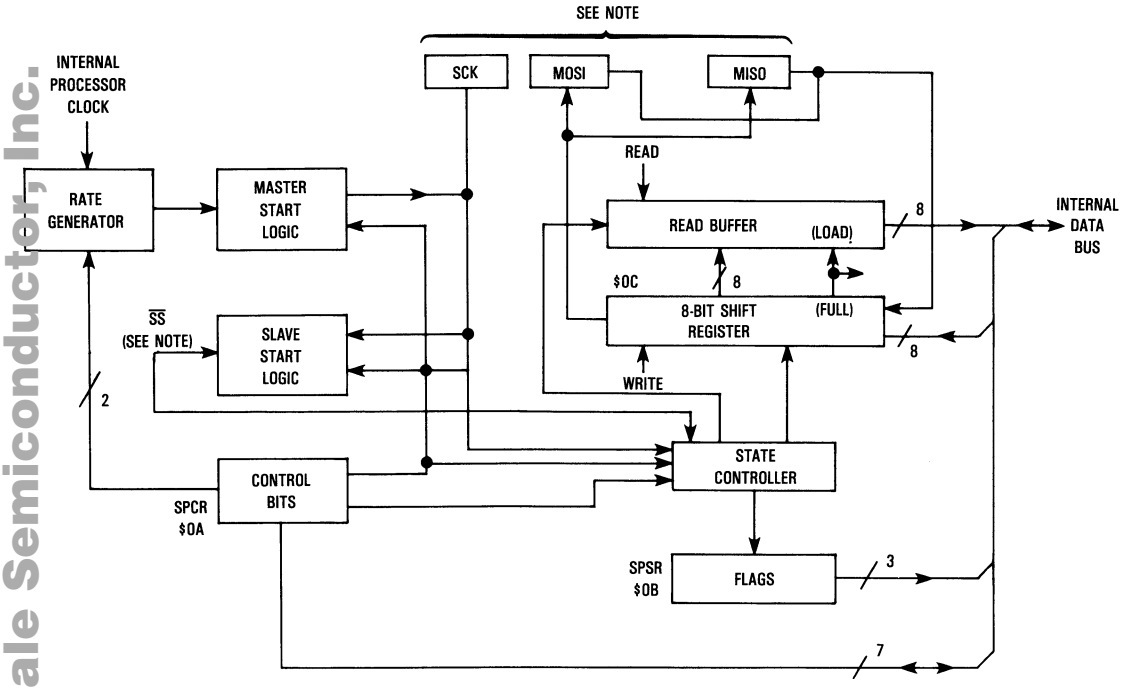
5.2.4 Serial Clock (SCK)

The serial clock is used to synchronize the movement of data both in and out of the device through its MOSI and MISO pins. The master and slave devices are capable of exchanging a data byte of information during a sequence of eight clock pulses. Since SCK is generated by the master device, the SCK line becomes an input on all slave devices and synchronizes slave data transfer. The type of clock and its relationship to data are controlled by the CPOL and CPHA bits in the serial peripheral control register (location $0A). Refer to Figure 5-2 for timing.

The master device generates the SCK through a circuit driven by the internal processor clock. Two bits (SPR0 and SPR1) in the serial peripheral control register (location $0A) of the master device select the clock rate. The master device uses the SCK to latch incoming slave device data on the MISO line and shifts out data to the slave device on the MOSI line. Both master and slave devices must be operated in the same timing mode as controlled by the CPOL and CPHA bit in the serial peripheral control register. In the slave device, SPR0, SPR1 have no effect on the operation of the serial peripheral interface. Timing is shown in Figure 5-2.

A block diagram of the serial peripheral interface is shown in Figure 5-3. In a master configuration, the master start logic receives an input from the CPU (in the form of a write to the SPI rate generator) and originates the serial clock (SCK) based on the internal processor clock. This clock is also used internally to control the state controller as well as the 8-bit shift register. As a master device, data is parallel loaded into the 8-bit shift register (from the internal bus) during a write cycle and then shifted out serially to the MOSI pin for application to the slave device(s). During a read cycle, data is applied serially from a slave device via the MISO pin to the 8-bit shift register. After the 8-bit shift register is loaded, its data is parallel transferred to the read buffer and then is made available to the internal data bus during a CPU read cycle.

In a slave configuration, the slave start logic receives a logic low (from a master device) at the $\overline{SS}$ pin and a serial clock input (from the same master device) at the SCK pin. Thus, the slave is synchronized with the master. Data from the master is received serially at the slave MOSI pin and loads the 8-bit shift register. After the 8-bit shift register is loaded, its data is parallel transferred to the read buffer and then is made available to the internal data bus during a CPU read cycle. During a write cycle, data is parallel loaded into the 8-bit shift register from the internal data bus and then shifted out serially to the MISO pin for application to the master device.



NOTES:
The $\overline{SS}$, SCK, MOSI, and MISO are external pins which provide the following functions:
a. MOSI—Provides serial output to slave unit(s) when device is configured as a master. Receives serial input from master unit when device is configured as a slave unit.
b. MISO—Receives serial input from slave unit(s) when device is configured as a master. Provides serial output to master when device is configured as a slave unit.
c. SCK—Provides system clock when device is configured as a master unit. Receives system clock when device is configured as a slave unit.
d. $\overline{SS}$—Provides a logic low to select a slave device for a transfer with a master device.

Figure 5-3. Serial Peripheral Interface Block Diagram

ustrates the MOSI, MISO, and SCK master-slave interconnections. Note that in Figure 5-4 the master $\overline{SS}$ pin is tied to a logic high and the slave $\overline{SS}$ pin is a logic low. Figure 5-1 provides a larger system connection for these same pins. Note that in Figure 5-1, all $\overline{SS}$ pins are connected to a port pin of a master/slave device. In this case any of the devices can be a slave.

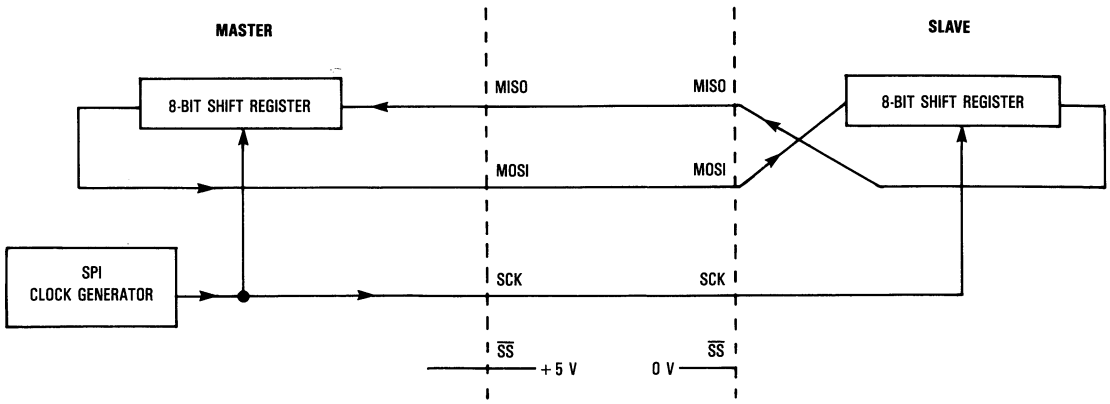


Figure 5-4. Serial Peripheral Interface Master-Slave Interconnection

5.4 REGISTERS

There are three registers in the serial parallel interface which provide control, status, and data storage functions. These registers are called the serial peripheral control register (SPCR, location $0A), serial peripheral status register (SPSR, location $0B), and serial peripheral data I/O register (SPDR, location $0C) are described in the following paragraphs.

5.4.1 Serial Peripheral Control Register (SPCR)

	7	6	5	4	3	2	1	0
$000A	SPIE	SPE	–	MSTR	CPOL	CPHA	SPR1	SPR0

The serial peripheral control register bits are defined as follows:

SPIE—Serial Peripheral Interrupt Enable

When the serial peripheral interrupt enable bit is high, it allows the occurrence of a processor interrupt, and forces the proper vector to be loaded into the program counter if the serial peripheral status register flag bit (SPIF and/or MODF) is set to a logic one. It does not inhibit the setting of a status bit. The SPIE bit is cleared by reset.

SPE—Serial Peripheral Output Enable

When the serial peripheral output enable control bit is set, all output drive is applied to the external pins and the system is enabled. When the SPE bit is set, it enables the SPI system by connecting it to the external pins thus allowing it to interface with the external SPI bus. The pins that are defined as output depend on which mode (master or slave) the device is in. Because the SPI bit is cleared by reset, the SPI system is not connected to the external pins upon reset.

MSTR—Master

The master bit determines whether the device is a master or a slave. If the MSTR bit is a logic zero it indicates a slave device and a logic one denotes a master device. If the master mode is selected, the function of the SCK pin changes from an input to an output and the function of the MSIO and

ıs are reversed. This allows the user to wire device pins MISO to MISO, and MOSI to MOSI, and SCK to SCK without incident.The MSTR bit is cleared by reset; therefore, the device is always placed in the slave mode during reset.

CPOL—Clock Polarity

The clock polarity bit controls the normal or steady state value of the clock when data is not being transferred. The CPOL bit affects both the master and slave modes. It must be used in conjunction with the clock phase control bit (CPHA) to produce the wanted clock-data relationship between a master and a slave device. When the CPOL bit is a logic zero, it produces a steady state low value at the SCK pin of the master device. If the CPOL bit is a logic one, a high value is produced at the SCK pin of the master device when data is not being transferred. The CPOL bit is not affected by reset. Refer to Figure 5-2.

CPHA—Clock Phase

The clock phase bit controls the relationship between the data on the MISO and MOSI pins and the clock produced or received at the SCK pin. This control has effect in both the master and slave modes. It must be used in conjunction with the clock polarity control bit (CPOL) to produce the wanted clock-data relation. The CPHA bit in general selects the clock edge which captures data and allows it to change states. It has its greatest impact on the first bit transmitted (MSB) in that it does or does not allow a clock transition before the first data capture edge. The CPHA bit is not affected by reset. Refer to Figure 5-2.

SPR0, SPR1—Serial Peripheral Rate

These two serial peripheral rate bits select one of four baud rates to be used as SCK if the device is a master; however, they have no effect in the slave mode. The slave device is capable of shifting data in and out at a maximum rate which is equal to the CPU clock. A rate table is given below for the generation of the SCK from the master. The SPR1 and SPR0 bits are not affected by reset.

SPR1	SPR0	Internal Processor Clock Divide By
0	0	2
0	1	4
1	0	16
1	1	32

5.4.2 Serial Peripheral Status Register (SPSR)

	7	6	5	4	3	2	1	0
$000B	SPIF	WCOL	–	MODF	–	–	–	–

The status flags which generate a serial peripheral interface (SPI) interrupt may be blocked by the SPIE control bit in the serial peripheral control register. The WCOL bit does not cause an interrupt. The serial peripheral status register bits are defined as follows:

SPIF—Serial Peripheral Data Transfer Rate

The serial peripheral data transfer flag bit notifies the user that a data transfer between the device and an external device has been completed. With the completion of the data transfer, SPIF is set, and if SPIE is set, a serial peripheral interrupt is generated. During the clock cycle that SPIF is being set, a copy of the received data byte in the shift register is moved to a buffer. When the data register is read, it is the buffer that is read. During an overrun condition, when the master device has sent several bytes of data and the slave device has not responded to the first SPIF, only the first byte sent is contained in the receiver buffer and all other bytes are lost.

nsfer of data is initiated by the master device writing its serial peripheral data register.

Clearing the SPIF bit is accomplished by a software sequence of accessing the serial peripheral status register while SPIF is set and followed by a write to or a read of the serial peripheral data register. While SPIF is set, all writes to the serial peripheral data register are inhibited until the serial peripheral status register is read. This occurs in the master device. In the slave device, SPIF can be cleared (using a similar sequence) during a second transmission; however, it must be cleared before the second SPIF in order to prevent an overrun condition. The SPIF bit is cleared by reset.

WCOL—Write Collision Status

The function of the write collision status bit is to notify the user that an attempt was made to write the serial peripheral data register while a data transfer was taking place with an external device. The transfer continues uninterrupted; therefore, a write will be unsuccessful. A "read collision" will never occur since the received data byte is placed in a buffer in which access is always synchronous with the MCU operation. If a "write collision" occurs, WCOL is set but no SPI interrupt is generated. The WCOL bit is a status flag only.

Clearing the WCOL bit is accomplished by a software sequence of accessing the serial peripheral status register while WCOL is set, followed by 1), a read of the serial peripheral data register prior to the SPIF bit being set,or 2) a read or write of the serial peripheral data register after the SPIF bit is set. A write to the serial peripheral data register (SPDR) prior to the SPIF bit being set, will result in generation of another WCOL status flag. Both the SPIF and WCOL bits will be cleared in the same sequence. If a second transfer has started while trying to clear (the previously set) SPIF and WCOL bits with a clearing sequence containing a write to the serial peripheral data register, only the SPIF bit will be cleared.

A collision of a write to the serial peripheral data register while an external data transfer is taking place can occur in both the master mode and the slave mode, although with proper programming the master device should have sufficient information to preclude this collision.

Collision in the master device is defined as a write of the serial peripheral data register while the internal rate clock (SCK) is in the process of transfer. The signal on the $\overline{SS}$ pin is always high on the master device.

A collision in a slave device is defined in two separate modes. One problem arises in a slave device when the CPHA control bit is a logic zero. When CPHA is a logic zero, data is latched with the occurrence of the first clock transition. The slave device does not have any way of knowing when that transition will occur; therefore, the slave device collision occurs when it attempts to write the serial peripheral data register after its $\overline{SS}$ pin has been pulled low. The $\overline{SS}$ pin of the slave device freezes the data in its serial peripheral data register and does not allow it to be altered if the CPHA bit is logic zero. The master device must raise the $\overline{SS}$ pin of the slave device high between each byte it transfers to the slave device.

The second collision mode is defined for the state of the CPHA control bit being a logic one. With the CPHA bit set, the slave device will be receiving a clock (SCK) edge prior to the latch of the first data transfer. This first clock edge will freeze the data in the slave device I/O register and allow the most significant bit onto the external MISO pin of the slave device. The $\overline{SS}$ pin low state enables the slave device but the drive onto the MSIO pin does not take place until the first data transfer clock edge. The WCOL bit will only be set if the I/O register is accessed while a transfer is taking place. By definition of the second collision mode, a master device might hold a slave device $\overline{SS}$ pin low during a transfer of several bytes of data without a problem.

In case of write collision occurs in the slave device. This happens when the master device starts a transfer sequence (an edge or SCK for CPHA = 1; or an active $\overline{SS}$ transition for CPHA = 0) at the same time the slave device CPU is writing to its serial peripheral interface data register. In this case it is assumed that the data byte written (in the slave device serial peripheral interface) is lost and the contents of the slave device read buffer becomes the byte that is transferred. Because the master device receives back the last byte transmitted, the master device can detect that a fatal write collision occurred.

Since the slave device is operating asynchronously with the master device, the WCOL bit may be used as an indicator of a collision occurrence. This helps alleviate the user from a strict real-time programming effort. The WCOL bit is cleared by reset.

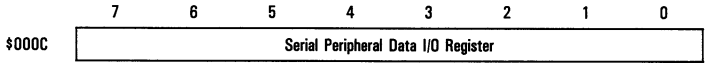
MODF—Mode Fault

The function of the mode fault flag bit is defined for the master mode (device). If the device is a slave device the MODF bit will be prevented from toggling from a logic zero to a logic one; however, this does not prevent the device from being in the slave mode with the MODF bit set. The MODF bit is normally a logic zero and is set only when the master device has its $\overline{SS}$ pin pulled low. Toggling the MODF bit to a logic one affects the internal serial peripheral interface (SPI) system in the following ways:

1. MODF is set and SPI interrupt is generated if SPIE = 1.
2. The SPE bit is forced to a logic zero. This blocks all output drive from the device, disables the SPI system.
3. The MSTR bit is forced to a logic zero, thus forcing the device into the slave mode.

Clearing the MODF is accomplished by a software sequence of accessing the serial peripheral status register while MODF is set followed by a write to the serial peripheral control register. Control bits SPE and MSTR may be restored to their original set state during this clearing sequence or after the MODF bit has been cleared. Hardware does not allow the user to set the SPE and MSTR bit while MODF is a logic one unless it is during the proper clearing sequence. The MODF flag bit indicates that there might have been a multi-master conflict for system control and allows a proper exit from system operation to a reset or default system state. The MODF bit is cleared by reset.

5.4.3 Serial Peripheral Data I/O Register (SPDR)



The serial peripheral data I/O register is used to transmit and receive data on the serial bus. Only a write to this register will initiate transmission/reception of another byte and this will only occur in the master device. A slave device writing to this data I/O register will not initiate a transmission. At the completion of transmitting a byte of data, the SPIF status bit is set in both the master and slave devices. A write or read of the serial peripheral data I/O register, after accessing the serial peripheral status register with SPIF set, will clear SPIF.

During the clock cycle that the SPIF bit is being set, a copy of the received data byte in the shift register is being moved to a buffer. When the user reads the serial peripheral data I/O register, the buffer is actually being read. During an overrun condition, when the master device has sent several bytes of data and the slave device has not internally responded to clear the first SPIF, only the first byte is contained in the receive buffer of the slave device; all others are lost. The user may read the buffer at any time. The first

be cleared by the time a second transfer of data from the shift register to the read buffer is initiated or an overrun condition will exist.

A write to the serial peripheral data I/O register is not buffered and places data directly into the shift register for transmission.

The ability to access the serial peripheral data I/O register is limited when a transmission is taking place. It is important to read the discussion defining the WCOL and SPIF status bits to understand the limits on using the serial peripheral data I/O register.

5.5 SPI DURING WAIT MODE

When the MCU enters the wait mode, the CPU clock is halted. All CPU action is suspended; however, the SPI system remains active. In fact an interrupt from the SPI (in addition to a logic low on the $\overline{IRQ}$ or $\overline{RESET}$ pins) causes the processor to exit the wait mode.

5.6 SPI DURING STOP MODE

When the processor executes the STOP instruction, the internal oscillator is turned off. This halts all internal CPU processing including the operation of the serial peripheral interface. The only way for the MCU to "wake up" from the stop mode is by receipt of an external interrupt (logic low on $\overline{IRQ}$ pin) or by the detection of a reset (logic low on $\overline{RESET}$ pin or a power-on reset).

When the MCU enters the stop mode, the baud rate generator which drives the SPI shuts down. This essentially stops all master mode SPI operation, thus the master SPI is unable to transmit or receive any data. If the STOP instruction is executed during an SPI transfer, that transfer is halted until the MCU exits the stop mode (provided it is an exit resulting from a logic low on the $\overline{IRQ}$ pin). If the stop mode is exited by a reset, then the appropriate control/status bits are cleared and the SPI is disabled. If the device is in the slave mode when the STOP instruction is executed, the slave SPI will still operate. It can still accept data and clock information in addition to transmitting its own data back to a master device.

Since the M6805 HCMOS master device is the bus master, it internally controls the function of its MOSI and MISO lines, thus writing data to the slave devices on the MOSI and reading data from the slave devices on the MISO lines. The master device selects the individual slave devices by using four pins of a parallel port to control the four $\overline{SS}$ pins of the slave devices. A slave device is selected when the master device pulls its $\overline{SS}$ pin low. The $\overline{SS}$ pins are pulled high during reset since the master device ports will be forced to be inputs at that time, thus disabling the slave devices. Note that the slave devices do not have to be enabled in a mutually exclusive fashion except to prevent bus contention on the MISO line.

For example, three slave devices, enabled for a transfer, are permissible if only one has the capability of being read by the master. An example of this is a write to several display drivers to clear a display with a single I/O operation. To ensure that proper data transmission is occurring between the master device and a slave device, the master device may have the slave device respond with a previously received data byte (this data byte could be inverted or at least be a byte that is different from the last one sent by the master device). The master device will always receive the previous byte back from the slave device if all MISO and MOSI lines are connected and the slave has not written its data I/O register. Other transmission security methods might be defined using ports for handshake lines or data bytes with command fields.

A multi-master system may also be configured by the user. A system of this type is shown in Figure 5-1(b). An exchange of master control could be implemented using a handshake method through the I/O ports or by an exchange of code messages through the serial peripheral interface system. The major device control that plays a part in this system is the MSTR bit in the serial peripheral control register and the MODF bit in the serial peripheral status register.

SECTION 6
LIQUID CRYSTAL DISPLAY DRIVER AND TONE GENERATOR

6.1 INTRODUCTION

This section contains a description of the liquid crystal display driver and the tone generator. Also included is a discussion of a common method for obtaining the LCD power supply voltage (V_{LL}), some short software segments to do some common LCD functions, some correlation information between a typical 7-segment display and the frontplane and backplane outputs for both x3 and x4 multiplexing, and some sample waveforms.

6.2 LIQUID CRYSTAL DISPLAY DRIVER

Figure 6-1 is a functional block diagram of the LCD driver circuitry. The processor internal data bus is connected to the LCD address register ($08), LCD address register ($09), and the configuration register.

The LCD address selects one of the 12 LCD data latches for the processor to read or write. The LCD data register will contain the data that is in the addressed latch. The data can either be read by the processor or the processor can change the data by writing new data into it.

The first write to the LCD address register is applied to the configuration register. This data must contain the following information: fast charge on or off, selection of x3 or x4 multiplexing, selection of which voltage generator to use, and selection of the crystal/LCD operating frequency ratio. After this first write, a latch is set and all writes to the LCD address register are accepted by only that register. All following writes to the LCD address register should contain only LCD latch address information.

The address information fed to the address register is a pure binary number between zero and eleven and the pointer or decode logic associated with the address register selects the associated LCD data latch. As soon as a latch is selected, the data in that latch is available at the LCD data register ($08) for use by the processor. The LCD data latches function like a dual-port RAM in that the data contained therein can be accessed from two different directions (in this case, the LCD data register and the scan logic).

While the processor is either reading or writing to the LCD data register, the data in the addressed latch is also being read or scanned by the scan logic circuitry. This circuitry is used to read the contents of all the LCD data latches and move that data to the LCD driver circuit where the proper frontplane (FPx) and backplane(BPx) outputs are fed the proper drive voltages to select the desired pixel.

Figure 6-2 is a map of the relationship of the individual backplane/frontplane pins to the individual LCD data latch bit positions. This is shown for x4 multiplexing. In x3 multiplexing, each BP4 position is not available.

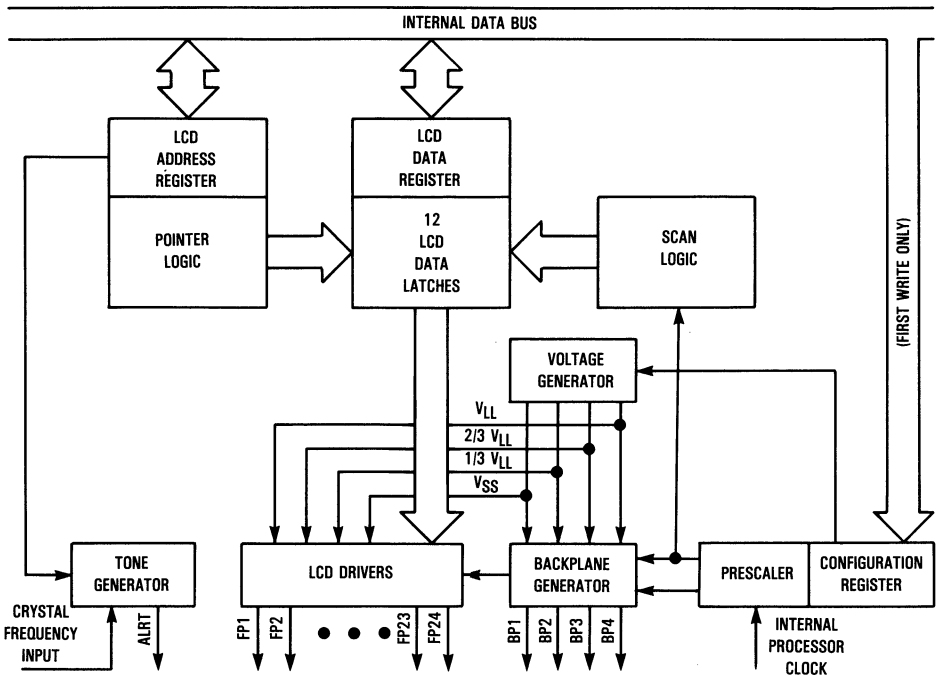


Figure 6-1. LCD Driver Circuitry—Functional Block Diagram

LCD DATA LATCH 00	7	6	5	4	3	2	1	0
($00)	BP4	BP3	BP2	BP1	BP4	BP3	BP2	BP1
	FP02				FP01			

LCD DATA LATCH 01	7	6	5	4	3	2	1	0
($01)	BP4	BP3	BP2	BP1	BP4	BP3	BP2	BP1
	FP04				FP03			

LCD DATA LATCH 02	7	6	5	4	3	2	1	0
($02)	BP4	BP3	BP2	BP1	BP4	BP3	BP2	BP1
	FP06				FP05			

LCD DATA LATCH 03	7	6	5	4	3	2	1	0
($03)	BP4	BP3	BP2	BP1	BP4	BP3	BP2	BP1
	FP08				FP07			

LCD DATA LATCH 04	7	6	5	4	3	2	1	0
($04)	BP4	BP3	BP2	BP1	BP4	BP3	BP2	BP1
	FP10				FP09			

LCD DATA LATCH 05	7	6	5	4	3	2	1	0
($05)	BP4	BP3	BP2	BP1	BP4	BP3	BP2	BP1
	FP12				FP11			

LCD DATA LATCH 06	7	6	5	4	3	2	1	0
($06)	BP4	BP3	BP2	BP1	BP4	BP3	BP2	BP1
	FP14				FP13			

LCD DATA LATCH 07	7	6	5	4	3	2	1	0
($07)	BP4	BP3	BP2	BP1	BP4	BP3	BP2	BP1
	FP16				FP15			

LCD DATA LATCH 08	7	6	5	4	3	2	1	0
($08)	BP4	BP3	BP2	BP1	BP4	BP3	BP2	BP1
	FP18				FP17			

LCD DATA LATCH 09	7	6	5	4	3	2	1	0
($09)	BP4	BP3	BP2	BP1	BP4	BP3	BP2	BP1
	FP20				FP19			

LCD DATA LATCH 10	7	6	5	4	3	2	1	0
($0A)	BP4	BP3	BP2	BP1	BP4	BP3	BP2	BP1
	FP22				FP21			

LCD DATA LATCH 11	7	6	5	4	3	2	1	0
($0B)	BP4	BP3	BP2	BP1	BP4	BP3	BP2	BP1
	FP24				FP23			

Figure 6-2. Backplane/Frontplane Pin to LCD Data Latch Bit Relationship

The voltage generator circuitry divides the liquid crystal display voltage (V_{LL}) into the 2/3, 1/3 voltage components required by multiplexed liquid crystal displays. These voltage levels are applied to the driver circuitry. Additional circuitry within the voltage generator provides the higher current needed when the fast charge option is selected.

The fast charge feature is used to quickly charge the capacitance of very large displays by using an extra high-current voltage generator for a very short period after each backplane or frontplane signal edge. This circuitry also contains additional supplies that are used for larger displays. The fast charge option is also available with the larger display voltage generator.

6.4 TONE GENERATOR

The tone generator is a counter which uses the crystal frequency input to produce an audio frequency square wave at the ALRT pin. (The ratio between the processor clock frequency and the tone frequency is determined by a user option at the time the part is manufactured.) The controls for the alert tone generator are located in the LCD address register. Figure 6-3 is a simplified diagram of the tone generator. The crystal frequency table shown in Figure 6-3 provides a list of the tone output frequencies provided by some of the more common crystal frequencies and the four divider options available.

OPTION	CRYSTAL FREQUENCY			
	8.389 MHz	4.194 MHz	2.097 MHz	0.524 MHz
A N = 1024	2048	1024	512	128
B N = 512	4096	2048	1024	512
C N = 256	8192	4096	2048	1024
D N = 64	32768	16384	8192	2048

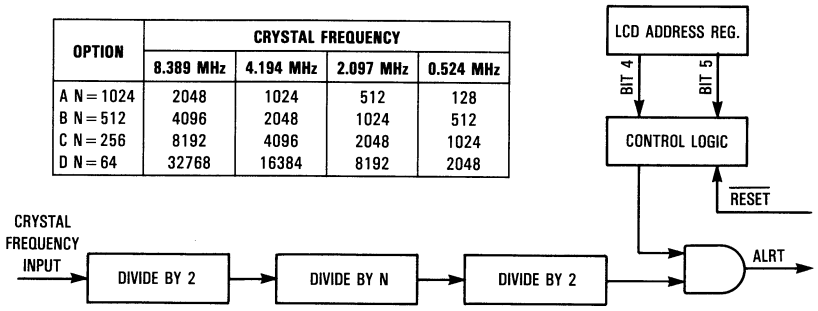


Figure 6-3. Tone Generator—Simplified Diagram

6.5 LCD ADDRESS REGISTER

The LCD address register is used to point to one of twelve data latches, numbered 0 through 11. The most significant nybble always reads as zeroes although it will perform some special functions as explained in the following paragraphs. This allows the use of all the read-modify-write instructions to quickly access any latch.

	7	6	5	4	3	2	1	0	
$0009	0	0	0	0	AR3	AR2	AR1	AR0	(READ)
	LON	LOF	TON	TOF	AR3	AR2	AR1	AR0	(WRITE)

The special functions contained in bits 7 through 4 of this register allow the user to directly control the LCD driver and tone generator circuitry.

If a one is written into bit 7, the liquid crystal display drivers are turned on. If a one is written into bit 6, they are turned off. The LCD drivers are disabled by a hardware reset ($\overline{RESET} = 0$).

, bit 5 will enable the tone generator; bit 4 will disable it. A hardware reset will also disable the tone generator.

Before the liquid crystal display driver circuitry can be used, the proper operating frequency ratio must be selected. This is accomplished by writing the appropriate value into the address register with the first write to this location after a reset.

6.6 LCD CONFIGURATION REGISTER

The first write to the LCD address register after a reset is used to configure the LCD driver system. Three parameters are selected by this write operation: the crystal oscillator to liquid crystal display operating frequency ratio, the voltage generator to be used, and multiplexing mode x3 or x4.

	7	6	5	4	3	2	1	0
$0009	EFC	x3	VGN	FR4	FR3	FR2	FR1	FR0

The EFC bit enables the fast charge feature if it is set.

The x3 bit selects the x3 multiplexing mode if it is set and the x4 multiplexing mode if it is clear.

The VGN bit selects the high current voltage generator if it is set and the low current voltage generator if it is clear.

The low order bits FR4 through FR0 are used to select the operating frequency ratio. Table 6-1 lists all the possibilities.

Table 6-1. Operating Frequency Ratio Chart

FR4 FR0	Xtal/LCD	Bus/LCD	FR4 FR0	Xtal/LCD	Bus/LCD
00000	512	256	10000	2048	1024
00001	1024	512	10001	4096	2048
00010	2048	1024	10010	8192	4096
00011	4096	2048	10011	16348	8192
00100	8192	4096	10100	32768	16348
00101	16384	8192	10101	65536	32768*
00110	32768	16384	10110	131072	65536
00111	65536	32768*	10111	262144	131072
01000	1536	768	11000	6656	3328
01001	3072	1536	11001	13312	6656
01010	6144	3072	11010	26624	13312
01011	12288	6144	11011	53248	26624
01100	24576	12288	11100	106496	53248
01101	49152	24576	11101	212992	106496
01110	98304	49152	11110	425984	212992
01111	196608	98304	11111	851968	425984

Example: *60 Hz LCD with a 4 MHz crystal.

6.7 LCD DATA REGISTER

The LCD data register is used to accept data from the processor that will go into the selected LCD data latch. This register will also contain the data present in any addressed LCD data latch and make it directly available for the processor to read. This register is initialized whenever a write to the LCD address register occurs.

6.8 LCD AND TONE GENERATOR DURING WAIT MODE

The liquid crystal display driver circuitry and tone generator are not affected by the WAIT instruction.

The liquid crystal display driver circuitry and tone generator are disabled by the STOP instruction.

6.10 COMMON METHOD FOR OBTAINING V_{LL}

The operating voltage for the liquid crystal display driver circuitry is the V_{LL} supply. It may range from 3.0 to 6.0 volts as needed. (Refer to the display manufacturer's recommendations for the required level.)

The MC68HC05L6 has internal level translators so that the liquid crystal display driver voltage (V_{LL}) is independent of the operating voltage (V_{DD}). Either voltage may be higher than the other.

A common method for determining the optimum drive level experimentally can be done with the configuration shown in Figure 6-4. V_{LL} can then be adjusted for the best contrast. The final design could simply replace the potentiometer with two fixed resistors if desired.

Most liquid crystal displays have a temperature coefficient of −8 mV/°C. A temperature compensated V_{LL} supply may be easily obtained using the circuit shown in Figure 6-5.

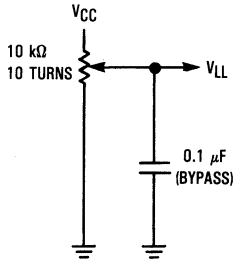


Figure 6-4. Test Circuit for Determining V_{LL} Drive Level

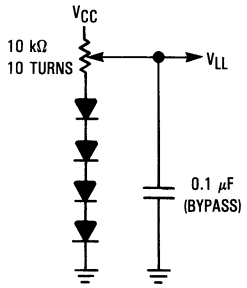


Figure 6-5. Test Circuit for Incorporating Temperature Compensation in the V_{LL} Supply

To determine which voltage generator to use and whether the fast charge feature is needed, connect the microcomputer and the liquid crystal display as it will be in the final product (using the final printed circuit board, crystal, etc. if possible) and use an oscilloscope (with a low capacitance probe) to examine the liquid crystal display driver waveforms. Choose the combination which yields the best waveforms (least noise).

In general most displays can be driven with the fast charge disabled. Fast charge is usually required for very large displays only.

6.11 LCD SOFTWARE EXAMPLES

The following software segments are provided as suggested examples of how to load the registers to perform a total display change, to scroll left , and to scroll right. NOTE: The parenthetical number (x) in the comment column of the listing represents the number of machine cycles that it takes to execute the given line of code. This is provided to assist you in determining the time required to execute the examples given.

```
            LDX     #11         (2) Initialize the pointer
LOOP        STX     ADDREG      (4) Point to the desired data register
            LDA     DATA,X      (4) Fetch the data
            STA     DATREG      (4) Store the data
            DEX                 (3) Update the data pointer
            BPL     LOOP        (3) Branch unless finished
11 Bytes, 218 Cycles
```

Scroll Left

```
            LDX     #1          (2) Initialize the pointer
LOOP1       STX     ADDREG      (4) Point to the desired data register
            LDA     DATREG      (3) Fetch the data
            DEC     ADDREG      (5) Point to the previous data register
            STA     DATREG      (3) Store the data
            INX                 (5) Update the data pointer
            CMPX    #12         (2) Are we finished?
            BNE     LOOP1       (3) Branch if no
15 Bytes, 302 Cycles
```

Scroll Right

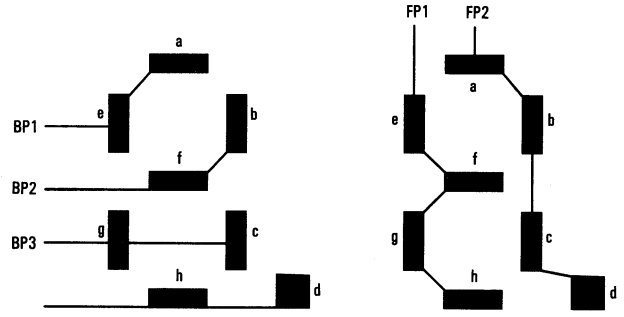
```
            LDX     #10         (2) Initialize the pointer
LOOP2       STX     ADDREG      (4) Point to the desired data register
            LDA     DATREG      (3) Fetch the data
            INC     ADDREG      (5) Point to the next data register
            STA     DATREG      (4) Store the data
            DEX                 (3) Update the data pointer
            BPL     LOOP2       (3) Branch unless finished
13 Bytes, 266 Cycles
```

6.12 7-SEGMENT DISPLAY CONNECTIONS

Figures 6-6 and 6-7 provide correlation between the frontplane and backplane outputs of the MC68HC05L6 and a typical 7-segment liquid crystal displays. Both the x3 and x4 mulitplexing modes are covered.

6.13 MULTIPLEXED SEGMENTED DISPLAY WAVEFORMS

Figure 6-8 (sheets 1 and 2) present examples of the waveforms generated by the driver circuitry of the MC68HC05L6 for application to multiplexed displays.



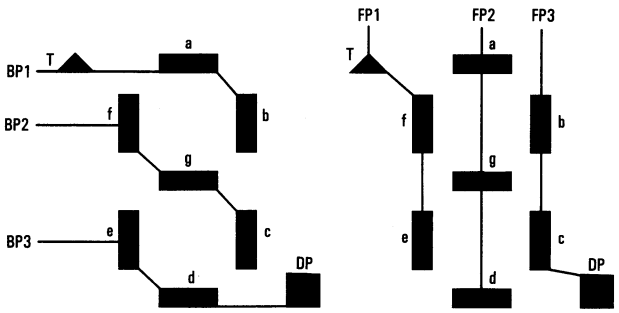
	FP1	FP2	FP3	FP4	.....	FP23	FP24
BP1	a1	e1	a2	e2	.....	a12	e12
BP2	b1	f1	b2	f2	.....	b12	f12
BP3	c1	g1	c2	g2	.....	c12	g12
BP4	d1	h1	d2	h2	.....	d12	h12
	DIGIT 1		DIGIT 2			DIGIT 12	

SEGMENT TRUTH TABLE*

	FP1	FP2
BP1	a1	e1
BP2	b1	f1
BP3	c1	g1
BP4	d1	h1

*Since there is no standard or backplane and frontplane connections on multiplexed displays, this truth table may be used for this example only.

Figure 6-6. Frontplane and Backplane Connections to a Multiplexed-by-Four 7-Segment LCD (Includes Decimal Point)



	FP1	FP2	FP3	FP4 ..... FP23	FP24
BP1	T1	a1	b2	T2 ..... a12	e12
BP2	f1	g1	c2	f2 ..... b12	f12
BP3	e1	d1	DP2	e2 ..... c12	g12
	DIGIT1			DIGIT 2 ..... DIGIT 8	

SEGMENT TRUTH TABLE*

	FP1	FP2	FP3
BP1	T1	a1	b2
BP2	f1	g1	c1
BP3	e1	d1	DP2

*Since there is no standard for backplane and frontplane connections on multiplexed displays, this truth table may be used for this example only.

Figure 6-7. Frontplane and Backplane Connections to a Multiplexed-by-Three 7-Segment LCD (Includes Decimal Point and Annunciator)

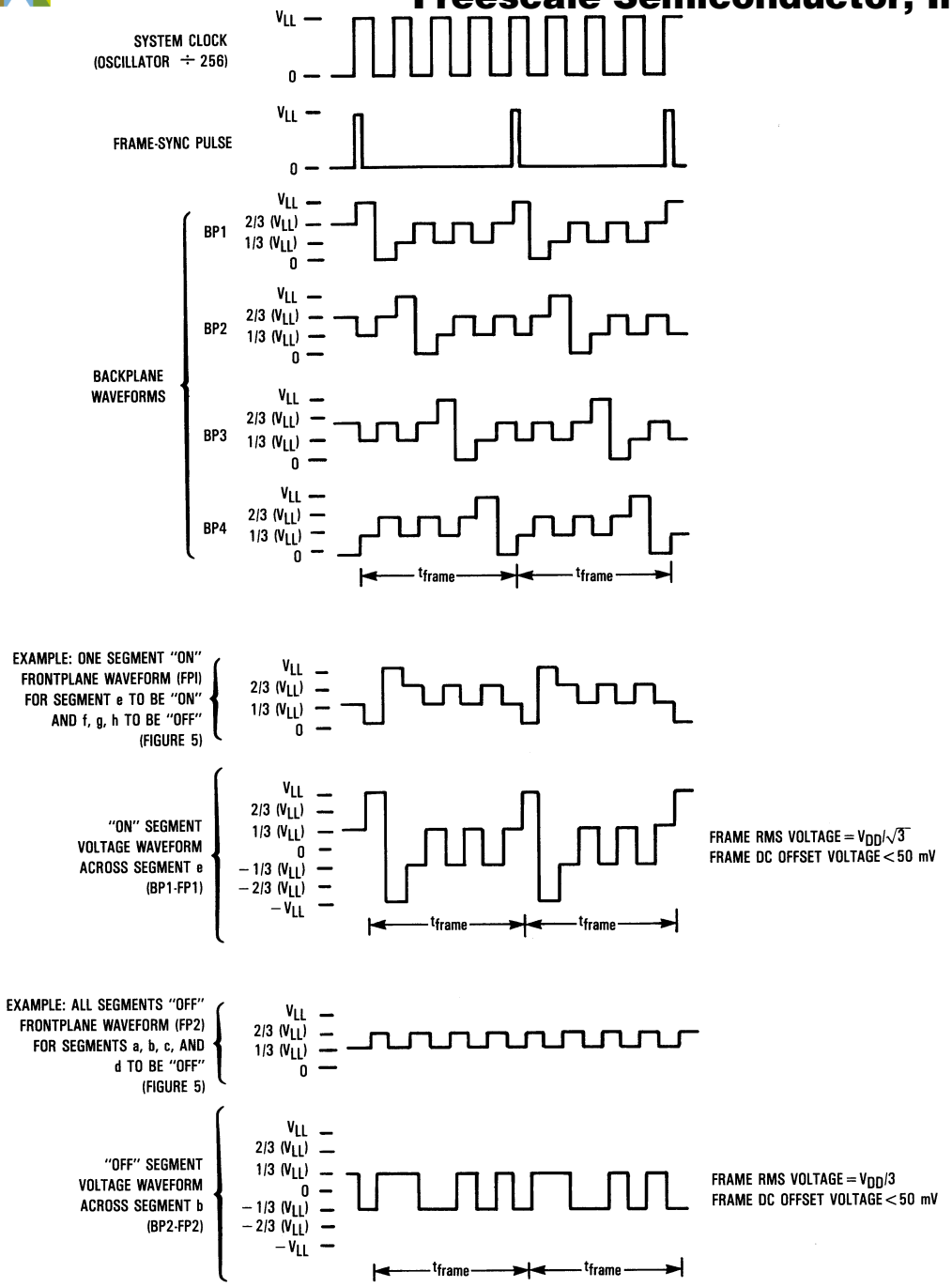


Figure 6-8. Multiplexed Segmented Display Waveforms (Sheet 1 of 2)

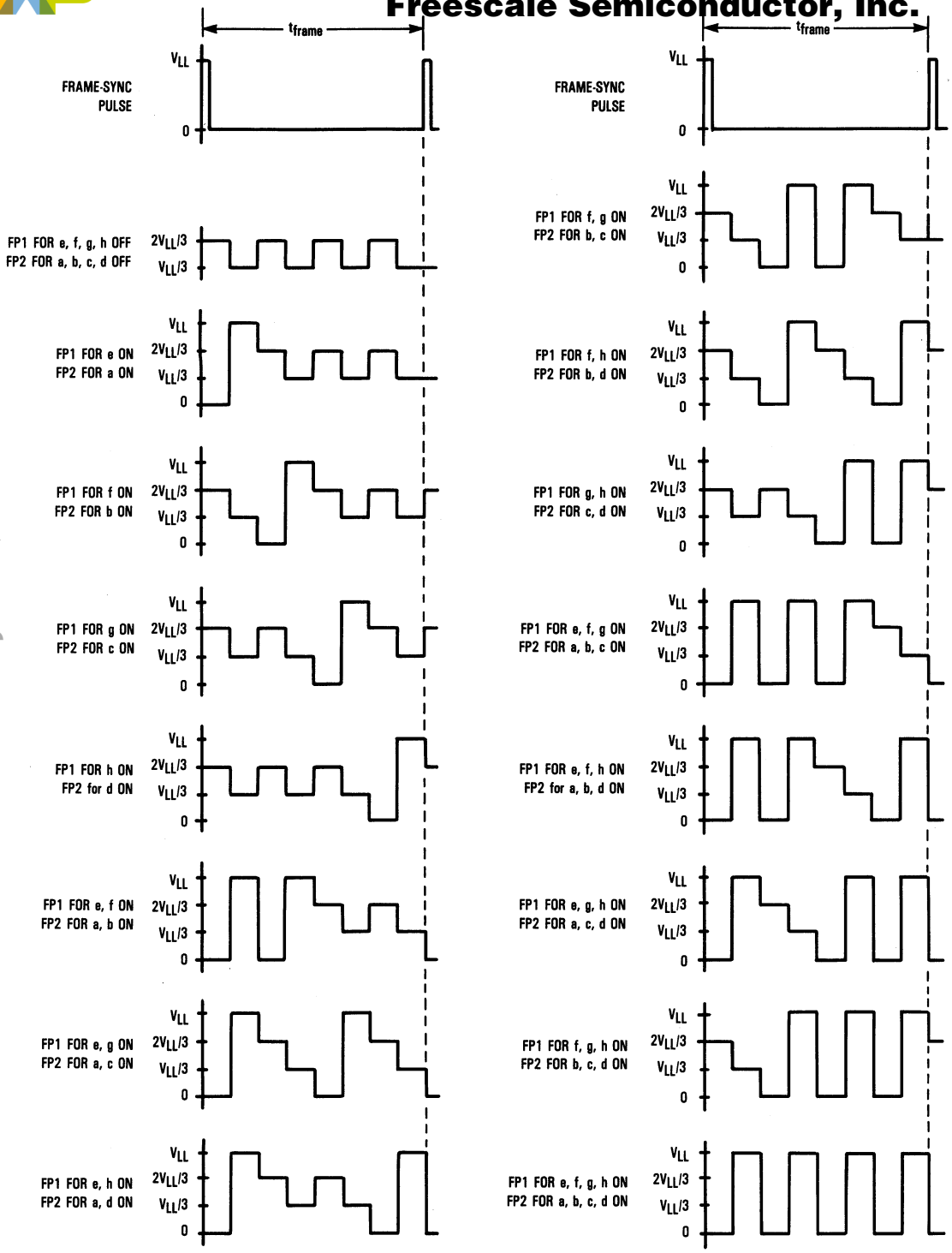


Figure 6-8. Multiplexed Segmented Display Waveforms (Sheet 2 of 2)

SECTION 7
INSTRUCTION SET AND ADDRESSING MODES

7.1 INTRODUCTION

This section contains a functional compilation of the instruction set for the MC68HC05L6 along with a brief description of the addressing modes available.

7.2 INSTRUCTION SET

All of the instructions used in the M146805 CMOS Family are used in the MC68HC05L6 MCU, plus an additional one, the multiply (MUL) instruction. This instruction allows for unsigned multiplication of the contents of the accumulator (A) and the index register (X). The higher order product is then stored in the index register and the low order product is stored in the accumulator. A detailed definition of the MUL instruction is shown below.

Operation: X:A ← X*A

Description: Multiplies the eight bits in the index register by the eight bits in the accumulator to obtain a 16-bit unsigned number in the concatenated accumulator and index register.

Condition Codes:
H—Cleared
I—Not Affected
N—Not Affected
Z—Not Affected
C—Cleared

Source Form(s): MUL

Addressing Mode	Cycles	Bytes	Opcode
Inherent	11	1	$42

The MC68HC05L6 has a set of 62 basic instructions. They can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type. All the instructions within a given type are presented in individual tables.

7.2.1 Register/Memory Instructions

Most of these instructions use two operands. The first operand is either the accumulator or the index register. The second operand is obtained from memory using one of the addressing modes. The operand for the jump unconditional (JMP) and jump to subroutine (JSR) instructions is the program counter. Refer to Table 7-1.

7.2.2 Read-Modify-Write Instructions

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to Table 7-2.

Table 7-1. Register/Memory Instructions

		Addressing Modes																		
		Immediate			Direct			Extended			Indexed (No Offset)			Indexed (8-Bit Offset)			Indexed (16-Bit Offset)		
Function	Mnem.	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Load A from Memory	LDA	A6	2	2	B6	2	3	C6	3	4	F6	1	3	E6	2	4	D6	3	5
Load X from Memory	LDX	AE	2	2	BE	2	3	CE	3	4	FE	1	3	EE	2	4	DE	3	5
Store A in Memory	STA	—	—	—	B7	2	4	C7	3	5	F7	1	4	E7	2	5	D7	3	6
Store X in Memory	STX	—	—	—	BF	2	4	CF	3	5	FF	1	4	EF	2	5	DF	3	6
Add Memory to A	ADD	AB	2	2	BB	2	3	CB	3	4	FB	1	3	EB	2	4	DB	3	5
Add Memory and Carry to A	ADC	A9	2	2	B9	2	3	C9	3	4	F9	1	3	E9	2	4	D9	3	5
Subtract Memory	SUB	A0	2	2	B0	2	3	C0	3	4	F0	1	3	E0	2	4	D0	3	5
Subtract Memory from A with Borrow	SBC	A2	2	2	B2	2	3	C2	3	4	F2	1	3	E2	2	4	D2	3	5
AND Memory to A	AND	A4	2	2	B4	2	3	C4	3	4	F4	1	3	E4	2	4	D4	3	5
OR Memory with A	ORA	AA	2	2	BA	2	3	CA	3	4	FA	1	3	EA	2	4	DA	3	5
Exclusive OR Memory with A	EOR	A8	2	2	B8	2	3	C8	3	4	F8	1	3	E8	2	4	D8	3	5
Arithmetic Compare A with Memory	CMP	A1	2	2	B1	2	3	C1	3	4	F1	1	3	E1	2	4	D1	3	5
Arithmetic Compare X with Memory	CPX	A3	2	2	B3	2	3	C3	3	4	F3	1	3	E3	2	4	D3	3	5
Bit Test Memory with A (Logical Compare)	BIT	A5	2	2	B5	2	3	C5	3	4	F5	1	3	E5	2	4	D5	3	5
Jump Unconditional	JMP	—	—	—	BC	2	2	CC	3	3	FC	1	2	EC	2	3	DC	3	4
Jump to Subroutine	JSR	—	—	—	BD	2	5	CD	3	6	FD	1	5	ED	2	6	DD	3	7

Table 7-2. Read-Modify-Write Instructions

		Addressing Modes														
		Inherent (A)			Inherent (X)			Direct			Indexed (No Offset)			Indexed (8-Bit Offset)		
Function	Mnemonic	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Increment	INC	4C	1	3	5C	1	3	3C	2	5	7C	1	5	6C	2	6
Decrement	DEC	4A	1	3	5A	1	3	3A	2	5	7A	1	5	6A	2	6
Clear	CLR	4F	1	3	5F	1	3	3F	2	5	7F	1	5	6F	2	6
Complement	COM	43	1	3	53	1	3	33	2	5	73	1	5	63	2	6
Negate (2's Complement)	NEG	40	1	3	50	1	3	30	2	5	70	1	5	60	2	6
Rotate Left Thru Carry	ROL	49	1	3	59	1	3	39	2	5	79	1	5	69	2	6
Rotate Right Thru Carry	ROR	46	1	3	56	1	3	36	2	5	76	1	5	66	2	6
Logical Shift Left	LSL	48	1	3	58	1	3	38	2	5	78	1	5	68	2	6
Logical Shift Right	LSR	44	1	3	54	1	3	34	2	5	74	1	5	64	2	6
Arithmetic Shift Right	ASR	47	1	3	57	1	3	37	2	5	77	1	5	67	2	6
Test for Negative or Zero	TST	4D	1	3	5D	1	3	3D	2	4	7D	1	4	6D	2	5
Multiply	MUL	42	1	11	—	—	—	—	—	—	—	—	—	—	—	—

Most branch instructions test the state of the condition code register and if certain criteria are met, a branch is executed. This adds an offset between −127 and +128 to the current program counter. Refer to Table 7-3.

Table 7-3. Branch Instructions

		Relative Addressing Mode		
Function	Mnemonic	Op Code	# Bytes	# Cycles
Branch Always	BRA	20	2	3
Branch Never	BRN	21	2	3
Branch IFF Higher	BHI	22	2	3
Branch IFF Lower or Same	BLS	23	2	3
Branch IFF Carry Clear	BCC	24	2	3
(Branch IFF Higher or Same)	(BHS)	24	2	3
Branch IFF Carry Set	BCS	25	2	3
(Branch IFF Lower)	(BLO)	25	2	3
Branch IFF Not Equal	BNE	26	2	3
Branch IFF Equal	BEQ	27	2	3
Branch IFF Half Carry Clear	BHCC	28	2	3
Branch IFF Half Carry Set	BHCS	29	2	3
Branch IFF Plus	BPL	2A	2	3
Branch IFF Minus	BMI	2B	2	3
Branch IFF Interrupt Mask Bit is Clear	BMC	2C	2	3
Branch IFF Interrupt Mask Bit is Set	BMS	2D	2	3
Branch IFF Interrupt Line is Low	BIL	2E	2	3
Branch IFF Interrupt Line is High	BIH	2F	2	3
Branch to Subroutine	BSR	AD	2	6

7.2.4 Bit Manipulation Instructions

The MCU is capable of setting or clearing any bit which resides in the first 256 bytes of the memory space except for ROM, port D data location ($03), serial peripheral status register ($0B), timer status register ($13), and timer input capture register ($14-$15). All port registers, port DDRs, timer, the serial system, on-chip RAM, and 48 bytes of user ROM reside in the first 256 bytes (page zero). An additional feature allows the software to test and branch on the state of any bit within the first 256 locations. The bit set, bit clear, and bit test and branch functions are all implemented with a single instruction. For the test and branch instructions, the value of the bit tested is automatically placed in the carry bit of the condition code register. Refer to Table 7-4.

Table 7-4. Bit Manipulation Instructions

		Addressing Modes					
		Bit Set/Clear			Bit Test and Branch		
Function	Mnemonic	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Branch IFF Bit n is Set	BRSET n (n=0...7)	—	—	—	2•n	3	5
Branch IFF Bit n is Clear	BRCLR n (n=0...7)	—	—	—	01+2•n	3	5
Set Bit n	BSET n (n=0...7)	10+2•n	2	5	—	—	—
Clear Bit n	BCLR n (n=0...7)	11+2•n	2	5	—	—	—

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to Table 7-5.

Table 7-5. Control Instructions

		Inherent		
Function	Mnemonic	Op Code	# Bytes	# Cycles
Transfer A to X	TAX	97	1	2
Transfer X to A	TXA	9F	1	2
Set Carry Bit	SEC	99	1	2
Clear Carry Bit	CLC	98	1	2
Set Interrupt Mask Bit	SEI	9B	1	2
Clear Interrupt Mask Bit	CLI	9A	1	2
Software Interrupt	SWI	83	1	10
Return from Subroutine	RTS	81	1	6
Return from Interrupt	RTI	80	1	9
Reset Stack Pointer	RSP	9C	1	2
No-Operation	NOP	9D	1	2
Stop	STOP	8E	1	2
Wait	WAIT	8F	1	2

7.3 INSTRUCTION SET—ALPHABETICAL LISTING

The complete instruction set is given in alphabetical order in Table 7-6.

7.4 INSTRUCTION SET—OPCODE MAP

Table 7-7 is an opcode map for the instructions.

7.5 ADDRESSING MODES

The MC68HC05L6 uses ten different addressing modes to provide the programmer with an opportunity to optimize the code to all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single byte instructions, while the longest instructions (three bytes) permit accessing tables throughout memory. Short absolute (direct) and long absolute (extended) addressing are also included. One and two byte direct addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory. Table 7-7 shows the addressing modes for each instruction, with the effects each instruction has on the condition code register.

The term "effective address" (EA) is used in describing the various addressing modes, and is defined as the byte address to or from which the argument for an instruction is fetched or stored. The ten addressing modes of the processor are described in the following paragraphs. Parentheses are used to indicate "contents of" the location or register referred to; e.g., (PC) indicates the contents of the location pointed to by the PC. An arrow indicates "is replaced by", and a colon indicates concatenation of two bytes.

7.5.1 Inherent

In inherent instructions, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator, and no other arguments, are included in this mode.

Table 7-6. Instruction Set—Alphabetical Listing

Mnemonic	Addressing Modes										Condition Codes				
	Inherent	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8 Bits)	Indexed (16 Bits)	Bit Set/ Clear	Bit Test & Branch	H	I	N	Z	C
ADC		X	X	X		X	X	X			Λ	•	Λ	Λ	Λ
ADD		X	X	X		X	X	X			Λ	•	Λ	Λ	Λ
AND		X	X	X		X	X	X			•	•	Λ	Λ	•
ASL	X		X			X	X				•	•	Λ	Λ	Λ
ASR	X		X			X	X				•	•	Λ	Λ	Λ
BCC					X						•	•	•	•	•
BCLR									X		•	•	•	•	•
BCS					X						•	•	•	•	•
BEQ					X						•	•	•	•	•
BHCC					X						•	•	•	•	•
BHCS					X						•	•	•	•	•
BHI					X						•	•	•	•	•
BHS					X						•	•	•	•	•
BIH					X						•	•	•	•	•
BIL					X						•	•	•	•	•
BIT		X	X	X		X	X	X			•	•	Λ	Λ	•
BLO					X						•	•	•	•	•
BLS					X						•	•	•	•	•
BMC					X						•	•	•	•	•
BMI					X						•	•	•	•	•
BMS					X						•	•	•	•	•
BNE					X						•	•	•	•	•
BPL					X						•	•	•	•	•
BRA					X						•	•	•	•	•
BRN					X						•	•	•	•	•
BRCLR										X	•	•	•	•	Λ
BRSET										X	•	•	•	•	Λ
BSET									X		•	•	•	•	•
BSR					X						•	•	•	•	•
CLC	X										•	•	•	•	0
CLI	X										•	0	•	•	•
CLR	X		X			X	X				•	•	0	1	•
CMP		X	X	X		X	X	X			•	•	Λ	Λ	Λ
COM	X		X			X	X				•	•	Λ	Λ	1
CPX		X	X	X		X	X	X			•	•	Λ	Λ	Λ
DEC	X		X			X	X				•	•	Λ	Λ	•
EOR		X	X	X		X	X	X			•	•	Λ	Λ	•
INC	X		X			X	X				•	•	Λ	Λ	•
JMP			X	X		X	X	X			•	•	•	•	•
JSR			X	X		X	X	X			•	•	•	•	•
LDA		X	X	X		X	X	X			•	•	Λ	Λ	•
LDX		X	X	X		X	X	X			•	•	Λ	Λ	•
LSL	X		X			X	X				•	•	Λ	Λ	Λ
LSR	X		X			X	X				•	•	0	Λ	Λ
MUL	X										0	•	•	•	0
NEG	X		X			X	X				•	•	Λ	Λ	Λ
NOP	X										•	•	•	•	•
ORA		X	X	X		X	X	X			•	•	Λ	Λ	•
ROL	X		X			X	X				•	•	Λ	Λ	Λ
ROR	X		X			X	X				•	•	Λ	Λ	Λ
RSP	X										•	•	•	•	•
RTI	X										?	?	?	?	?
RTS	X										•	•	•	•	•
SBC		X	X	X		X	X	X			•	•	Λ	Λ	Λ
SEC	X										•	•	•	•	1
SEI	X										•	1	•	•	•
STA			X	X		X	X	X			•	•	Λ	Λ	•
STOP	X										•	0	•	•	•
STX			X	X		X	X	X			•	•	Λ	Λ	•
SUB		X	X	X		X	X	X			•	•	Λ	Λ	Λ
SWI	X										•	1	•	•	•
TAX	X										•	•	•	•	•
TST	X		X			X	X				•	•	Λ	Λ	•
TXA	X										•	•	•	•	•
WAIT	X										•	0	•	•	•

Condition Code Symbols:

H	Half Carry (From Bit 3)	Λ	Test and Set if True Cleared Otherwise
I	Interrupt Mask	•	Not Affected
N	Negate (Sign Bit)	?	Load CC Register From Stack
Z	Zero	0	Cleared
C	Carry/Borrow	1	Set

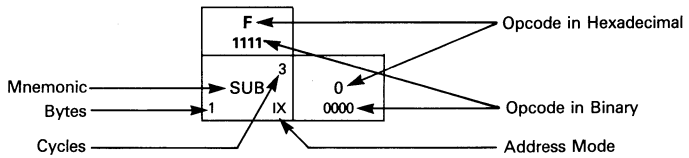
Table 7-1. Instruction Set Opcode Map

	Bit Manipulation		Branch	Read/Modify/Write					Control		Register/Memory						
	BTB	BSC	REL	DIR	INH	INH	IX1	IX	INH	INH	IMM	DIR	EXT	IX2	IX1	IX	
Hi / Low	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 / Low
0 0000	BRSET0 5 3 BTB	BSET0 5 2 BSC	BRA 3 2 REL	NEG 5 2 DIR	NEGA 3 1 INH	NEGX 3 1 INH	NEG 6 2 IX1	NEG 5 1 IX	RTI 9 1 INH		SUB 2 2 IMM	SUB 3 2 DIR	SUB 4 3 EXT	SUB 5 3 IX2	SUB 4 2 IX1	SUB 3 1 IX	0 0000
1 0001	BRCLR0 5 3 BTB	BCLR0 5 2 BSC	BRN 3 2 REL						RTS 6 1 INH		CMP 2 2 IMM	CMP 3 2 DIR	CMP 4 3 EXT	CMP 5 3 IX2	CMP 4 2 IX1	CMP 3 1 IX	1 0001
2 0010	BRSET1 5 3 BTB	BSET1 5 2 BSC	BHI 3 2 REL		MUL 11 1 INH						SBC 2 2 IMM	SBC 3 2 DIR	SBC 4 3 EXT	SBC 5 3 IX2	SBC 4 2 IX1	SBC 3 1 IX	2 0010
3 0011	BRCLR1 5 3 BTB	BCLR1 5 2 BSC	BLS 3 2 REL	COM 5 2 DIR	COMA 3 1 INH	COMX 3 1 INH	COM 6 2 IX1	COM 5 1 IX	SWI 10 1 INH		CPX 2 2 IMM	CPX 3 2 DIR	CPX 4 3 EXT	CPX 5 3 IX2	CPX 4 2 IX1	CPX 3 1 IX	3 0011
4 0100	BRSET2 5 3 BTB	BSET2 5 2 BSC	BCC 3 2 REL	LSR 5 2 DIR	LSRA 3 1 INH	LSRX 3 1 INH	LSR 6 2 IX1	LSR 5 1 IX			AND 2 2 IMM	AND 3 2 DIR	AND 4 3 EXT	AND 5 3 IX2	AND 4 2 IX1	AND 3 1 IX	4 0100
5 0101	BRCLR2 5 3 BTB	BCLR2 5 2 BSC	BCS 3 2 REL								BIT 2 2 IMM	BIT 3 2 DIR	BIT 4 3 EXT	BIT 5 3 IX2	BIT 4 2 IX1	BIT 3 1 IX	5 0101
6 0110	BRSET3 5 3 BTB	BSET3 5 2 BSC	BNE 3 2 REL	ROR 5 2 DIR	RORA 3 1 INH	RORX 3 1 INH	ROR 6 2 IX1	ROR 5 1 IX			LDA 2 2 IMM	LDA 3 2 DIR	LDA 4 3 EXT	LDA 5 3 IX2	LDA 4 2 IX1	LDA 3 1 IX	6 0110
7 0111	BRCLR3 5 3 BTB	BCLR3 5 2 BSC	BEQ 3 2 REL	ASR 5 2 DIR	ASRA 3 1 INH	ASRX 3 1 INH	ASR 6 2 IX1	ASR 5 1 IX		TAX 2 1 INH		STA 4 2 DIR	STA 5 3 EXT	STA 6 3 IX2	STA 5 2 IX1	STA 4 1 IX	7 0111
8 1000	BRSET4 5 3 BTB	BSET4 5 2 BSC	BHCC 3 2 REL	LSL 5 2 DIR	LSLA 3 1 INH	LSLX 3 1 INH	LSL 6 2 IX1	LSL 5 1 IX		CLC 2 1 INH	EOR 2 2 IMM	EOR 3 2 DIR	EOR 4 3 EXT	EOR 5 3 IX2	EOR 4 2 IX1	EOR 3 1 IX	8 1000
9 1001	BRCLR4 5 3 BTB	BCLR4 5 2 BSC	BHCS 3 2 REL	ROL 5 2 DIR	ROLA 3 1 INH	ROLX 3 1 INH	ROL 6 2 IX1	ROL 5 1 IX		SEC 2 1 INH	ADC 2 2 IMM	ADC 3 2 DIR	ADC 4 3 EXT	ADC 5 3 IX2	ADC 4 2 IX1	ADC 3 1 IX	9 1001
A 1010	BRSET5 5 3 BTB	BSET5 5 2 BSC	BPL 3 2 REL	DEC 5 2 DIR	DECA 3 1 INH	DECX 3 1 INH	DEC 6 2 IX1	DEC 5 1 IX		CLI 2 1 INH	ORA 2 2 IMM	ORA 3 2 DIR	ORA 4 3 EXT	ORA 5 3 IX2	ORA 4 2 IX1	ORA 3 1 IX	A 1010
B 1011	BRCLR5 5 3 BTB	BCLR5 5 2 BSC	BMI 3 2 REL							SEI 2 1 INH	ADD 2 2 IMM	ADD 3 2 DIR	ADD 4 3 EXT	ADD 5 3 IX2	ADD 4 2 IX1	ADD 3 1 IX	B 1011
C 1100	BRSET6 5 3 BTB	BSET6 5 2 BSC	BMC 3 2 REL	INC 5 2 DIR	INCA 3 1 INH	INCX 3 1 INH	INC 6 2 IX1	INC 5 1 IX		RSP 2 1 INH		JMP 2 2 DIR	JMP 3 3 EXT	JMP 4 3 IX2	JMP 3 2 IX1	JMP 2 1 IX	C 1100
D 1101	BRCLR6 5 3 BTB	BCLR6 5 2 BSC	BMS 3 2 REL	TST 4 2 DIR	TSTA 3 1 INH	TSTX 3 1 INH	TST 5 2 IX1	TST 4 1 IX		NOP 2 1 INH	BSR 6 2 REL	JSR 5 2 DIR	JSR 6 3 EXT	JSR 7 3 IX2	JSR 6 2 IX1	JSR 5 1 IX	D 1101
E 1110	BRSET7 5 3 BTB	BSET7 5 2 BSC	BIL 3 2 REL						STOP 2 1 INH		LDX 2 2 IMM	LDX 3 2 DIR	LDX 4 3 EXT	LDX 5 3 IX2	LDX 4 2 IX1	LDX 3 1 IX	E 1110
5 1111	BRCLR7 5 3 BTB	BCLR7 5 2 BSC	BIH 3 2 REL	CLR 5 2 DIR	CLRA 3 1 INH	CLRX 3 1 INH	CLR 6 2 IX1	CLR 5 1 IX	WAIT 2 1 INH	TXA 2 1 INH		STX 4 2 DIR	STX 5 3 EXT	STX 6 3 IX2	STX 5 2 IX1	STX 4 1 IX	F 1111

Abbreviations for Address Modes

INH	Inherent
A	Accumulator
X	Index Register
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

Legend



In immediate addressing, the operand is contained in the byte immediately following the opcode. Immediate addressing is used to access constants which do not change during program execution (e.g. a constant used to initialize a loop counter).

$$EA = PC + 1;\ PC \leftarrow PC + 2$$

7.5.3 Direct

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two-byte instruction. This includes all on-chip RAM and I/O registers, and 128 bytes of on-chip ROM. Direct addressing is efficient in both memory and time.

$$EA = (PC + 1);\ PC \leftarrow PC + 2$$
$$\text{Address Bus High} \leftarrow 0;\ \text{Address Bus Low} \leftarrow (PC + 1)$$

7.5.4 Extended

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode. Instructions with extended addressing modes are capable of referencing arguments anywhere in memory with a single three-byte instruction. When using the Motorola assembler, the user need not specify whether an instruction uses direct or extended addressing. The assembler automatically selects the most efficient addressing mode.

$$EA = (PC + 1):(PC + 2);\ PC \leftarrow PC + 3$$
$$\text{Address Bus High} \leftarrow (PC + 1);\ \text{Address Bus Low} \leftarrow (PC + 2)$$

7.5.5 Indexed, No Offset

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. Thus, this addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is used to move a pointer through a table or to address a frequently referenced RAM or I/O location.

$$EA = X;\ PC \leftarrow PC + 1$$
$$\text{Address Bus High} \leftarrow 0;\ \text{Address Bus Low} \leftarrow X$$

7.5.6 Indexed, 8-Bit Offset

Here the EA is obtained by adding the contents of the byte following the opcode to that of the index register; therefore, the operand is located anywhere within the lowest 511 memory locations. For example, this mode of addressing is useful for selecting the mth element in a n element table. All instructions are two bytes. The content of the index register (X) is not changed. The content of (PC + 1) is an unsigned 8-bit integer. One byte offset indexing permits look-up tables to be easily accessed in either RAM or ROM.

$$EA = X + (PC + 1);\ PC \leftarrow PC + 2$$
$$\text{Address Bus High} \leftarrow K;\ \text{Address Bus Low} \leftarrow X + (PC + 1)$$

where:

K = The carry from the addition of X + (PC + 2)

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This addressing mode can be used in a manner similar to indexed 8-bit offset, except that this three-byte instruction allows tables to be anywhere in memory (e.g., jump tables in ROM). As with direct and extended, the M6805 assembler determines the most efficient form of indexed offset; 8- or 16-bit. The content of the index register is not changed.

$$EA = X + [(PC + 1):(PC + 2)];\ PC \leftarrow PC + 3$$
$$\text{Address Bus High} \leftarrow (PC + 1) + K;\ \text{Address Bus Low} \leftarrow X + (PC + 2)$$

where:

K = The carry from the addition of $X + (PC + 2)$

7.5.8 Relative

Relative addressing is only used in branch instructions. In relative addressing, the content of the 8-bit signed byte following the opcode (the offset) is added to the PC if and only if the branch condition is true. Otherwise, control proceeds to the next instruction. The span of relative addressing is limited to the range of -126 to $+129$ bytes from the branch instruction opcode location. The Motorola assembler calculates the proper offset and checks to see if it is with in the span of the branch.

$$EA = PC + 2 + (PC + 1);\ PC \leftarrow EA \text{ if branch taken;}$$
$$\text{otherwise, } EA = PC \leftarrow PC + 2$$

7.5.9 Bit Set/Clear

Direct addressing and bit addressing are combined in instructions which set and clear individual memory and I/O bits. In the bit set and clear instructions, the byte is specified as a direct address in the location following the opcode. The first 256 addressable locations are thus accessed. The bit to be modified within that byte is specified in the first three bits of the opcode. The bit set and clear instructions occupy two bytes, one for the opcode (including the bit number) and the other to address the byte which contains the bit of interest.

$$EA = (PC + 1);\ PC \leftarrow PC + 2$$
$$\text{Address Bus High} \leftarrow 0;\ \text{Address Bus Low} \leftarrow (PC + 1)$$

7.5.10 Bit Test and Branch

Bit test and branch is a combination of direct addressing, bit set/clear addressing, and relative addressing. The actual bit to be tested, within the byte, is specified within the low order nibble of the opcode. The address of the data byte to be tested is located via a direct address in the location following the opcode byte (EA1). The signed relative 8-bit offset is in the third byte (EA2) and is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any bit in the first 256 locations of memory.

$$EA1 = (PC + 1)$$
$$\text{Address Bus High} \leftarrow 0;\ \text{Address Bus Low} \leftarrow (PC + 1)$$
$$EA2 = PC + 3 + (PC + 2);\ PC \leftarrow EA2 \text{ if branch taken;}$$
$$\text{otherwise, } PC \leftarrow PC + 3$$

SECTION 8
ELECTRICAL SPECIFICATIONS

8.1 INTRODUCTION

This section contains the electrical specifications and associated timing information for the MC68HC05L6.

8.2 MAXIMUM RATINGS

(Voltages referenced to V_{SS})

Rating	Symbol	Value	Unit
Supply Voltage	V_{DD}	−0.3 to +7.0	V
Input Voltage	V_{in}	V_{SS}−0.3 to V_{DD}+0.3	V
Self-Check Mode ($\overline{IRQ}$ Pin Only)	V_{in}	V_{SS}−0.3 to $2\times V_{DD}$+0.3	V
Current Drain Per Pin Excluding V_{DD} and V_{SS}	I	25	mA
Operating Temperature Range MC68HC05L6	T_A	T_L to T_H 0 to +70	°C
Storage Temperature Range	T_{stg}	−65 to +150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \le (V_{in}$ or $V_{out}) \le V_{DD}$. Reliablity of operation is enhanced if unused inputs are connected to an appropriate logic voltage level (e.g., either V_{SS} or V_{DD}).

8.3 THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic Leaded Chip Carrier	θ_{JA}	70	°C/W

V_{DD} = 4.5 V

PINS	R1	R2	C1
PA0–PA7 PB0–PB7 PC0–PC7	3.26 kΩ	2.38 kΩ	50 pF

V_{DD} = 3.0 V

PINS	R1	R2	C1
PA0–PA7 PB0–PB7 PC0–PC7	10.91 kΩ	6.32 kΩ	50 pF

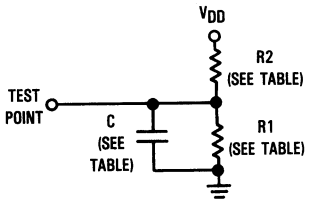


Figure 8-1. Equivalent Test Load

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

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

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
$P_D = P_{INT} + P_{I/O}$
$P_{INT} = I_{CC} \times V_{CC}$, Watts — Chip Internal Power
$P_{I/O}$ = Power Dissipation on Input and Output Pins, Watts — User Determined

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

An approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected) is:

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

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

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

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

8.5 DC ELECTRICAL CHARACTERISTICS

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

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage ($I_{load} \le 10.0\ \mu A$)	V_{OL}	—	—	0.1	V
	V_{OH}	$V_{DD}-0.1$	—	—	V
Output High Voltage (See Figure 8-2) (I_{load} = 0.8 mA) PA0-PA7, PB0-PB7, PC0-PC7, TCMP	V_{OH}	$V_{DD}-0.8$	—	—	V
Output Low Voltage (See Figure 8-3) (I_{load} = 1.6 mA) PA0-PA7, PB0-PB7, PC0-PC7, TCMP	V_{OL}	—	—	0.4	V
Input High Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD3, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IH}	$0.7 \times V_{DD}$	—	V_{DD}	V
Input Low Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD3, TCAP, $\overline{IRQ}$, $\overline{RESET}$, 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 (See Notes)	I_{DD}				mA
Run (See Figures 8-4 and 8-5)		—	3.5	7.0	
Wait (See Figures 8-4 and 8-5)		—	1.6	4.0	
Stop (See Figure 8-5)					μA
25°C		—	2.0	50	
0° to 70°C (Standard)		—	—	140	
LCD (V_{ref} large)	I_{LL}	—	15	TBD	μA
(V_{ref} small)		—	5	TBD	
I/O Port Hi-Z Leakage Current PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD3	I_{IL}	—	—	±10	μA
Input Current $\overline{RESET}$, $\overline{IRQ}$, TCAP, OSC1, PD3	I_{in}	—	—	±1	μA
Capacitance Ports (as Input or Output)	C_{out}	—	—	12	pF
$\overline{RESET}$, $\overline{IRQ}$, TCAP, OSC1, PD3	C_{in}	—	—	8	

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 active, add 10%.
4. Run (Operating) I_{DD}, 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 = 200 pF on OSC2.
5. Wait, 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° to 70°C.
8. Wait I_{DD} is affected linearly by the OSC2 capacitance.

RICAL CHARACTERISTICS

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

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage ($I_{load} \leq 10.0\ \mu A$)	V_{OL}	–	–	0.1	V
	V_{OH}	$V_{DD} - 0.1$	–	–	V
Output High Voltage (See Figure 8-2) (I_{load} = 0.8 mA) PA0–PA7, PB0–PB7, PC0–PC7, TCMP	V_{OH}	$V_{DD} - 0.8$	–	–	V
Output Low Voltage (See Figure 8-3) (I_{load} = 1.6 mA) PA0–PA7, PB0–PB7, PC0–PC7, TCMP	V_{OL}	–	–	0.3	V
Input High Voltage PA0–PA7, PB0–PB7, PC0–PC7, PD0–PD3, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IH}	$0.7 \times V_{DD}$	–	V_{DD}	V
Input Low Voltage PA0–PA7, PB0–PB7, PC0–PC7, PD0–PD3, TCAP, $\overline{IRQ}$, $\overline{RESET}$, 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 (See Notes)	I_{DD}				mA
Run (See Figures 8-4 and 8-6)		–	1.0	2.5	
Wait (See Figures 8-4 and 8-6)		–	0.5	1.4	
Stop (See Figure 8-6)					μA
25°C		–	1.0	30	
0° to 70°C (Standard)		–	–	80	
LCD (V_{ref} large)	I_{LL}	–	15	TBD	μA
(V_{ref} small)		–	5	TBD	
I/O Port Hi-Z Leakage Current PA0–PA7, PB0–PB7, PC0–PC7, PD0–PD3	I_{IL}	–	–	±10	μA
Input Current $\overline{RESET}$, $\overline{IRQ}$, TCAP, OSC1, PD3	I_{in}	–	–	±1	μA
Capacitance Ports (as Input or Output)	C_{out}	–	–	12	pF
$\overline{RESET}$, $\overline{IRQ}$, TCAP, OSC1, PD3	C_{in}	–	–	8	

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 active, add 10%.
4. Run (Operating) I_{DD}, 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 = 200 pF on OSC2.
5. Wait, 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° to 70°C.
8. Wait I_{DD} is affected linearly by the OSC2 capacitance.

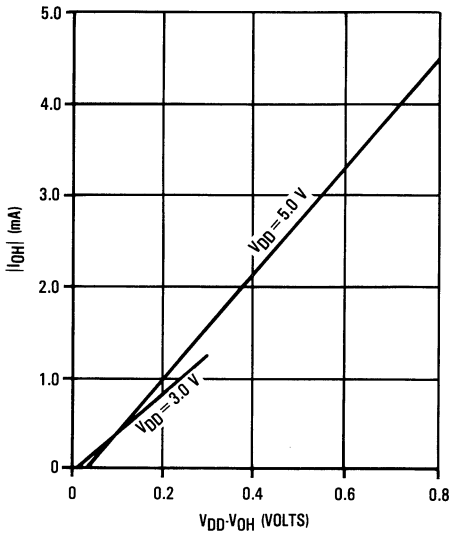


Figure 8-2. Typical V_{OH} vs I_{OH} for Ports A, B, C, and TCMP

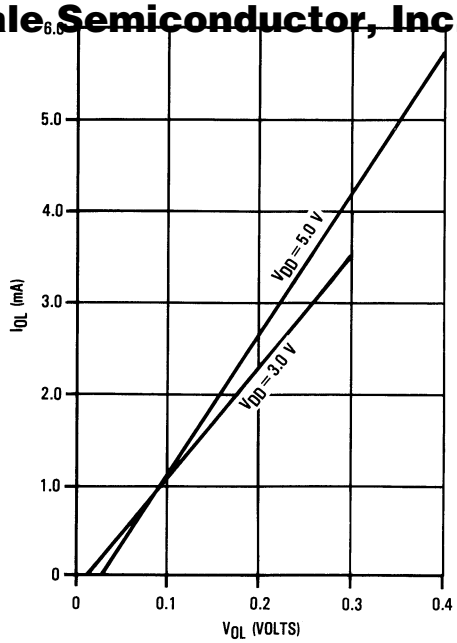
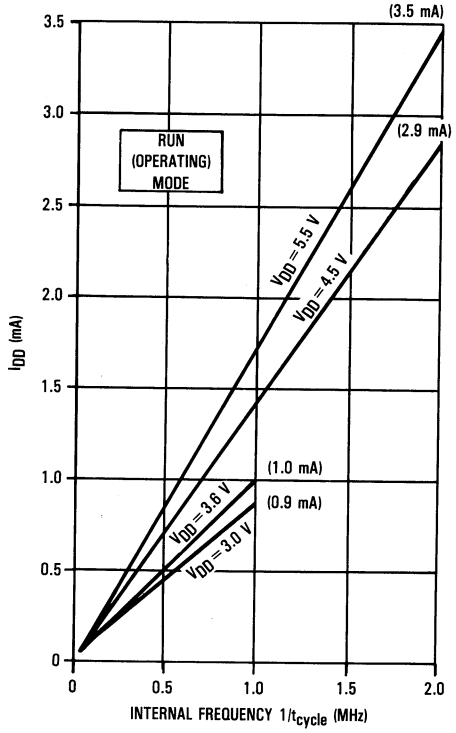


Figure 8-3. Typical V_{OL} vs I_{OL} for All Ports



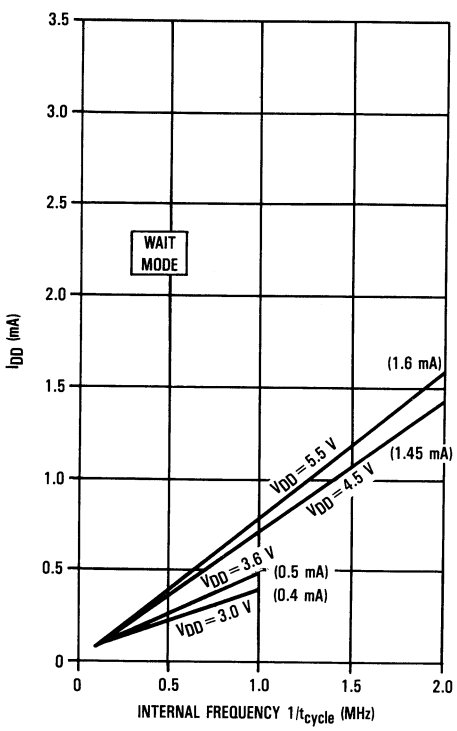


Figure 8-4. Typical Current vs Internal Frequency for Run and Wait Modes

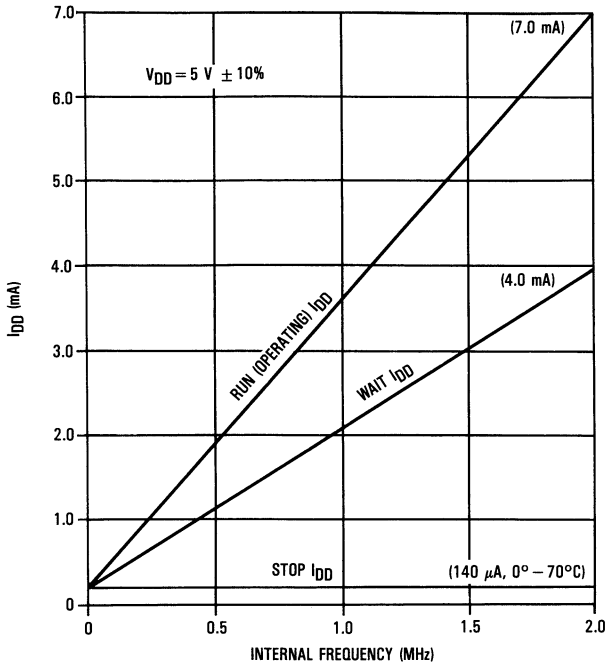


Figure 8-5. Maximum I_{DD} vs Frequency for V_{DD} = 5.0 Vdc

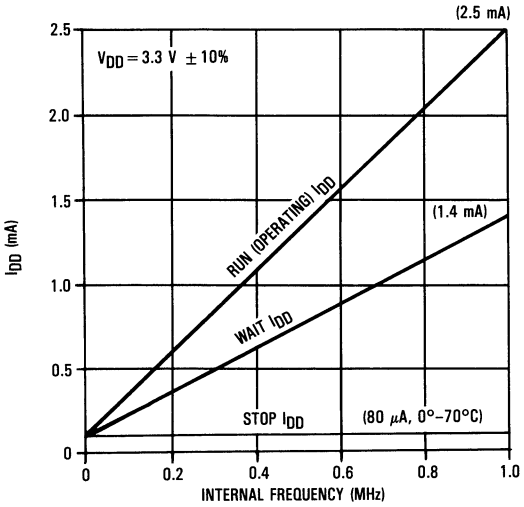


Figure 8-6. Maximum I_{DD} vs Frequency for V_{DD} = 3.3 Vdc

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

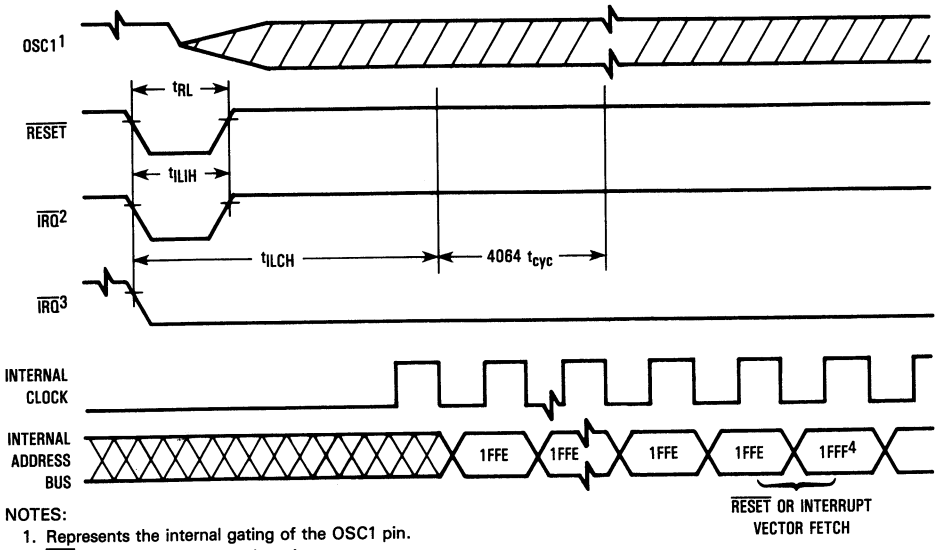
Characteristic	Symbol	Min	Max	Unit
Frequency of Operation Crystal Operation External Clock Option	f_{osc}	 – dc	 4.2 4.2	MHz
Internal Operating Frequency Crystal (f_{osc}/2) External Clock (f_{osc}/2)	f_{op}	 – dc	 2.1 2.1	MHz
Cycle Time	t_{cyc}	480	–	ns
Crystal Oscillator Startup Time	t_{OXOV}	–	100	ms
Stop Recovery Startup Time (Crystal Oscillator) (See Figure 8-7)	t_{ILCH}	–	100	ms
$\overline{\text{RESET}}$ Pulse Width	t_{RL}	1.5	–	t_{cyc}
Timer Resolution** Input Capture Pulse Width (See Figure 8-8) Input Capture Pulse Period (See Figure 8-8)	 t_{RES1} t_{TH}, t_{TL} t_{TLTL}	 4.0 125 ***	 – – –	 t_{cyc} ns t_{cyc}
Input Pulse Width Low (Edge-Triggered)	t_{ILIH}	125	–	ns
Interrupt Pulse Period	t_{ILIL}	*	–	t_{cyc}
OSC1 Pulse Width	t_{OH}, t_{OL}	90	–	ns

*The minimum period t_{ILIL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus 21 t_{cyc}.
**Since a two-bit prescaler in the timer must count four internal cycles, this is the limiting minimum factor in determining the timer resolution.
***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}.

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

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation Crystal Operation 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
Cycle Time	t_{cyc}	1000	–	ns
Crystal Oscillator Startup Time	t_{OXOV}	–	100	ms
Stop Recovery Startup Time (Crystal Oscillator) (See Figure 8-7)	t_{ILCH}	–	100	ms
$\overline{\text{RESET}}$ Pulse Width	t_{RL}	1.5	–	t_{cyc}
Timer Resolution** Input Capture Pulse Width (See Figure 8-8) Input Capture Pulse Period (See Figure 8-8)	 t_{RES1} t_{TH}, t_{TL} t_{TLTL}	 4.0 250 ***	 – – –	 t_{cyc} ns t_{cyc}
Input Pulse Width Low (Edge-Triggered)	t_{ILIH}	250	–	ns
Interrupt Pulse Period	t_{ILIL}	*	–	t_{cyc}
OSC1 Pulse Width	t_{OH}, t_{OL}	200	–	ns

*The minimum period t_{ILIL} should not be less than the number of cycle times it takes to execute the interrupt service routine plus 21 t_{cyc}.
**Since a two-bit prescaler in the timer must count four internal cycles, this is the limiting minimum factor in determining the timer resolution.
***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}.



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

Figure 8-7. Stop Recovery Timing Diagram

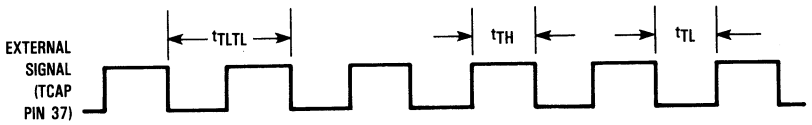


Figure 8-8. Timer Relationships

AL PERIPHERAL INTERFACE (SPI) TIMING (Figure 8-9)

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

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
3	Enable Lag Time Master Slave	 $t_{lag(m)}$ $t_{lag(s)}$	 * 240	 — —	ns
4	Clock (SCK) High Time Master Slave	 $t_{w(SCKH)m}$ $t_{w(SCKH)s}$	 340 190	 — —	ns
5	Clock (SCK) Low Time Master Slave	 $t_{w(SCKL)m}$ $t_{w(SCKL)s}$	 340 190	 — —	ns
6	Data Setup Time (Inputs) Master Slave	 $t_{su(m)}$ $t_{su(s)}$	 100 100	 — —	ns
7	Data Hold Time (Inputs) Master Slave	 $t_{h(m)}$ $t_{h(s)}$	 100 100	 — —	ns
8	Access Time (Time to Data Active from High Impedance State) Slave	t_a	0	120	ns
9	Disable Time (Hold Time to High Impedance State) Slave	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, MISO) Slave (SCK, MOSI, MISO, $\overline{SS}$)	 $t_{r(m)}$ $t_{r(s)}$	 — —	 100 2.0	ns
13	Fall Time (20% V_{DD} to 70% V_{DD}, C_L = 200 pF SPI Outputs (SCK, MOSI, MISO) Slave (SCK, MOSI, MISO, $\overline{SS}$)	 $t_{f(m)}$ $t_{f(s)}$	 — —	 100 2.0	ns

*Signal production depends on software.
**Assumes 200 pF load on all SPI pins.

SPI Timing diagrams (Figure 8-9) are located on foldout pages at the end of this document.

SERIAL PERIPHERAL INTERFACE (SPI) TIMING (Figure 8-9)

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

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
3	Enable Lag Time Master Slave	$t_{lag(m)}$ $t_{lag(s)}$	* 500	— —	ns
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
7	Data Hold Time (Inputs) Master Slave	$t_{h(m)}$ $t_{h(s)}$	200 200	— —	ns
8	Access Time (Time to Data Active from High Impedance State) Slave	t_a	0	250	ns
9	Disable Time (Hold Time to High Impedance State) Slave	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, MISO) Slave (SCK, MOSI, MISO, $\overline{SS}$)	$t_{r(m)}$ $t_{r(s)}$	— —	200 2.0	ns
13	Fall Time (20% V_{DD} to 70% V_{DD}, C_L = 200 pF SPI Outputs (SCK, MOSI, MISO) Slave (SCK, MOSI, MISO, $\overline{SS}$)	$t_{f(m)}$ $t_{f(s)}$	— —	200 2.0	ns

*Signal production depends on software.

**Assumes 200 pF load on all SPI pins.

SPI Timing diagrams (Figure 8-9) are located on foldout pages at the end of this document.

) DRIVER DC ELECTRICAL CHARACTERISTICS ($V_{GN}=0$)

Characteristics		Symbol	V_{LL} V	−55°C Min	−55°C Max	25°C Min	25°C Typ	25°C Max	125°C Min	125°C Max	Unit
RMS Voltage Across a Segment (BPi-FPj)	"ON" Segment	V_{ON}	3.0	–	–	–	1.73	–	–	–	V
			5.0	–	–	–	2.88	–	–	–	
	"OFF" Segment	V_{OFF}	3.0	–	–	–	1.00	–	–	–	V
			5.0	–	–	–	1.67	–	–	–	
Average dc Offset Voltage		Vdc	3.0	–	30	–	10	30	–	30	mV
			5.0	–	50	–	17	50	–	50	
Output Current, Backplanes and Frontplanes High-Current State*	$V_O=2.85$ V	I_{BH}	3.0	–	–	−100	−240	–	–	–	µA
	$V_O=1.85$ V			–	–	−15	−35	–	–	–	
	$V_O=1.15$ V			–	–	15	35	–	–	–	
	$V_O=0.15$ V			–	–	100	240	–	–	–	
	$V_O=4.85$ V		5.0	–	–	−200	−400	–	–	–	
	$V_O=3.18$ V			–	–	−50	−110	–	–	–	
	$V_O=1.82$ V			–	–	50	110	–	–	–	
	$V_O=0.15$ V			–	–	200	400	–	–	–	
Low-Current State*	$V_O=2.85$ V	I_{BL}	3.0	–	–	−100	−240	–	–	–	
	$V_O=1.85$ V			–	–	−0.25	−1.0	–	–	–	
	$V_O=1.15$ V			–	–	0.25	1.0	–	–	–	
	$V_O=0.15$ V			–	–	100	240	–	–	–	
	$V_O=4.85$ V		5.0	–	–	−200	−400	–	–	–	
	$V_O=3.18$ V			–	–	−1.0	−3.0	–	–	–	
	$V_O=1.82$ V			–	–	1.0	3.0	–	–	–	
	$V_O=0.15$ V			–	–	200	400	–	–	–	

*For time $1/(256\times f_{LCD})$ after the backplane or frontplane waveform changes to a new voltage level, the circuit is maintained in the high-current state to allow the load capacitors to charge quickly. Then the circuit is returned to the low-current state until the next voltage level change occurs.

8.12 LCD DRIVER DC ELECTRICAL CHARACTERISTICS ($V_{GN}=1$)

Characteristics		Symbol	V_{LL} V	−55°C Min	−55°C Max	25°C Min	25°C Typ	25°C Max	125°C Min	125°C Max	Unit
RMS Voltage Across a Segment (BPi-FPj)	"ON" Segment	V_{ON}	3.0	–	–	–	1.73	–	–	–	V
			5.0	–	–	–	2.88	–	–	–	
	"OFF" Segment	V_{OFF}	3.0	–	–	–	1.00	–	–	–	V
			5.0	–	–	–	1.67	–	–	–	
Average dc Offset Voltage		Vdc	3.0	–	30	–	10	30	–	30	mV
			5.0	–	50	–	17	50	–	50	
Output Current, Backplanes and Frontplanes High-Current State*	$V_O=2.85$ V	I_{BH}	3.0	–	–	−100	−240	–	–	–	µA
	$V_O=1.85$ V			–	–	−24	−75	–	–	–	
	$V_O=1.15$ V			–	–	24	75	–	–	–	
	$V_O=0.15$ V			–	–	100	240	–	–	–	
	$V_O=4.85$ V		5.0	–	–	−200	−400	–	–	–	
	$V_O=3.18$ V			–	–	−80	−250	–	–	–	
	$V_O=1.82$ V			–	–	80	250	–	–	–	
	$V_O=0.15$ V			–	–	200	400	–	–	–	
Low-Current State*	$V_O=2.85$ V	I_{BL}	3.0	–	–	−100	−240	–	–	–	
	$V_O=1.85$ V			–	–	−0.5	−2.0	–	–	–	
	$V_O=1.15$ V			–	–	0.5	2.0	–	–	–	
	$V_O=0.15$ V			–	–	100	240	–	–	–	
	$V_O=4.85$ V		5.0	–	–	−200	−400	–	–	–	
	$V_O=3.18$ V			–	–	−2.0	−7.0	–	–	–	
	$V_O=1.82$ V			–	–	2.0	7.0	–	–	–	
	$V_O=0.15$ V			–	–	200	400	–	–	–	

*For time $1/(256\times f_{LCD})$ after the backplane or frontplane waveform changes to a new voltage level, the circuit is maintained in the high-current state to allow the load capacitors to charge quickly. Then the circuit is returned to the low-current state until the next voltage level change occurs.

SECTION 9
ORDERING INFORMATION

Please contact your local sales representative for acceptable media to use for pattern transmittal and details on MCU ROM-based product flow.

Use a copy of this sheet to note the options that you require in the manufacturing of your microcomputer. Send this completed form to your Motorola Representative. They will ensure that it is forwarded to the factory in a timely manner.

Date ____/____/____

Customer _____ Customer PO _____

Address _____

City _____ State _____ Zip _____

Customer Contact _____ Phone _____

MC68HC05L6 Option Sheet

Internal Oscillator Input
- ☐ Crystal/Ceramic Resonator
- ☐ Resistor/Capacitor

Tone Generator (See Figure 6-3)
- A Crystal Frequency/(1024•2)
- B Crystal Frequency/(512•2)
- C Crystal Frequency/(256•2)
- D Crystal Frequency/(64•2)

Input Trigger
- ☐ Edge Sensitive
- ☐ Level and Edge Sensitive

Clock Frequency _____
Temperature Range _____

Marking Information (12 Characters Maximum)

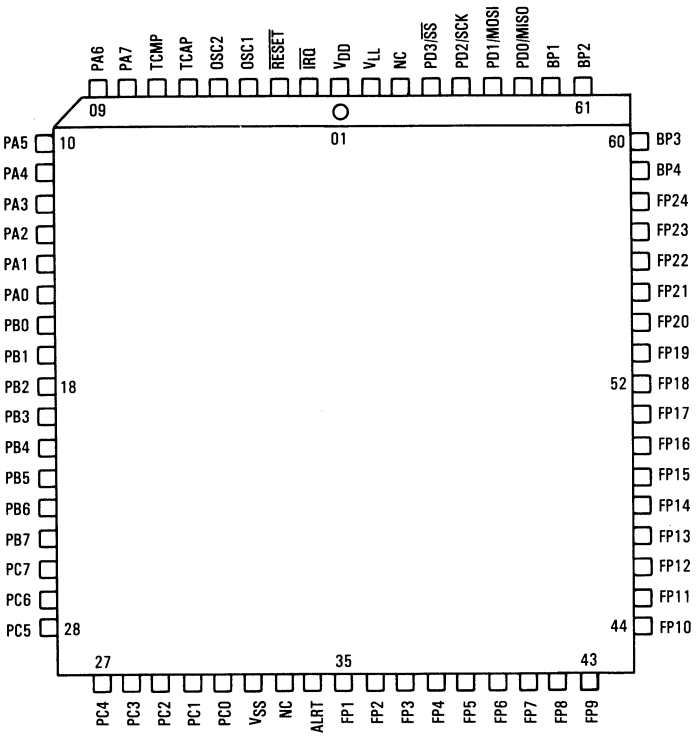
___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___

SECTION 10
MECHANICAL DATA

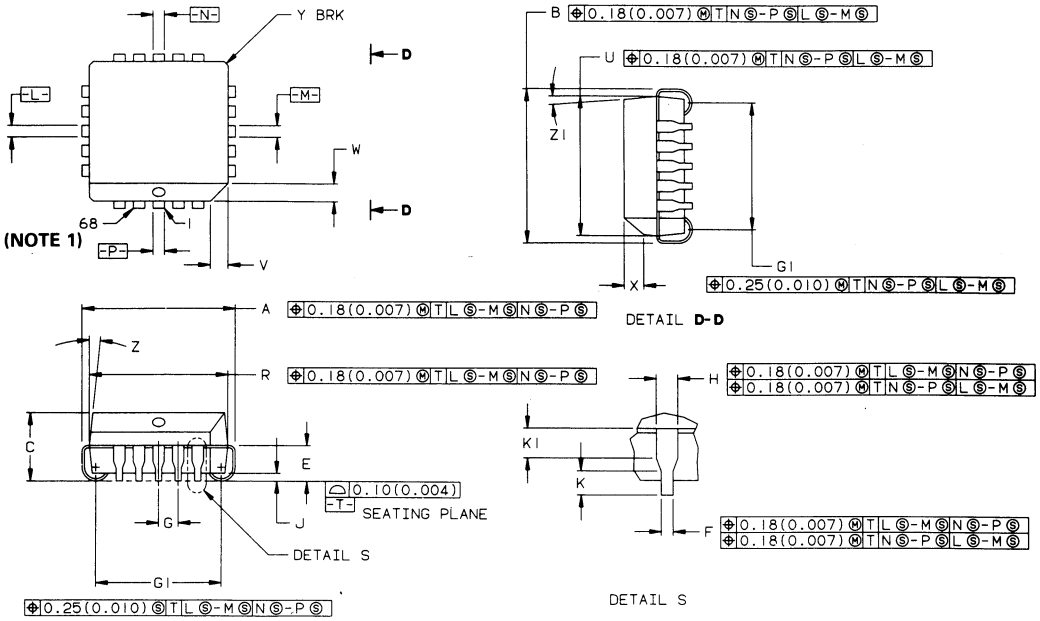
10.1 INTRODUCTION

This section contains the pin assignments and package information for the MC68HC05L6.

10.2 PIN ASSIGNMENT



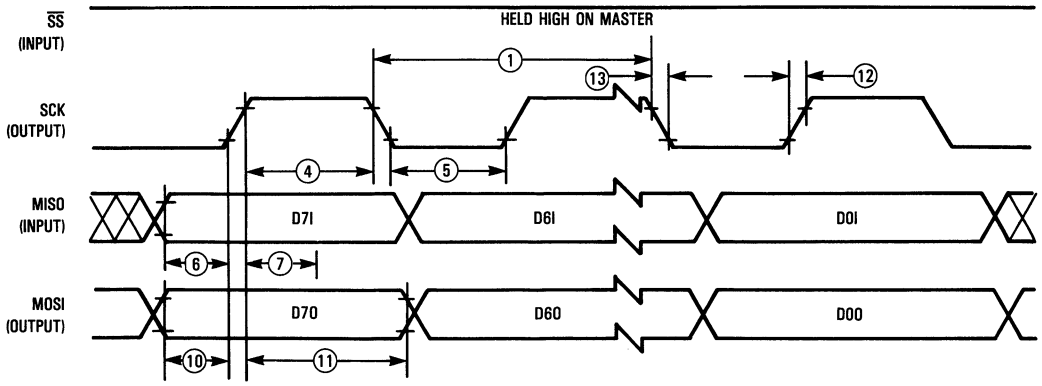
FN SUFFIX
68-LEAD PLASTIC LEADED CHIP CARRIER
CASE 779-02



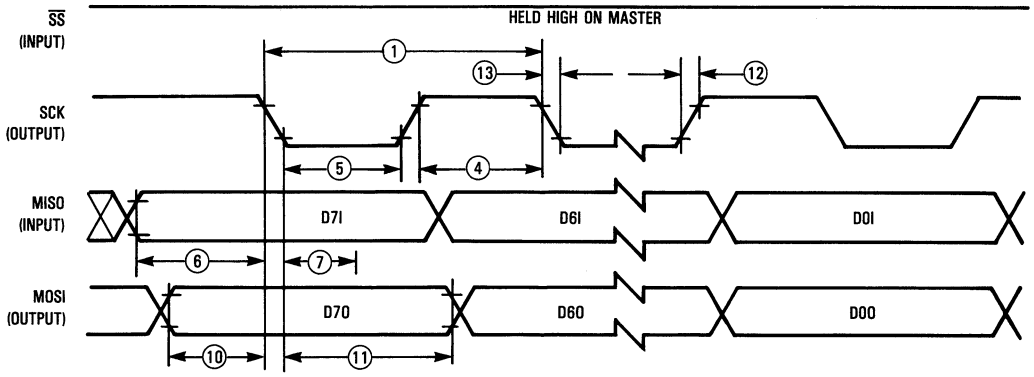
NOTES

1. **DUE TO SPACE LIMITATION, CASE 779-02 SHALL BE REPRESENTED BY A GENERAL (SMALLER) CASE OUTLINE DRAWING RATHER THAN SHOWING ALL 68 LEADS.**
2. DATUMS -L-, -M-, -N-, AND -P- DETERMINED WHERE TOP OF LEAD SHOULDER EXIT PLASTIC BODY AT MOLD PARTING LINE.
3. DIM G1, TRUE POSITION TO BE MEASURED AT DATUM -T-, SEATING PLANE.
4. DIM R AND U DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE MOLD PROTRUSION IS 0.25(0.010) PER SIDE.
5. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
6. CONTROLLING DIMENSION: INCH.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	25.02	25.27	0.985	0.995
B	25.02	25.27	0.985	0.995
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	24.13	24.28	0.950	0.956
U	24.13	24.28	0.950	0.956
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	23.12	23.62	0.910	0.930
K1	1.02	---	0.040	---
Z1	2°	10°	2°	10°

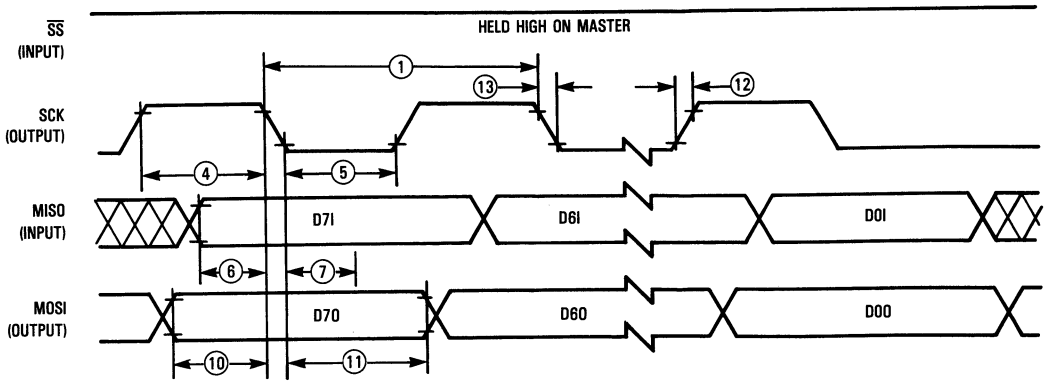


(c) SPI MASTER TIMING CPOL = 0, CPHA = 0

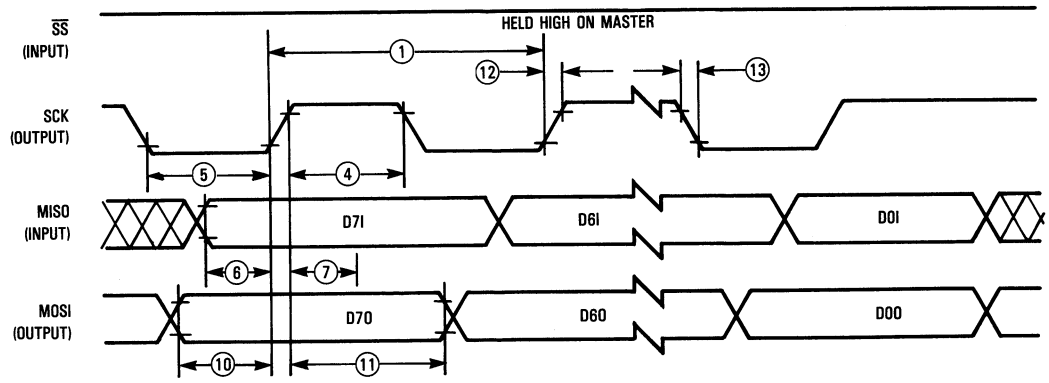


(d) SPI MASTER TIMING CPOL = 1, CPHA = 0

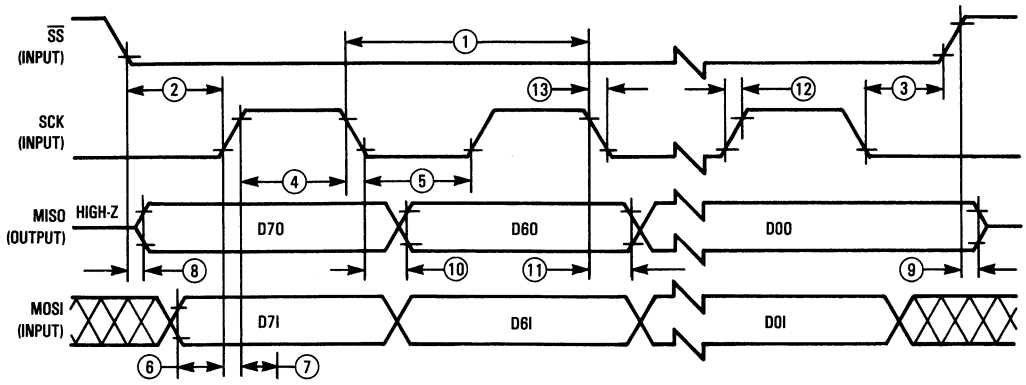
Figure 8-9. SPI Timing Diagrams (Sheet 1 of 2)



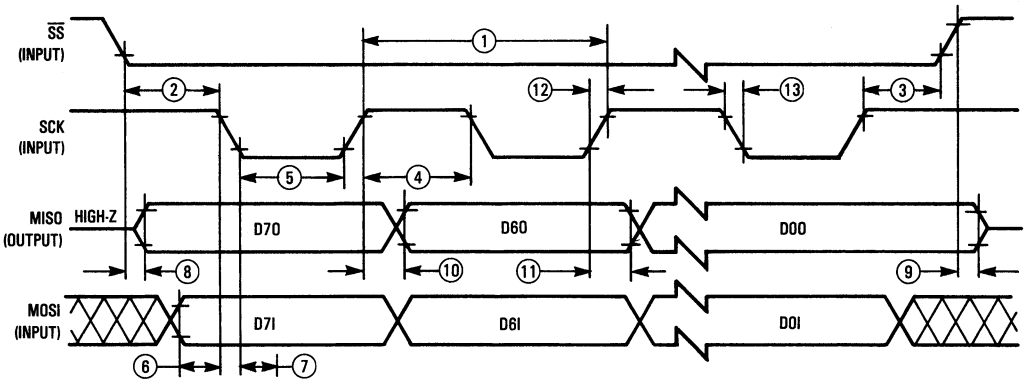
(a) SPI MASTER TIMING CPOL = 0, CPHA = 1



(b) SPI MASTER TIMING CPOL = 1, CPHA = 1



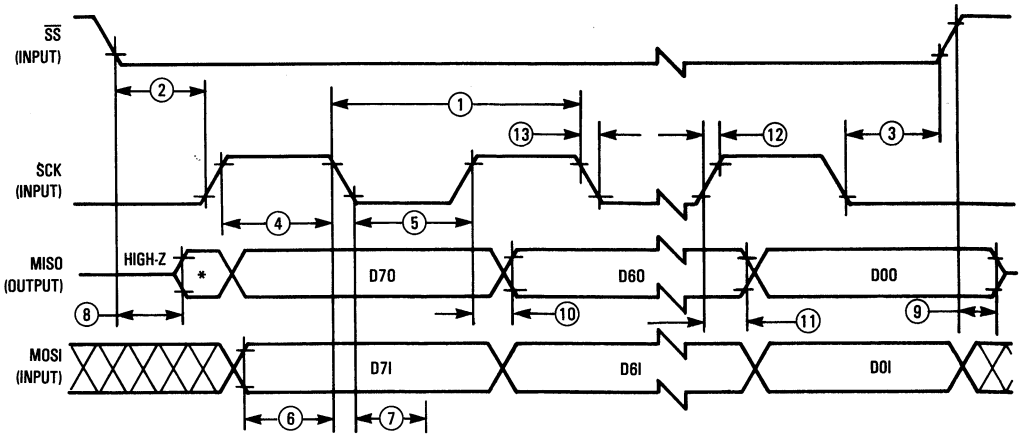
(g) SPI SLAVE TIMING CPOL = 0, CPHA = 0



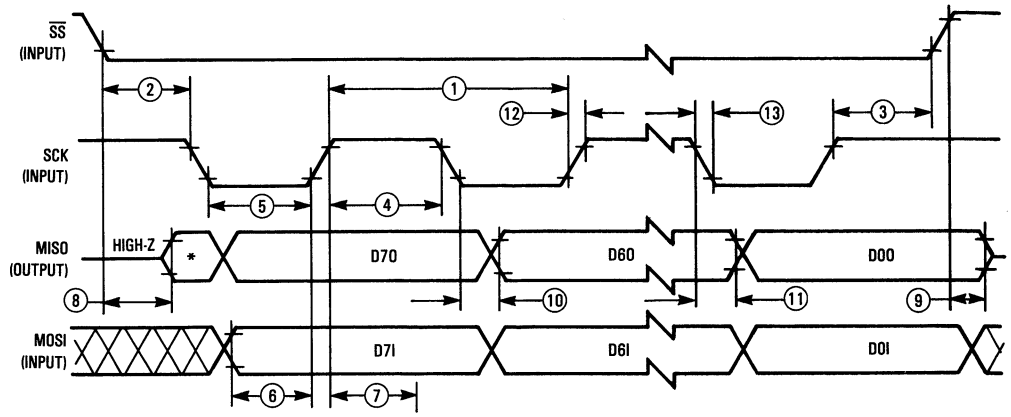
(h) SPI SLAVE TIMING CPOL = 1, CPHA = 0

NOTES:
(1) Measurement points are V_{OL}, V_{OH}, V_{IL}, and V_{IH}.
(2) *Denotes undefined, either high or low.

Figure 8-9. SPI Timing Diagrams (Sheet 2 of 2)



(e) SPI SLAVE TIMING CPOL = 0, CPHA = 1



(f) SPI SLAVE TIMING CPOL = 1, CPHA = 1

NOTES

NOTES

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.