TOSHIBA

TMP68HC05
TMP68HC11

TOSHIBA AMERICA ELECTRONIC COMPONENTS, INC.

TOSHIBA

TMP68HC05C4
TMP68HC11A8
TMP68HC11E9
TMP68C711J6

1. This technical data may be controlled under U.S. Export Administration Regulations and may be subject to the approval of the U.S. Department of Commerce prior to export. Any export or re-export directly or indirectly, in contravention of the U.S. Export Administration Regulations is strictly prohibited.

2. LIFE SUPPORT POLICY

   Toshiba products described in this databook are not authorized for use as critical components in life support systems without the written consent of the appropriate officer of Toshiba America, Inc. Life support systems are either systems intended for surgical implant in the body or systems which sustain life.

   A critical component is any component of a life support system whose failure to perform may cause a malfunction or failure of the life support system, or may affect its safety or effectiveness.

3. The information in this databook has been carefully checked and is believed to be reliable, however, no responsibility can be assumed for inaccuracies that may not have been caught. All information in this databook is subject to change without prior notice. Furthermore, Toshiba cannot assume responsibility for the use of any license under the patent rights of Toshiba or any third parties.

4. No part of this manual may be transferred or reproduced without prior permission of Toshiba Corporation.

SECTION 1
TMP68HC05C4

1. This technical data may be controlled under U.S. Export Administration Regulations and may be subject to the approval of the U.S. Department of Commerce prior to export. Any export or re-export directly or indirectly, in contravention of the U.S. Export Administration Regulations is strictly prohibited.

2. LIFE SUPPORT POLICY

   Toshiba products described in this databook are not authorized for use as critical components in life support systems without the written consent of the appropriate officer of Toshiba America, Inc. Life support systems are either systems intended for surgical implant in the body or systems which sustain life.

   A critical component is any component of a life support system whose failure to perform may cause a malfunction or failure of the life support system, or may affect its safety or effectiveness.

3. The information in this databook has been carefully checked and is believed to be reliable, however, no responsibility can be assumed for inaccuracies that may not have been caught. All information in this databook is subject to change without prior notice. Furthermore, Toshiba cannot assume responsibility for the use of any license under the patent rights of Toshiba or any third parties.

4. No part of this manual may be transferred or reproduced without prior permission of Toshiba Corporation.

CONTENTS

1. INTRODUCTION

1.1 GENERAL

The TMP68HC05C4 CMOS Microcomputer is a member of the 68HC05 Family of low-cost single-chip microcomputers. This 8-bit microcomputer unit (MCU) contains an on-chip oscillator, CPU, RAM, ROM, I/O, two serial interface systems, and timer. The fully static design allows operation at frequencies down to dc, further reducing its already low-power consumption.

1.2 FEATURES

The following are some of the hardware and software highlights of the TMP68HC05C4.

HARDWARE FEATURES

- CMOS Technology
- 8-Bit Architecture
- Power Saving Stop, Wait, and Data Retention Modes
- Fully Static Operation
- 176 Bytes of On-Chip RAM
- 4160 Bytes of On-Chip ROM
- 24 Bidirectional I/O Lines
- 2.1 MHz Internal Operating Frequency at 5 Volts;1.0 MHz at 3 Volts
- Internal 16-Bit Timer Similar to MC6801 Timer
- Serial Communications Interface System
- Serial Peripheral Interface System
- Self-Check Mode
- External, Timer, Serial Communications Interface, and Serial Peripheral Interface Interrupts
- Master Reset and Power-On Reset
- Single 3- to 6-Volt Supply (2 Volt Data Retention Mode)
- On-Chip Oscillator with RC or Crystal/Ceramic Resonator Mask Options
- 40-Pin Dual-In-Line Package
- 44-Lead PLCC (Plastic Leaded Chip Carrier) Chip Carrier Also Available

SOFTWARE FEATURES

- Similar to MC6800
- 8×8 Unsigned Multiply Instruction
- Efficient Use of Program Space
- Versatile Interrupt Handling
- True Bit Manipulation
- Addressing Modes with Indexed Addressing for Tables
- Efficient Instruction Set
- Memory Mapped I/O
- Two Power-Saving Standby Modes
- Upward Software Compatible with the M146805 CMOS Family
- Complete Development System Support on EXOR ciser and HDS-200

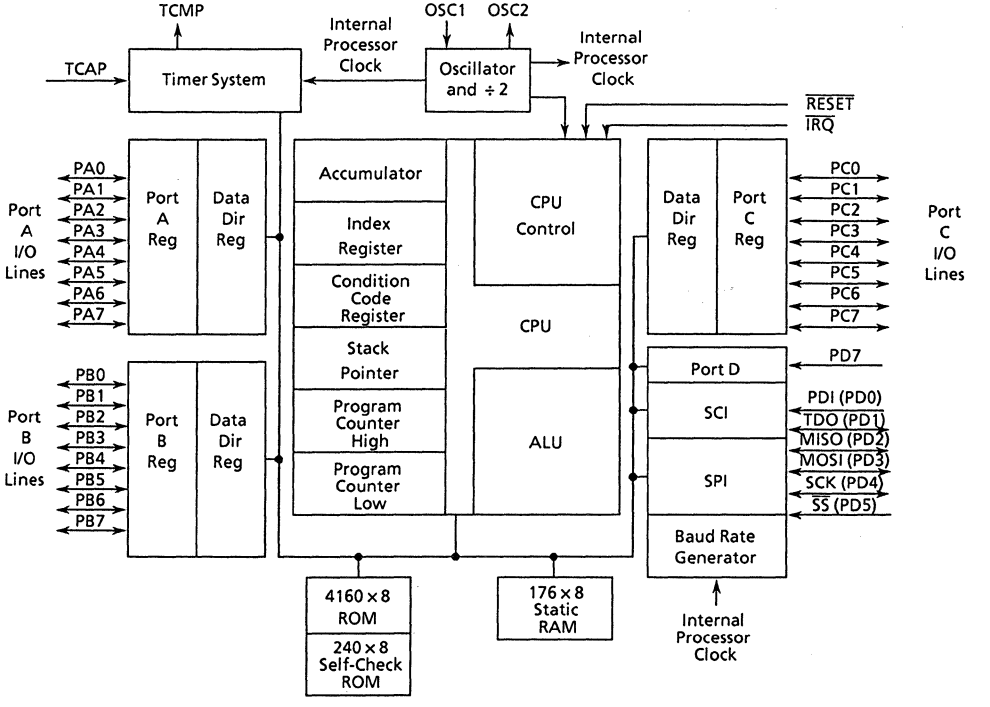


Figure 1.1 TMP68HC05C4 Microcomputer Block Diagram

2. FUNCTIONAL PIN DESCRIPTION, INPUT/OUTPUT PROGRAMMING, MEMORY, CPU REGISTERS, AND SELF-CHECK

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

2.1 FUNCTIONAL PIN DESCRIPTION

2.1.1 V_{DD} and V_{SS}

Power is supplied to the MCU using these two pins. V_{DD} is power and V_{SS} is ground.

2.1.2 $\overline{IRQ}$ (Maskable Interrupt Request)

$\overline{IRQ}$ is a mask programmable option which provides two different choices of interrupt triggering sensitivity. These options are: 1) negative edge-sensitive triggering only, or 2) both negative edge-sensitive and level-sensitive triggering. In the latter case, either type of input to the $\overline{IRQ}$ pin will produce the interrupt. The MCU 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 an interrupt has been requested. When the MCU completes its current instruction, the interrupt latch is tested. If the interrupt latch contains a logic one, and the interrupt mask bit (1 bit) in the condition code register is clear, the MCU then begins the interrupt sequence.

If the option is selected to include level-sensitive triggering, then 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.

2.1.3 $\overline{RESET}$

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

2.1.4 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.

2.1.5 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.

2.1.6 OSC1, OSC2

The TMP68HC05C4 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}).

2.1.6.1 CRYSTAL.

The circuit shown 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 9.7 or 9.8 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 paragraph 9.5 or 9.6 for V_{DD} specifications.

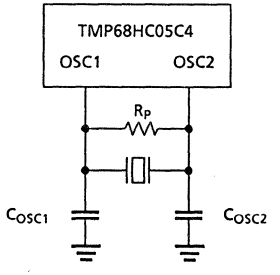
Crystal

	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

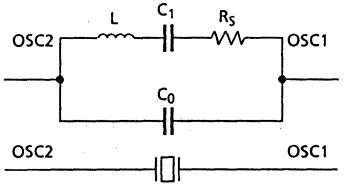
Ceramic Resonator

	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	–

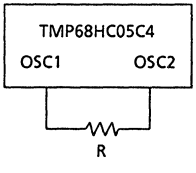
(a) Crystal/Ceramic Resonator Parameters



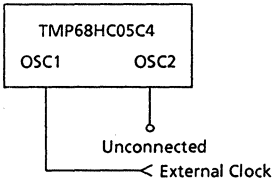
(b) Crystal/Ceramic Resonator Oscillator Connections



(c) Equivalent Crystal Circuit



(d) RC Oscillator Connections



(e) External Clock Source Connections
(Either Crystal or RC Mask Options)

Figuer 2.1 Oscillator Connections

2.1.6.2 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.

2.1.6.3 RC.

If the RC oscillator option 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.

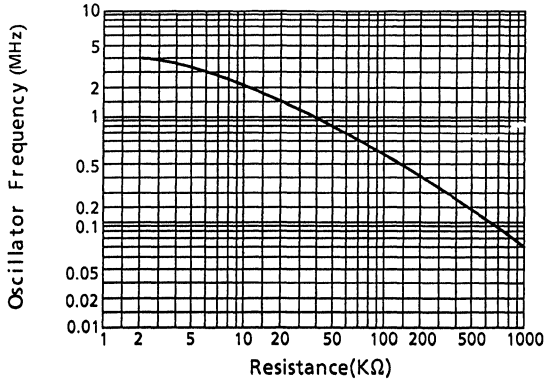


Figure 2.2 Typical Frequency vs Resistance for RC Oscillator Option Only(V_{DD}=5.0V)

2.1.6.4 EXTERNAL CLOCK.

An external clock should be applied to the OSC1 input with the OSC2 pin not connected, as shown in Figure 2.1(e). An external clock may be used with either the RC or crystal oscillator option. The t_{OXOV} or t_{ILCH} specifications do not apply when using an external clock input. The equivalent specification of the external clock source should be used in lieu of t_{OXOV} or t_{ILCH}.

2.1.7 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 INPUT/OUTPUT PROGRAMMING paragraph for a detailed description of I/O programming.

2.1.8 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 INPUT/OUTPUT PROGRAMMING paragraph for a detailed description of I/O programming.

2.1.9 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 INPUT/OUTPUT PROGRAMMING paragraph for a detailed description of I/O programming.

2.1.10 PD0-PD5, PD7

These seven lines comprise port D, a fixed input port that is enabled during power-on. All enabled special functions (SPI and SCI) affect the pins on this port. Four of these lines, PD2/MISO, PD3/MOSI, PD4/SCK, and PD5/$\overline{SS}$, are used in the serial peripheral interface (SPI) discussed in Section 6. Two of these lines, PD0/PDI and PD1/TDO, are used in the serial communications interface (SCI) discussed in Section 5. Refer to 2.2 INPUT/OUTPUT PROGRAMMING for a detailed description of I/O programming.

2.2 INPUT/OUTPUT PROGRAMMING

2.2.1 Parallel Ports

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 direction 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. A pin is configured as an input if its corresponding DDR bit is cleared to a logic zero. At power-on or reset, all DDRs are cleared, which configure 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. During the programmed output state, 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.

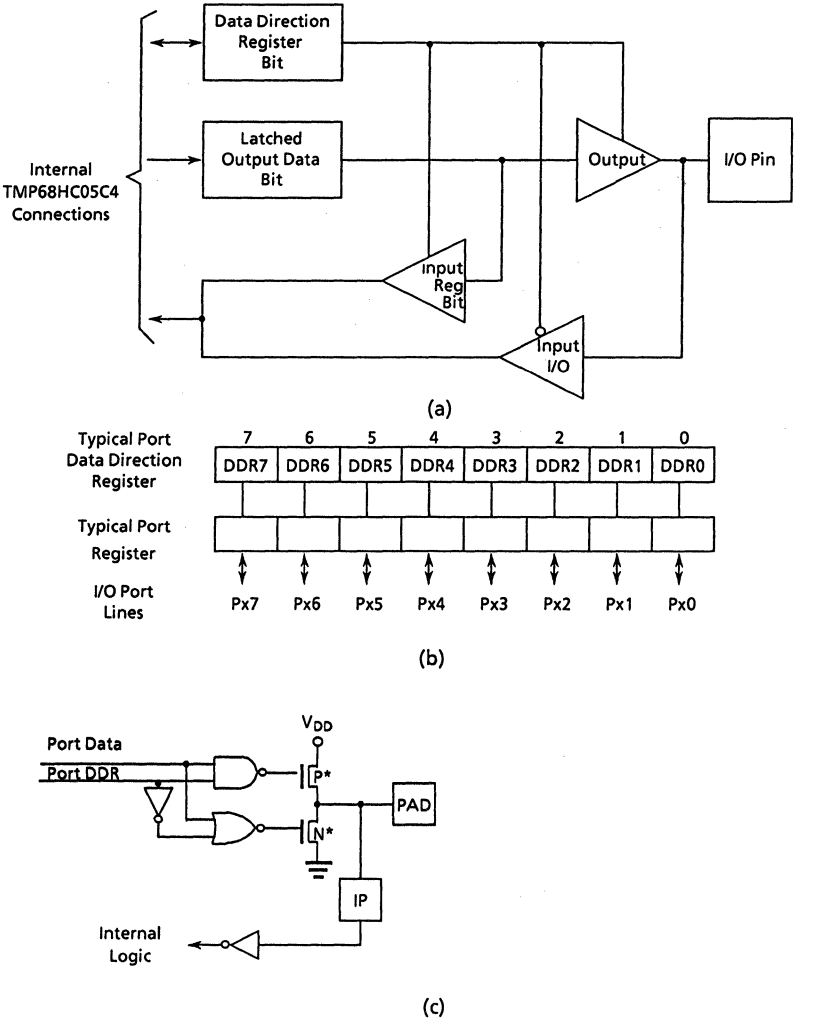
2.2.2 Fixed Port

Port D is a 7-bit fixed input port (PD0-PD5, PD7) that continually monitors the external pins whenever the SPI or SCI systems are disabled. During power-on reset or external reset all seven bits become valid input ports because all special function output drivers are disabled. For example, with the serial communications interface (SCI) system enabled, (RE=TE=1) PD0 and PD1 inputs will read zero. With the serial peripheral interface (SPI) system disabled (SPE=0) PD2 through PD5 will read the state of the pin at the time of the read operation. No data register is associated with the port when it is used as an input.

Note: It is recommended that all unused inputs and I/O ports be tied to an appropriate logic level (e.g., either V_{DD} or V_{SS}).

2.2.3 Serial Port (SCI and SPI)

The serial communications interface (SCI) and serial peripheral interface (SPI) use the port D pins for their functions. The SCI function requires two of the pins (PD0-PD1) for its receive data input (RDI) and transmit data output (TDO) respectively, whereas the SPI function requires four of the pins (PD2-PD5) for its serial data input/output (MISO), serial data output/input (MOSI), system clock (SCK), and slave select ($\overline{SS}$) respectively. Refer to SECTION 5 SERIAL COMMUNICATIONS INTERFACE and SECTION 6 SERIAL PERIPHERAL INTERFACE for a more detailed discussion.



Notes :

1. *Denotes devices have same physical size, and are enhancement type.
2. IP = Input protection.
3. Latch-up protection not shown.

Figure 2.3 Parallel Port I/O Circuitry

2.3 MEMORY

As shown in Figure 2.4, the MCU is capable of addressing 8192 bytes of memory and I/O registers with its program counter. The TMP68HC05C4 MCU has implemented 4601 bytes of these locations. The first 256 bytes of memory (page zero) include: 25 bytes of I/O features such as data ports, the port DDRs, timer, serial peripheral interface (SPI), and serial communication interface (SCI);48 bytes of user ROM, and 176 bytes of RAM. The next 4096 bytes complete the user ROM. The self-check ROM (224 bytes) and self-check vectors (16 bytes) are contained in memory locations $1F00 through $1FEF. The 16 highest address bytes contain the user defined reset and the interrupt vectors. Seven bytes of the lowest 32 memory locations are unused and the 176 bytes of user RAM include up to 64 bytes for the stack. Since most programs use only a small part of the allocated stack locations for interrupts and/or subroutine stacking purposes, the unused bytes are usable for program data storage.

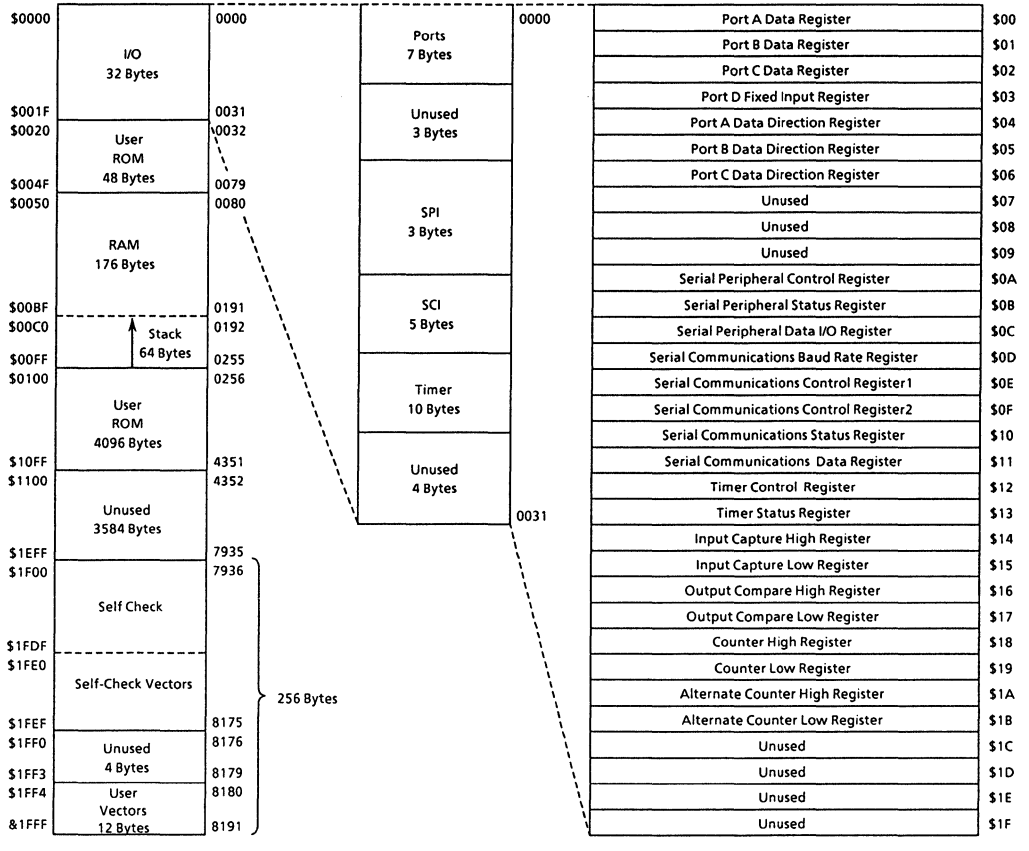


Figure 2.4 TMP68HC05C4 Memory Map

2.4 CPU REGISTERS

The TMP68HC05C4 CPU contains five registers, as shown in the programming model of Figure 2.5. The interrupt stacking order is shown in Figure 2.6.

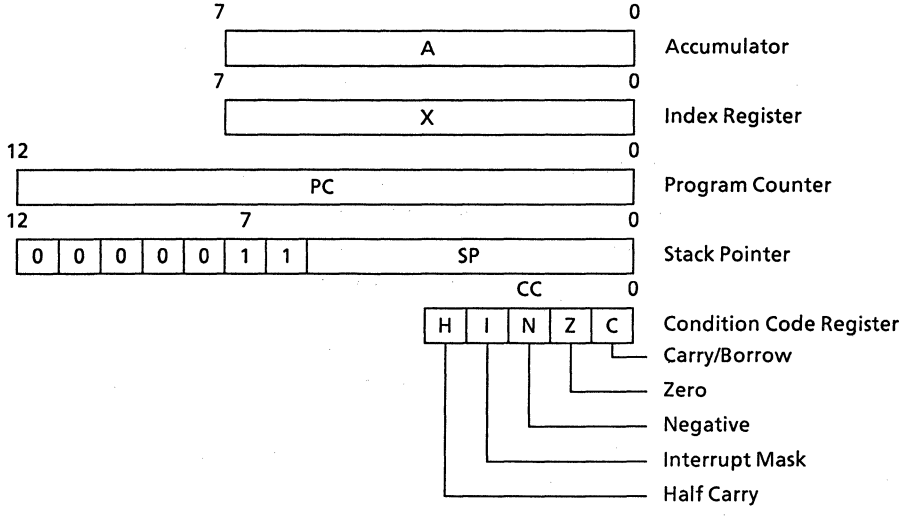
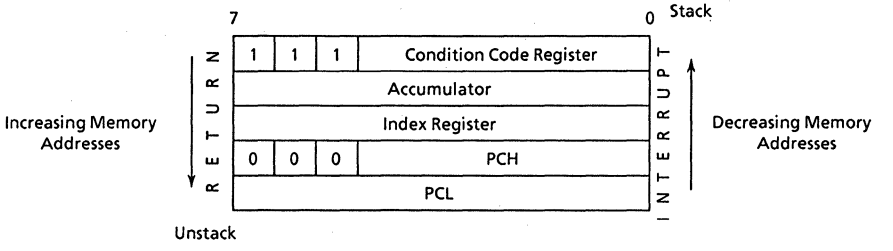


Figure 2.5 Programming Model



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

2.4.1 Accumulator (A)

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

2.4.2 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.

2.4.3 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.

2.4.4 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.

2.4.5 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.

2.4.5.1 HALF CARRY BIT (H).

The H bit is set to a one when a carry occurs between bits 3 and 4 of the ALU during an ADD or ADC instruction. The H bit is useful in binary coded decimal subroutines.

2.4.5.2 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 bits 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, SECTION 5 SERIAL COMMUNICATIONS INTERFACE, and SECTION 6 SERIAL PERIPHERAL INTERFACE for more information).

2.4.5.3 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).

2.4.5.4 ZERO (Z).

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation is zero.

2.4.5.5 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.

2.5 SELF-CHECK

The self-check capability of the TMP68HC05C4 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 V dc input (through a 4.7 KΩ resistor) to the $\overline{IRQ}$ pin and 5 V dc input (through a 4.7 KΩ 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

SCI – Transmission Test;checks for RDRF, TDRE, TC, and FE flags

ROM – Exclusive OR with odd ones parity result

SPI – Transmission test with check for SPIF, WCOL, and MODF flags

INTERRUPTS – Tests external, timer, SCI, and SPI interrupts.

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.

2.5.1 Timer Test Subroutine

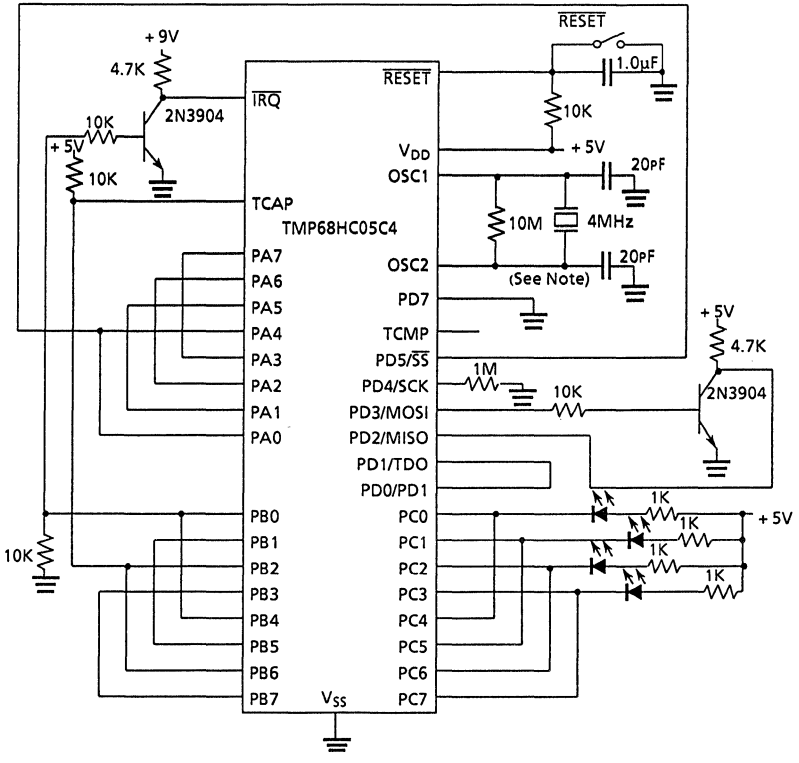
This subroutine returns with the Z bit cleared if any error is detected;otherwise, the Z bit is set.

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.

2.5.2 ROM Checksum 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 $1F93 with RAM location $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.



Note : The RC Oscillator Option may also be used in this circuit.

Figuer 2.7 Self-Check Circuit Schematic Diagram

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 SCI
1	1	0	1	Bad ROM
1	1	1	0	Bad SPI
1	1	1	1	Bad Interrupts or $\overline{IRQ}$ Request
Flashing				Good Device
Ail Others				Bad Device, Bad Port C, etc.

0 Indicates LED on;1 Indicates LED is off.

3. RESETS, INTERRUPTS, LOW POWER, AND DATA RETENTION MODES

3.1 RESETS

The TMP68HC05C4 has two reset modes:an active low external reset pin ($\overline{\text{RESET}}$) and a power-on reset function;refer to Figure 3.1.

3.1.1 $\overline{\text{RESET}}$ Pin

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 t_{cyc}. The $\overline{\text{RESET}}$ pin contains an internal Schmitt Trigger as part of its input to improve noise immunity.

3.1.2 Power-On Reset

The power-on reset occurs when a positive transition is detected on V_{DD}. 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 circuity provides for a 4064 t_{cyc} delay from the time that the oscillator becomes active. If the external $\overline{\text{RESET}}$ pin is low at the end of the 4064 t_{cyc} time out, the processor remains in the reset condition unit $\overline{\text{RESET}}$ goes high. The user must ensure that V_{DD} has risen to a point where the MCU can operate properly prior to the time the 4064 POR reset cycles have elapsed. If there is doubt, the external $\overline{\text{RESET}}$ pin should remain low until such time that V_{DD} has risen to the minimum operating voltage specified.

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).

3.2 INTERRUPTS

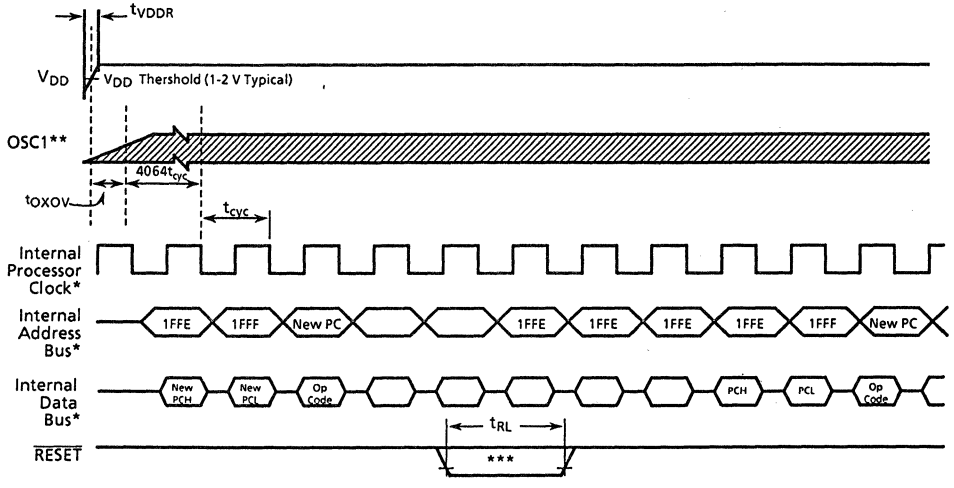
Systems often require that normal processing be interrupted so that some external event may be serviced. The TMP68HC05C4 may be interrupted by one of five different methods:either one of four maskable hardware interrupts I($\overline{\text{IRQ}}$, SPI, SCI, or Timer) and one non-maskable software interrupt (SWI). Interrupts such as Timer, SPI, and SCI 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.

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) 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, in the TMP68HC05C4 is provided in Table 3.2.



* 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	×	×
Timer counter configured to $FFFC	×	×
Timer output compare (TCMP) bit reset to zero	×	×
All timer interrupt enable bits cleared (ICIE, OCIE, and TOIE) to disable timer interrupts. The OLVL timer bit is also cleared by reset.	×	×
All data direction registers cleared to zero (input)	×	×
Configure stack pointer to $00FF	×	×
Force internal address bus to restart vector ($1FFE-$1FFF)	×	×
Set bit in condition code register to a logic one	×	×
Clear STOP latch	×*	×
Clear external interrupt latch	×	×
Clear WAIT latch	×	×
Disable SCI (serial control bits TE = 0 and RE = 0). Other SCI bits cleared by reset include: TIE, TCIE, RIE, ILIE, RWU, SBK, RDRF, IDLE, OR, NF, and FE.	×	×
Disable SPI (serial output enable condtrol bit SPE = 0). Other SPI bits cleared by reset include: SPIE, MSTR, SPIF, WCOL, and MODF.	×	×
Set serial status bits TDRE and TC	×	×
Clear all serial interrupt enable bits (SPIE, TIE, and TCIE)	×	×
Place SPI system in slave mode (MSTR = 0)	×	×
Clear SCI prescaler rate control bits SCP0-SCP1	×	×

*Indicates that timeout still occurs.

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	Exdternal Interrupt	$\overline{\text{IRQ}}$	$1FFA-$1FFB
Timer Status	ICF	Input Capture	TIMER	$1FF8-$1FF9
	OCF	Output Compare		
	TOF	Timer Overflow		
SCI Status	TDRE	Transmit Buffer Empty	SCI	$1FF6-$1FF7
	TC	Transmit Complete		
	RDRF	Receiver Buffer Full		
	IDLE	Idle Line Detect		
	OR	Overrun		
SPI Status	SPIF	Transfer Complete	SPI	$1FF4-$1FF5
	MODF	Mode Fault		

3.2.1 Hardware 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. Much of the MCU is configured to a known state during this type of reset as previously described in RESETS paragraph 3.1.

(b) 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.

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

3.2.2 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.

3.2.3 External Interrupt

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.

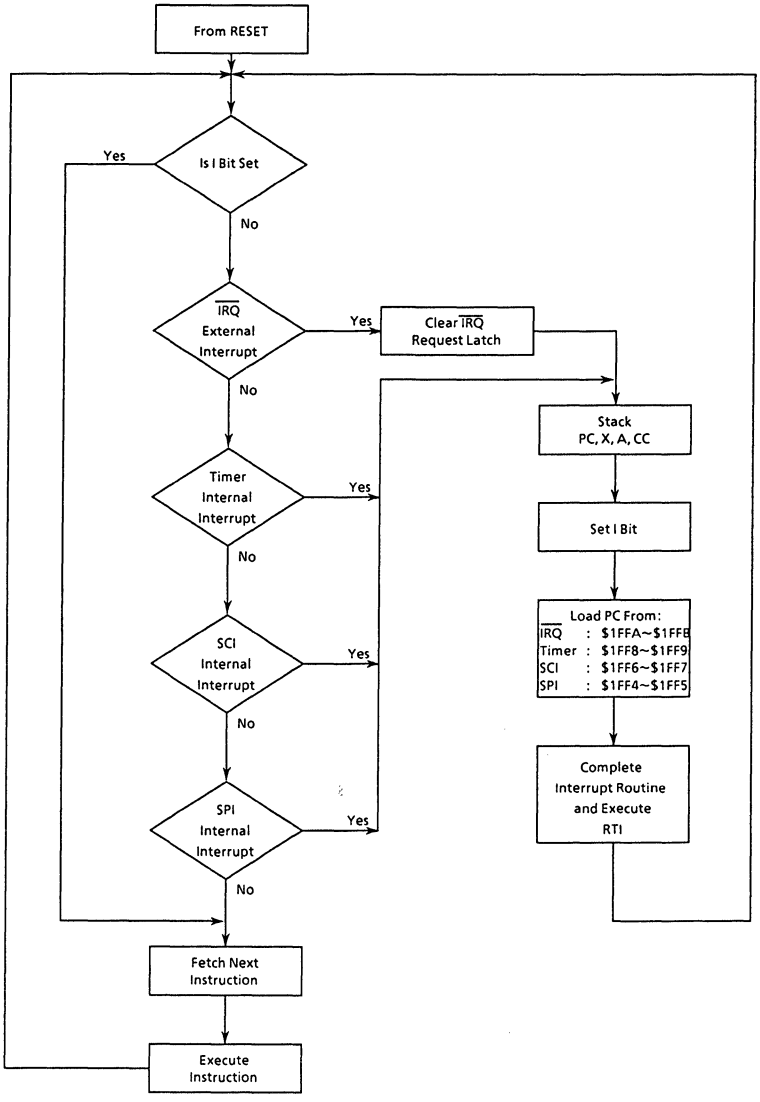


Figure 3.2 Hardware Interrupt Flowchart

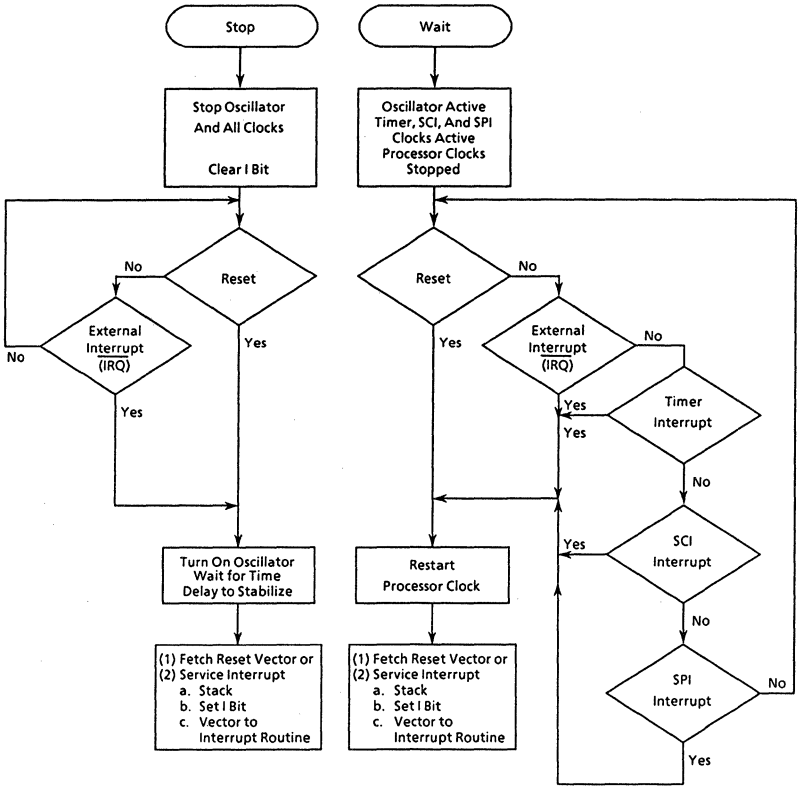
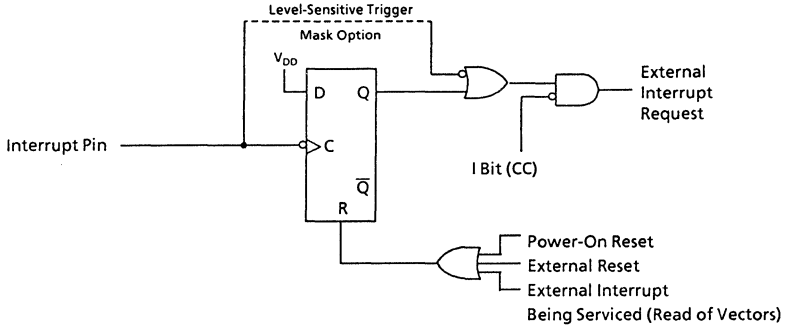
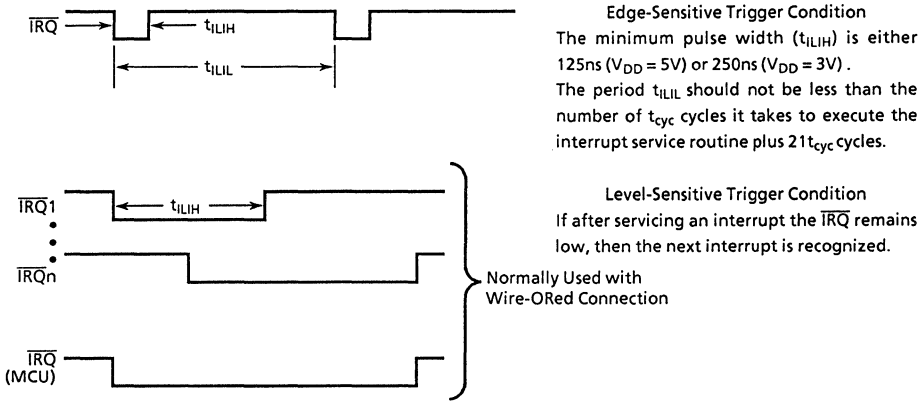


Figure 3.3 STOP/WAIT Flowcharts



(a)Interrupt Function Diagram



(b)Interrupt Mode Diagram

Figure 3.4 External Interrupt

3.2.4 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).

All interrupt 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 I bit is set. This masks further interrupts until the present one is serviced. The interrupt service routine address is specified by the contens 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.

3.2.5 Serial Communications Interface (SCI) Interrupts

An interrupt in the serial communications interface (SCI) occurs when one of the interrupt flag bits in the serial communications status register is set, provided the I bit in the condition code register is clear and the enable bit in the serial communications control register 2 (location $0F) 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 SCI interrupt causes the program counter to vector to memory location $1FF6 and $1FF7 which contains the starting address of the interrupt service routine. Software in the serial interrupt service routine must determine the priority and cause of the SCI interrupt by examining the interrupt flags and the status bits located in the serial communications status register (location $10). The general sequence for clearing an interrupt is a software sequence of accessing the serial communications status register while the flag is set followed by a read or write of an associated register. Refer to SECTION 5 SERIAL COMMUNICATIONS INTERFACE for a description of the SCI system and its interrupts.

3.2.6 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 6 SERIAL PERIPHERAL INTERFACE for a description of the SPI system and its interrupts.

3.3 LOW POWER MODES

3.3.1 STOP Instruction

The STOP instruction places the TMP68HC05C4 in its lowest power consumption mode. In the STOP 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.

3.3.2 WAIT Instruction

The WAIT instruction places the TMP68HC05C4 in a low power consumption mode, but the WAIT mode consumes somewhat more power than the STOP mode. In the WAIT mode, the internal clock remains active, and all CPU processing is stopped;however, the programmable timer, serial peripheral interface, and serial communications interface systems 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 continues until any interrupt or reset is sensed. At this time the program counter vectors to the memory location ($1FF4 through $1FFF) which contains the starting address of the interrupt or reset service routine.

3.4 DATA RETENTION MODE

The contents of RAM and CPU registers are retained at supply voltages as low as 2.0 Vdc. This is referred to as the data retention mode, where the data is held, but the device is not guaranteed to operate.

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 the high and low represent the significance of the byte). A description of each register is provided below.

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, and

Alternate Counter Low Register location $1B.

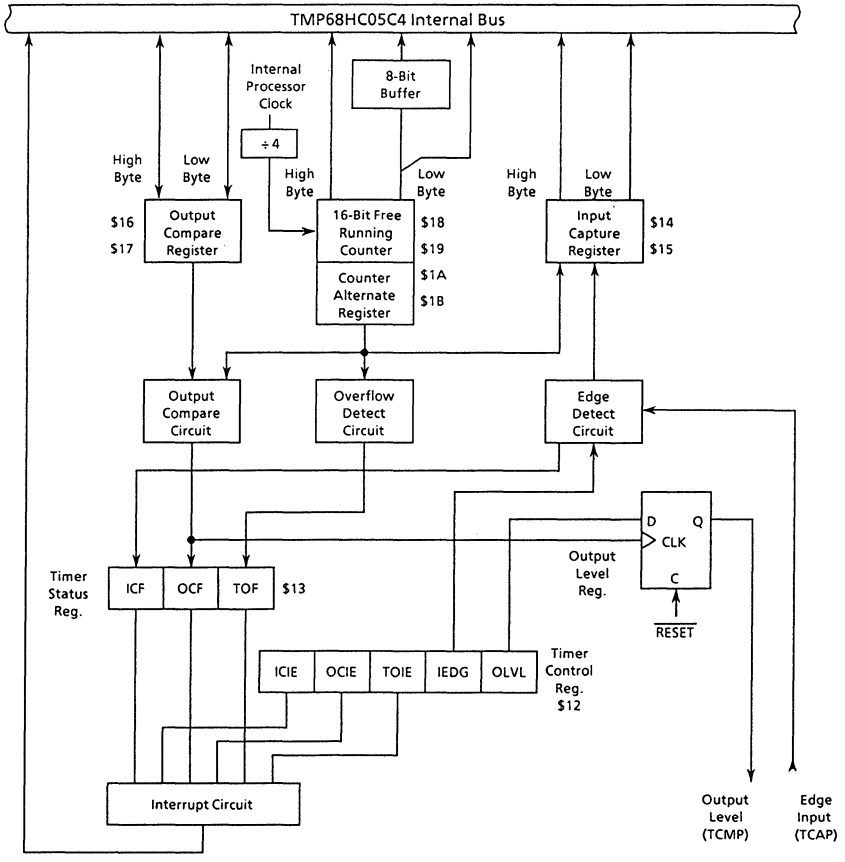
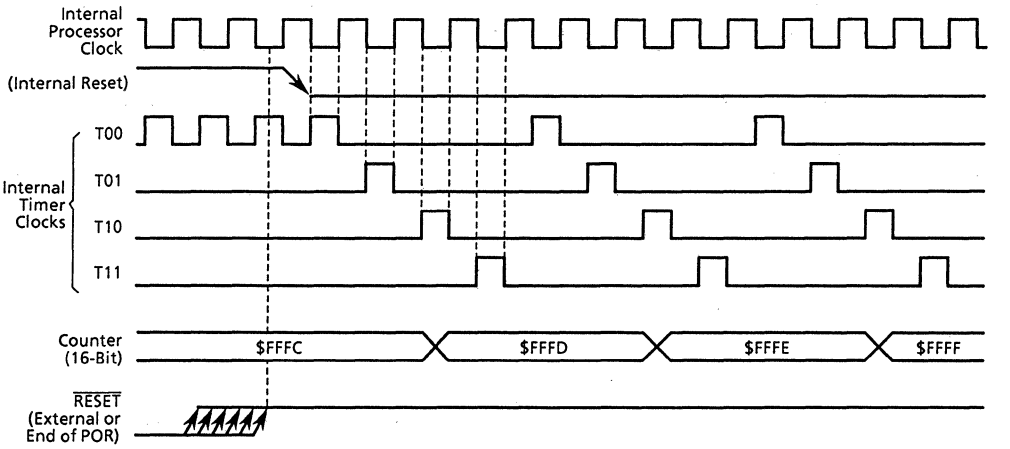
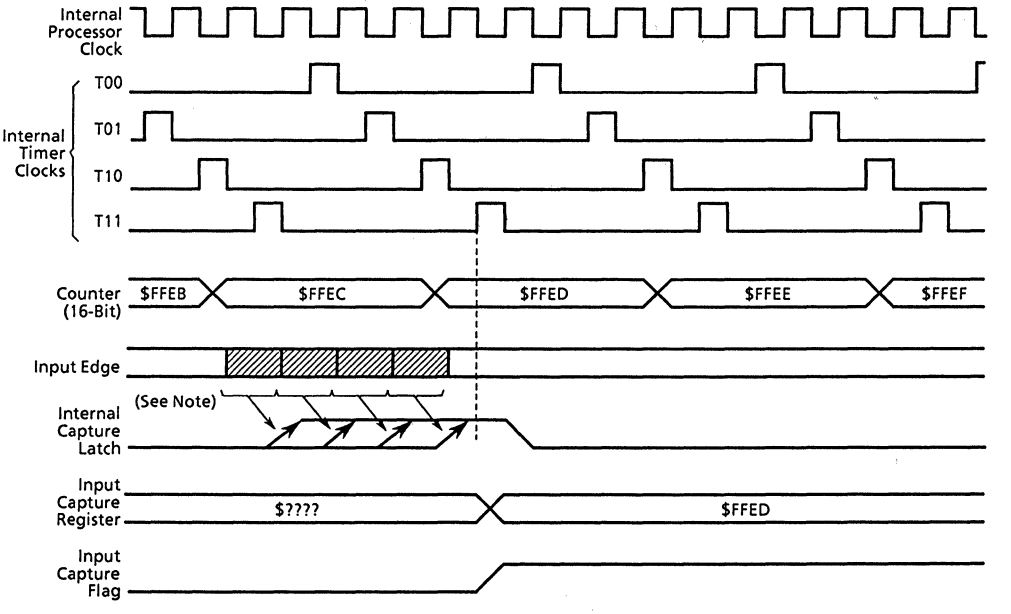


Figure 4.1 Programmable Timer Block Diagram



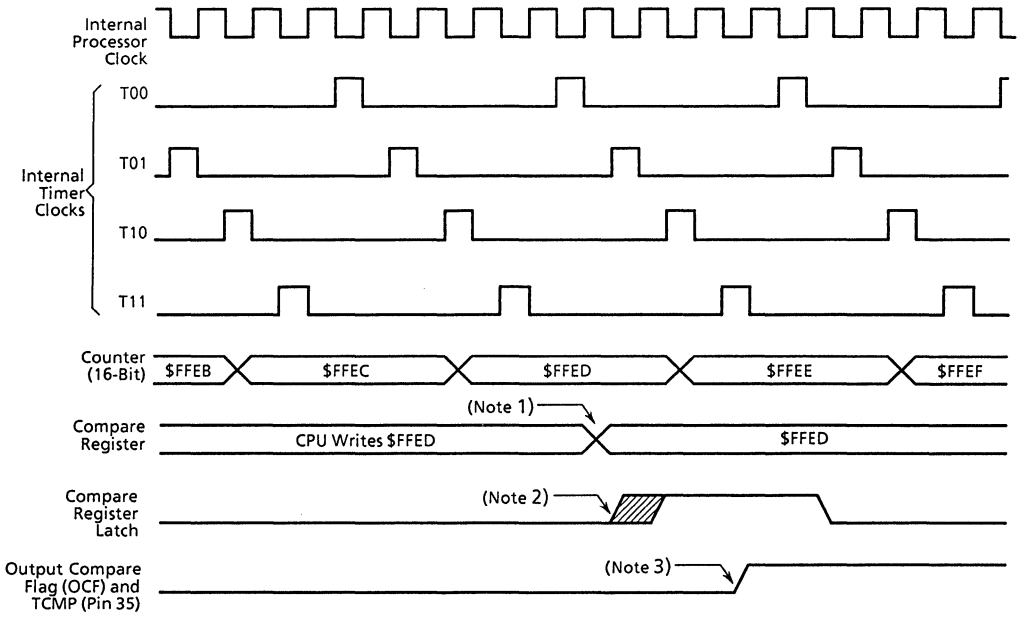
Note : The Counter Register and Timer Control Register are the only ones affected by $\overline{\text{RESET}}$.

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.

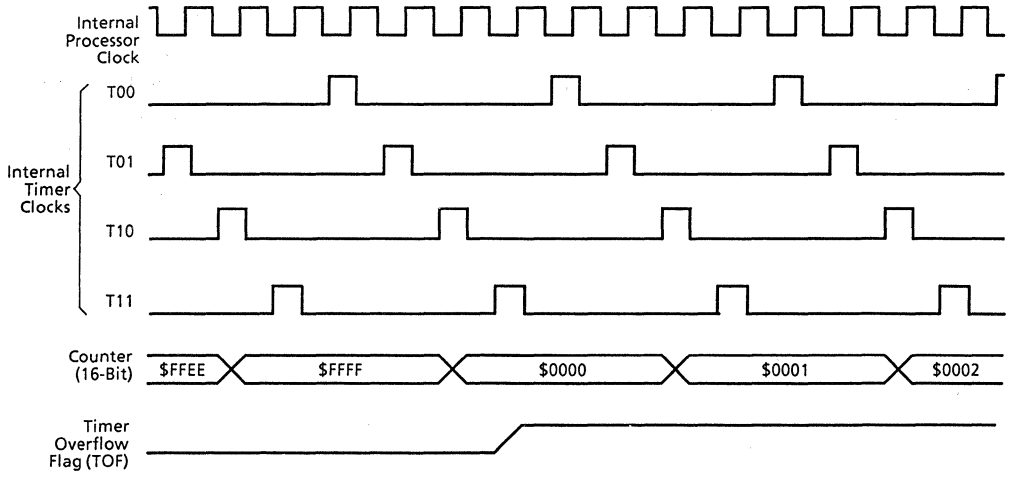
Figure 4.3 Timer State Timing Diagram for Input Capture



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

4.2 COUNTER

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 (POR), 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 MPU 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).

4.3 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 utilized, 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 lest 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 input of the output level register regardless of whether the output compare flag (OCF) is set or clear. A valid output compare must occur before the OLVL bit becomes available at the output compare pin (TCMP).

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 to 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 to the low byte of the output compare register 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
```

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.

A read of the 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 TIMER CONTROL REGISTER (TCR)

The timer control register (TCR, location $12) 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	
ICIE	OCIE	TOIE	0	0	0	IEDG	OLVL	$12

B7, ICIE — If the input capture interrupt enable (ICIE) 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.

B6, OCIE — If the output compare interrupt enable (OCIE) 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.

B5, TOIE — If the timer overflow interrupt enable (TOIE) 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.

B1, IEDG — The value of the input edge (IEDG) bit determines which level transition on TCMP pin 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

B0, OLVL The value of the output level (OLVL) bit is clocked into the output level register by the next successful output compare and will appear at TCMP pin. This bit and the output level register are cleared by reset.

0 = low output

1 = high output

4.6 TIMER STATUS REGISTER (TSR)

The timer status register (TSR) 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 at TCAP pin 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 Figure 4.2, 4.3, and 4.4 for timing relationship to the timer status register bits.

7	6	5	4	3	2	1	0	
ICF	OCF	TOF	0	0	0	0	0	$13

B7, ICF The input capture flag (ICF) 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.

B6, OCF The output compare flag (OCF) 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.

B5, TOF The timer overflow flag (TOF) 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.

During STOP and WAIT instructions, the programmable timer functions as follows: during the wait mode, the timer continues to operate normally and may generate an interrupt to trigger the CPU out of the wait state;during the stop mode, the timer holds at its current state, retaining all data, and resumes operation from this point when an external interrupt is received.

5. SERIAL COMMUNICATIONS INTERFACE (SCI)

5.1 INTRODUCTION

A full-duplex asynchronous serial communications interface (SCI) is provided with a standard NRZ format and a variety of baud rates. The SCI transmitter and receiver are functionally independent, but use the same data format and bit rate. The serial data format is standard mark/space (NRZ) which provide one start bit, eight or nine data bits, and one stop bit. "Baud" and "bit rate" are used synonymously in the following description.

5.1.1 SCI Two Wire System Features

- Standard NRZ (mark/space) format.
- Advanced error detection method includes noise detection for noise duration of up to 1/16 bit time.
- Full-duplex operation (simultaneous transmit and receive).
- Software programmable for one of 32 different baud rates.
- Software selectable word length (eight or nine bit words).
- Separate transmitter and receiver enable bits.
- SCI may be interrupt driven.
- Four separate enable bits available for interrupt control.

5.1.2 SCI Receiver Features

- Receiver wake-up function (idle or address bit).
- Idle line detect.
- Framing error detect.
- Noise detect.
- Overrun detect.
- Receiver data register full flag.

5.1.3 SCI Transmitter Features

- Transmit data register empty flag.
- Transmit complete flag.
- Break send.

Any SCI two-wire system requires receive data in (RDI) and transmit data out (TDO).

5.2 DATA FORMAT

Receive data in (RDI) or transmit data out (TDO) is the serial data which is presented between the internal data bus and the output pin (TDO), and between the input pin (RDI) and the internal data bus. Data format is as shown for the NRZ in Figure 5.1 and must meet the following criteria:

(1) A high level indicates a logic one and a low level indicates a logic zero.

(2) The idle line is in a high (logic one) state prior to transmission/reception of a message.

(3) A start bit (logic zero) is transmitted/received indicating the start of a message.

(4) The data is transmitted and received least-significant-bit first.

(5) A stop bit (high in the tenth or eleventh bit position) indicates the byte is complete.

(6) A break is defined as the transmission or reception of a low (logic zero) for some multiple of the data format.

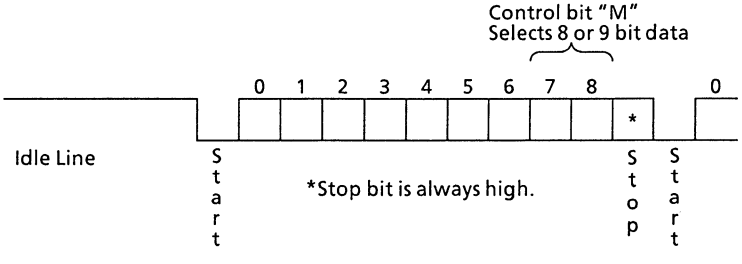


Figure 5.1 Data Format

5.3 WAKE-UP FEATURE

In a typical multiprocessor configuration, the software protocol will usually identify the addresse (s) at the beginning of the message. In order to permit uninterested MPUs to ignore the remainder of the message, a wake-up feature is included whereby all further SCI receiver flag (and interrupt) processing can be inhibited until its data line returns to the idle state. An SCI receiver is re-enabled by an idle string of at least ten (or eleven) consecutive ones. Software for the transmitter must provide for the required idle string between consecutive messages and prevent it from occurring within messages.

The user is allowed a second method of providing the wake-up feature in lieu of the idle string discussed above. This method allows the user to insert a logic one in the most significant bit of the transmit data word which needs to be received by all "sleeping" processors.

5.4 RECEIVE DATA IN

Receive data in is the serial data which is presented from the input pin via the SCI to the internal data bus. While waiting for a start bit, the receiver samples the input at a rate which is 16 times higher than the set baud rate. This 16 times higher-than-baud rate is referred to as the RT rate in Figures 5.2 and 5.3, and as the receiver clock in Figure 5.7. When the input (idle) line is detected low, it is tested for three more sample times (referred to as the start edge verification samples in Figure 5.2). If at least two of these three verification samples detect a logic low, a valid start bit is assumed to have been detected (by a logic low following the three start qualifiers) as shown in Figure 5.2; however, if in two or more of the verification samples a logic high is detected, the line is

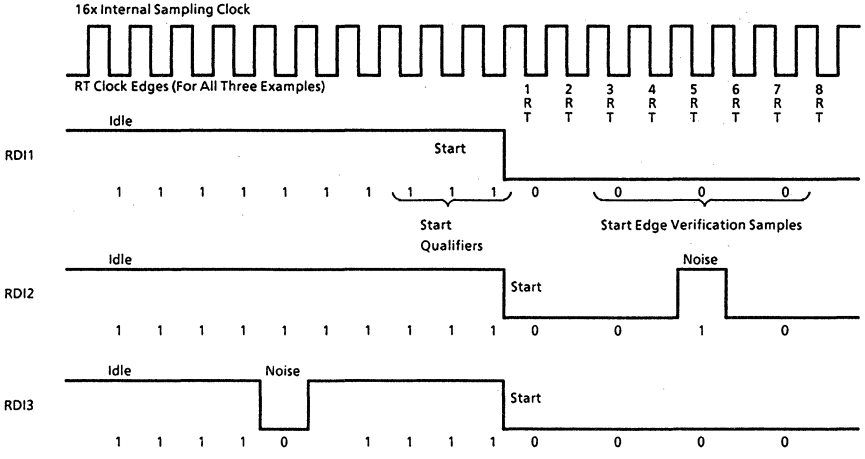


Figure 5.2 Examples of Start-Bit Sampling Technique

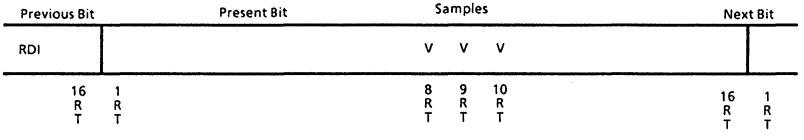


Figure 5.3 Sampling Technique Used on All Bits

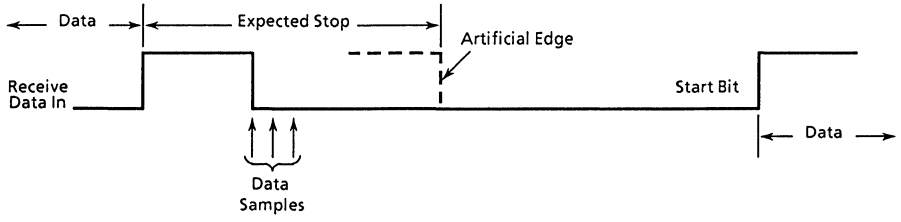
assumed to be idle. (A noise flag is set if one of the three verification samples detects a logic high, thus a valid start bit could be assumed and a noise flag still set.) The receiver clock generator is controlled by the baud rate register (see Figures 5.6 and 5.7);however, the serial communications interface is synchronized by the start bit (independent of the transmitter).

Once a valid start bit is detected, the start bit, each data bit, and the stop bit are sampled three times at RT intervals of 8RT, 9RT, and 10RT (1RT is the position where the bit is expected to start) as shown in Figure 5.3. The value of the bit is determined by voting logic which takes the value of the majority of samples (two or three out of three). A noise flag is set when all three samples on a valid start bit or a data bit or the stop bit do not agree. (As discussed above, a noise flag is also set when the start bit verification samples do not agree.)

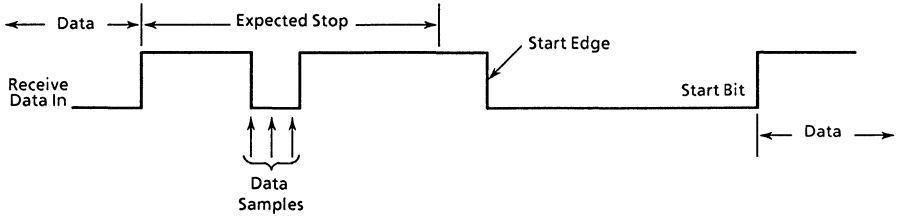
5.5 START BIT DETECTION FOLLOWING A FRAMING ERROR

If there has been a framing error without detection of a break (10 zeros for 8-bit format or 11 zeros for 9-bit format), the circuit continues to operate as if there actually were a stop bit and the start edge will be placed artificially. The last bit received in the data shift register is inverted to a logic one, and the three logic one start qualifiers (shown in Figure 5.2) are forced into the sample shift register during the interval when detection of a start bit is anticipated (see Figure 5.4);therefore the start bit will be accepted no sooner than it is anticipated.

If the receiver detects that a break (RDRF=1, FE=1, receiver data register=$00) produced the framing error, the start bit will not be artificially induced and the receiver must actually receive a logic one bit before start. See Figure 5.5.



(a) Case 1, Receive Line Low During Artificial Edge



(b) Case 2, Receive Line High During Expected Start Edge

Figure 5.4 SCI Artificial Start Following A Framing Error

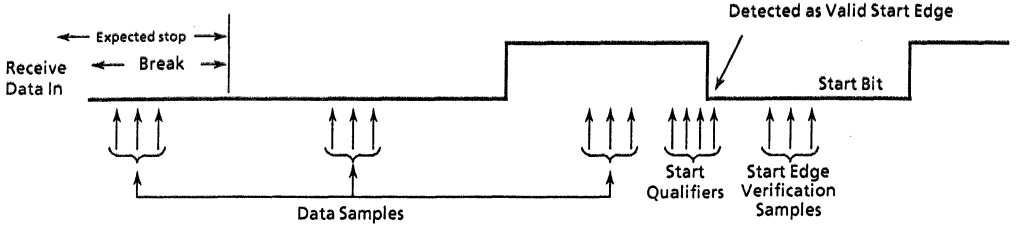


Figure 5.5 SCI Start Bit Following A Break

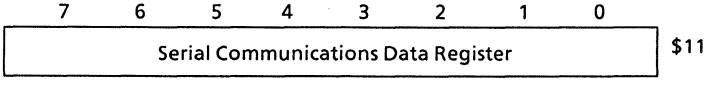
5.6 TRANSMIT DATA OUT (TDO)

Transmit data out is the serial data which is presented from the internal data bus via the SCI and then to the output pin. Data format is as discussed above and shown in Figure 5.1. The transmitter generates a bit time by using a derivative of the RT clock, thus producing a transmission rate equal to 1/16 that of the receiver sample clock.

5.7 REGISTERS

There are five different registers used in the serial communications interface (SCI) and the internal configuration of these registers is discussed in the following paragraphs. A block diagram of the SCI system is shown in Figure 5.6.

5.7.1 Serial Communications Data Register (SCDAT)



The serial communications data register (SCDAT) performs two functions in the serial communications interface;i.e. it acts as the receive data register when it is read and as the transmit data register when it is written. Figure 5.6 shows this register as two separate registers, namely:the receive data register (RDR) and the transmit data register (TDR). As shown in Figure 5.6, the TDR (transmit data register) provides the parallel interface from the internal data bus to the transmit shift register and the receive data register (RDR) provides the interface from the receive shift register to the internal data bus.

When SCDAT is read, it becomes the receive data register and contains the last byte of data received. The receive data register, represented above, is a read-only register containing the last byte of data received from the shift register for the internal data bus. The RDRF bit (receive data register full bit in the serial communications status register) is set to indicate that a byte has been transferred from the input serial shift register to the serial communications data register. The transfer is synchronized with the receiver

bit rate clock (from the receive control) as shown in Figure 5.6. All data is received least-significant-bit first.

When SCDAT is written, it becomes the transmit data register and contains the next byte of data to be transmitted. The transmit data register, also represented above, is a write-only register containing the next byte of data to be applied to the transmit shift register from the internal data bus. As long as the transmitter is enabled, data stored in the serial communications data register is transferred to the transmit shift register (after the current byte in the shift register has been transmitted). The transfer from the SCDAT to the transmit shift register is synchronized with the bit rate clock (from the transmit control) as shown in Figure 5.6. All data is transmitted least-significant-bit first.

5.7.2 Serial Communications Control Register 1 (SCCR1)

7	6	5	4	3	2	1	0	
R8	T8	–	M	WAKE	–	–	–	$0E

The serial communications control register 1 (SCCR1) provides the control bits which:1) determine the word length (either 8 or 9 bits), and 2) selects the method used for the wake-up feature. Bits 6 and 7 provide a location for storing the ninth bit for longer bytes.

B7, R8 If the M bit is a one, then this bit provides a storage location for the ninth bit in the receive data byte. Reset does not affect this bit.

B6, T8 If the M bit is a one, then this bit provides a storage location for the ninth bit in the transmit data byte. Reset does not affect this bit.

B4, M The option of the word length is selected by the configuration of this bit and is shown below. Reset does not affect this bit.

0 = 1 start bit, 8 data bits, 1 stop bit

1 = 1 start bit, 9 data bits, 1 stop bit

B3, WAKE This bit allows the user to select the method for receiver "wake up". If the WAKE bit is a logic zero, an idle line condition will "wake up" the receiver. If the WAKE bit is set to a logic one, the system acknowledges an address bit (most significant bit). The address bit is dependent on both the WAKE bit and the M bit level (table shown below). (Additionally, the receiver does not use the wake-up feature unless the RWU control bit in serial communications control register 2 is set as discussed below.) Reset does not affect this bit.

Wake	M	Method of Receiver "Wake-Up"
0	X	Detection of an idle line allows the next data byte received to cause the receive data register to fill and produce an RDRF flag.
1	0	Detection of a received one in the eighth data bit allows an RDRF flag and associated error flags.
1	1	Detection of a received one in the ninth data bit allows an RDRF flag and associated error flags.

5.7.3 Serial Communications Control Register 2 (SCCR2)

7	6	5	4	3	2	1	0	
TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK	$0F

The serial communications control register 2 (SCCR2) provides the control bits which:individually enable/disable the transmitter or receiver, enable the system interrupts, and provide the wake-up enable bit and a "send break code" bit. Each of these bits is described below. (The individual flags are discussed in the 5.7.4 Serial Communications Status Register.)

B7, TIE — When the transmit interrupt enable bit is set, the SCI interrupt occurs provided TDRE is set (see Figure 5.6). When TIE is clear, the TDRE interrupt is disabled. Reset clears the TIE bit.

B6, TCIE — When the transmission complete interrupt enable bit is set, the SCI interrupt occurs provided TC is set (see Figure 5.6). When TCIE is clear, the TC interrupt is disabled. Reset clears the TCIE bit.

B5, RIE — When the receive interrupt enable bit is set, the SCI interrupt occurs provided OR is set or RDRF is set (see Figure 5.6). When RIE is clear, the OR and RDRF interrupts are disabled. Reset clears the RIE bit.

B4,ILIE — When the idle interrupt enable bit is set, the SCI interrupt occurs provided IDLE is set (see Figure 5.6). When ILIE is clear, the IDLE interrupt is disabled. Reset clears the ILIE bit.

B3, TE — When the transmit enable bit is set, the transmit shift register output is applied to the TDO line. Depending on the state of control bit M in serial communications control register 1, a preamble of 10 (M=0) or 11 (M=1) consecutive ones is transmitted when software sets the TE bit from a cleared state. If a transmission is in progress, and TE is written to a zero, then the transmitter will wait until after the present byte has been transmitted before placing the TDO pin in the idle high-impedance state. If the TE bit has been written to a zero and then set to a one before the

current byte is transmitted, the transmitter will wait until that byte is transmitted and will then initiate transmission of a new preamble. After the preamble is transmitted, and provided the TDRE bit is set (no new data to transmit), the line remains idle (driven high while TE=1);otherwise, normal transmission occurs. This function allows the user to "neatly" terminate a transmission sequence. After loading the last byte in the serial communications data register and receiving the interrupt from TDRE, indicating the data has been transferred into the shift register, the user should clear TE. The last byte will then be transmitted and the line will go idle (high impedance). Reset clears the TE bit.

B2, RE — When the receive enable bit is set, the receiver is enabled. When RE is clear, the receiver is disabled and all of the status bits associated with the receiver (RDRF, IDLE, OR, NF, and FE) are inhibited. Reset clears the RE bit.

B1, RWU — When the receiver wake-up bit is set, it enables the "wake up" function. The type of "wake up" mode for the receiver is determined by the WAKE bit discussed above (in the SCCR1). When the RWU bit is set, no status flags will be set. Flags which were set previously will not be cleared when RWU is set. If the WAKE bit is cleared, RWU is cleared after receiving 10 (M=0) or 11 (M=1) consecutive ones. Under these conditions, RWU cannot be set if the line is idle. If the WAKE bit is set, RWU is cleared after receiving an address bit. The RDRF flag will then be set and the address byte will be stored in the receiver data register. Reset clears the RWU bit.

B0, SBK — When the send break bit is set the transmitter sends zeros in some number equal to a multiple of the data format bits. If the SBK bit is toggled set and clear, the transmitter sends 10 (M=0) or 11 (M=1) zeros and then reverts to idle or sending data. The actual number of zeros sent when SBK is toggled depends on the data format set by the M bit in the serial communications control register 1;therefore, the break code will be synchronous with respect to the data stream. At the completion of the break code, the transmitter sends at least one high bit to guarantee recognition of a valid start bit. Reset clears the SBK bit.

5.7.4 Serial Communications Status Register (SCSR)

7	6	5	4	3	2	1	0	
TDRE	TC	RDRF	IDLE	OR	NF	FE	–	$10

The serial communications status register (SCSR) provides inputs to the interrupt logic circuits for generation of the SCI system interrupt. In addition, a noise flag bit and a framing error bit are also contained in the SCSR.

B7, TDRE The transmit data register empty bit is set to indicate that the contents of the serial communications data register have been transferred to the transmit serial shift register. If the TDRE bit is clear, it indicates that the transfer has not yet occurred and a write to the serial communications data register will overwrite the previous value. The TDRE bit is cleared by accessing the serial communications status register (with TDRE set), followed by writing to the serial communications data register. Data can not be transmitted unless the serial communications status register is accessed before writing to the serial communications data register to clear the TDRE flag bit. Reset sets the TDRE bit.

B6, TC The transmit complete bit is set at the end of a data frame, preamble, or break condition if:

(1) TE=1, TDRE=1, and no pending data, preamble, or break is to be transmitted;or

(2) TE=0, and the data, preamble, or break (in the transmit shift register) has been transmitted.

The TC bit is a status flag which indicates that one of the above conditions has occurred. The TC bit is cleared by accessing the serial communications status register (with TC set), followed by writing to the serial communications data register. It does not inhibit the transmitter function in any way. Reset sets the TC bit.

B5, RDRF When the receive data register full bit is set, it indicates that the receiver serial shift register is transferred to the serial communications data register. If multiple errors are detected in any one received word, the NF, FE, and RDRF bits will be affected as appropriate during the same clock cycle. The RDRF bit is cleared when the serial communications status register is accessed (with RDRF set) followed by a read of the serial communications data register. Reset clears the RDRF bit.

B4, IDLE When the idle line detect bit is set, it indicates that a receiver idle line is detected (receipt of a minimum number of ones to constitute the number of bits in the byte format). The minimum number of ones needed will be 10 (M=0) or 11 (M=1). This allows a receiver that is not in the wake-up mode to detect the end of a message, detect the preamble of a new message, or to resynchronize with the transmitter. The IDLE bit is cleared by accessing the serial communications status register (with IDLE set)

followed by a read of the serial communications data register. The IDLE bit will not be set again until after an RDRF has been set;i.e., a new idle line occurs. The IDLE bit is not set by an idle line when the receiver "wakes up" from the wake-up mode. Reset clears the IDLE bit.

B3, OR

When the overrun error bit is set, it indicates that the next byte is ready to be transferred from the receive shift register to the serial communications data register when it is already full (RDRF bit is set). Data transfer is then inhibited until the RDRF bit is cleared. Data in the serial communications data register is valid in this case, but additional data received during an overrun condition (including the byte causing the overrun) will be lost. The OR bit is cleared when the serial communications status register is accessed (with OR set), followed by a read of the serial communications data register. Reset clears the OR bit.

B2, NF

The noise flag bit is set if there is noise on a "valid" start bit or if there is noise on any of the data bits or if there is noise on the stop bit. It is not set by noise on the idle line nor by invalid (false) start bits. If there is noise, the NF bit is not set until the RDRF flag is set. Each data bit is sampled three times as described above in RECEIVE DATA IN and shown in Figure 5.3. The NF bit represents the status of the byte in the serial communications data register. For the byte being received (shifted in)

there will also be a "working" noise flag the value of which will be transferred to the NF bit when the serial data is loaded into the serial communications data register. The NF bit does not generate an interrupt because the RDRF bit gets set with NF and can be used to generate the interrupt. The NF bits is cleared when the serial communications status register is accessed (with NF set), followed by a read of the serial communications data register. Reset clears the NF bit.

B1, FE

The framing error bit is set when the byte boundaries in the bit stream are not synchronized with the receiver bit counter (generated by a "lost" stop bit). The byte is transferred to the serial communications data register and the RDRF bit is set. The FE bit does not generate an interrupt because the RDRF bit is set at the same time as FE and can be used to generate the interrupt. Note that if the byte received causes a framing error and it will also cause an overrun if transferred to the serial communications data register, then the overrun bit will be set, but not the framing error bit, and the byte will not be transferred to the serial communications data register. The FE bit is cleared when the serial communications status register is accessed (with FE set) followed by a read of the serial communications data register. Reset clears the FE bit.

5.7.5 Baud Rate Register

7	6	5	4	3	2	1	0	
–	–	SCP1	SCP0	–	SCR2	SCR1	SCR0	$0D

The baud rate register provides the means for selecting different baud rates which may be used as the rate control for the transmitter and receiver. The SCP0-SCP1 bits function as a prescaler for the SCR0-SCR2 bits. Together, these five bits provide multiple, baud rate combinations for a given crystal frequency.

B5, SCP1
B4, SCP0

These two bits in the baud rate register are used as a prescaler to increase the range of standard baud rates controlled by the SCR0-SCR2 bits. A table of the prescaler internal processor clock division versus bit levels is provided below. Reset clears SCP1-SCP0 bits (divide-by-one).

SCP1	SCP0	Internal Processor Clock Divide By
0	0	1
0	1	3
1	0	4
1	1	13

B2, SCR2
B1, SCR1
B0, SCR0

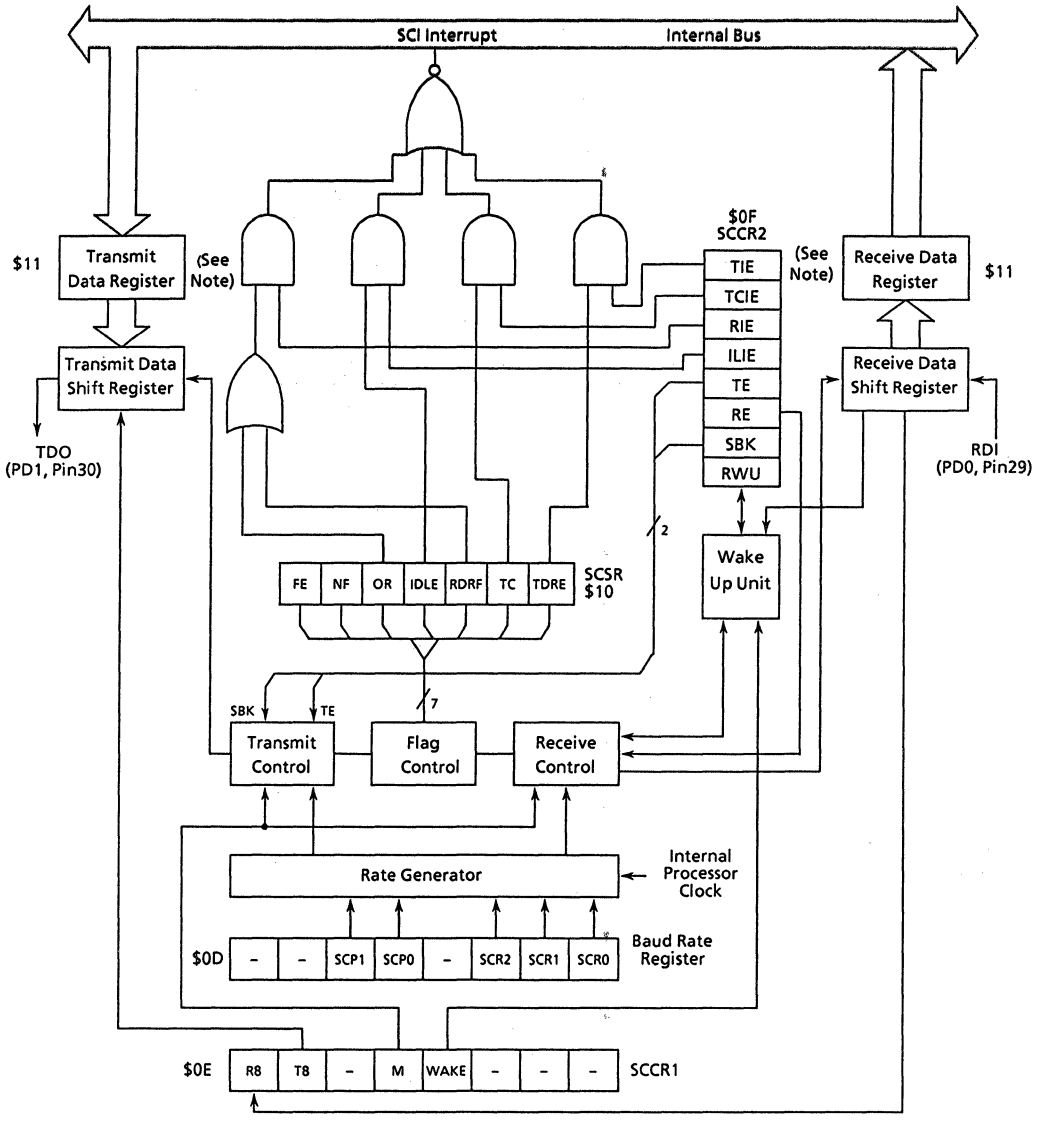
These three bits in the baud rate register are used to select the baud rates of both the transmitter and receiver. A table of baud rates versus bit levels is shown below. Reset does not affect the SCR2-SCR0 bits.

SCR2	SCR1	SCR0	Prescaler Output Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

The diagram of Figure 5.7 and Tables 5.1 and 5.2 illustrate the divided chain used to obtain the baud rate clock (transmit clock). Note that there is a fixed rate divide-by-16 between the receive clock (RT) and the transmit clock (Tx). The actual divider chain is controlled by the combined SCP0-SCP1 and SCR0-SCR2 bits in the baud rate register as

illustrated. All divided frequencies shown in the first table represent the final transmit clock (the actual baud rate) resulting from the internal processor clock division shown in the "divide-by" column only (prescaler division only). The second table illustrates how the prescaler output can be further divided by action of the SCI select bits (SCR0-SCR2). For example, assume that a 9600 Hz baud rate is required with a 2.4576 MHz external crystal. In this case the prescaler bits (SCP0-SCP1) could be configured as a divide-by-one or a divide-by-four. If a divide-by-four prescaler is used, then the SCR0-SCR2 bits must be configured as a divide-by-two. This results in a divide-by-128 of the internal processor clock to produce a 9600 Hz baud rate clock. Using the same crystal, the 9600 baud rate can be obtained with a prescaler divide-by-one and the SCR0-SCR2 bits configured for a divide-by-eight.

Note : The crystal frequency is internally divided-by-two to generate the internal processor clock.



Note: The Serial Communications Data Register (SCDAT) is controlled by the internal R/W signal. It is the transmit data register when written and receive data register when read.

Figure 5.6 Serial Communications Interface Block Diagram

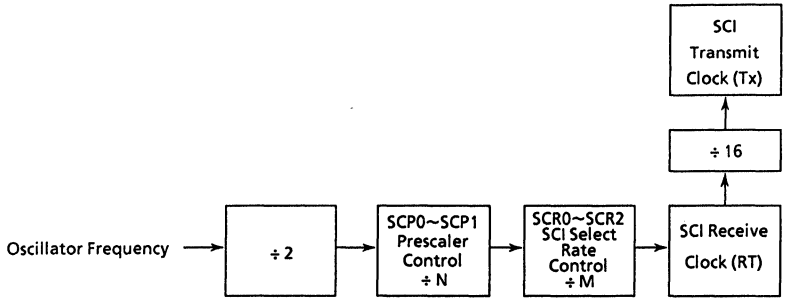


Figure 5.7 Rate Generator Division

Table 5.1 Prescaler Highest Baud Rate Frequency Output

SCP Bit		Clock* Divided By	Crystal Frequency MHz				
1	0		4.194304	4.0	2.4576	2.0	1.8432
0	0	1	131.072 KHz	125.000 KHz	76.80 KHz	62.50 KHz	57.60 KHz
0	1	3	43.691 KHz	41.666 KHz	25.60 KHz	20.833 KHz	19.20 KHz
1	0	4	32.768 KHz	31.250 KHz	19.20 KHz	15.625 KHz	14.40 KHz
1	1	13	10.082 KHz	9600 Hz	5.907 KHz	4800 Hz	4430 Hz

* The clock in the "Clock Divided By" column is the internal processor clock.

Note : The divided frequencies shown in Table 5.1 represent baud rates which are the highest transmit baud rate (Tx) that can be obtained by a specific crystal frequency and only using the prescaler division. Lower baud rates may be obtained by providing a further division using the SCI rate select bits as shown below for some representative prescaler outputs.

Table 5.2 Transmit Baud Rate Output for a Given Prescaler Output

SCR Bits			Divide By	Representative Highest Prescaler Baud Rate Output				
2	1	0		131.072 KHz	32.768 KHz	76.80 KHz	19.20 KHz	9600 KHz
0	0	0	1	131.072 KHz	32.768 KHz	76.80 KHz	19.20 KHz	9600 Hz
0	0	1	2	65.536 KHz	16.384 KHz	38.40 KHz	9600 Hz	4800 Hz
0	1	0	4	32.768 KHz	8.192 KHz	19.20 KHz	4800 Hz	2400 Hz
0	1	1	8	16.384 KHz	4.096 KHz	9600 Hz	2400 Hz	1200 Hz
1	0	0	16	8.192 KHz	2.048 KHz	4800 Hz	1200 Hz	600 Hz
1	0	1	32	4.096 KHz	1.024 KHz	2400 Hz	600 Hz	300 Hz
1	1	0	64	2.048 KHz	512 Hz	1200 Hz	300 Hz	150 Hz
1	1	1	128	1.024 KHz	256 Hz	600 Hz	150 Hz	75 Hz

Note : Table 5.2 illustrates how the SCI select bits can be used to provide lower transmitter baud rates by further dividing the prescaler output frequency. The five examples are only representative samples. In all cases, the baud rates shown are transmit baud rates (trasmit clock) and the receiver clock is 16 times higher in frequency than the actual baud rate.

6. SERIAL PERIPHERAL INTERFACE (SPI)

6.1 INTRODUCTION AND FEATURES

6.1.1 Introduction

The serial peripheral interface (SPI) is an interface built into the TMP68HC05C4 MCU which allows several TMP68HC05C4 MCUs, or TMP68HC05C4 plus peripheral devices, to be interconnected within a single "black box" or on the same printed circuit board. In a serial peripheral interface (SPI), 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 in one containing one master MCU and several slave MCUs, or in a system in which an MCU is capable of being either a master or a slave.

Figure 6.1 illustrates two different system configurations. Figure 6.1a represents a system of five different MCUs in which there are 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 6.1b represents a system of five MCUs in which three can be master or slave and two are slave only.

6.1.2 Features

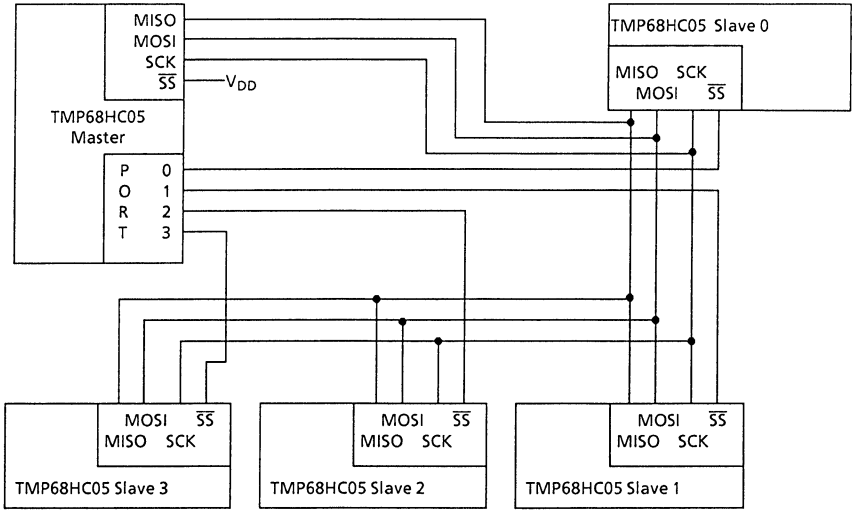
- Full duplex, three-wire synchronous transfers
- Master or slave operation
- 1.05 MHz (maximum) master bit frequency
- 2.1 MHz (maximum) slave bit frequency
- Four programmable master bit rates
- Programmable clock polarity and phase
- End of transmission interrupt flag
- Write collision flag protection
- Master-Master mode fault protection capability

6.2 SIGNAL DESCRIPTION

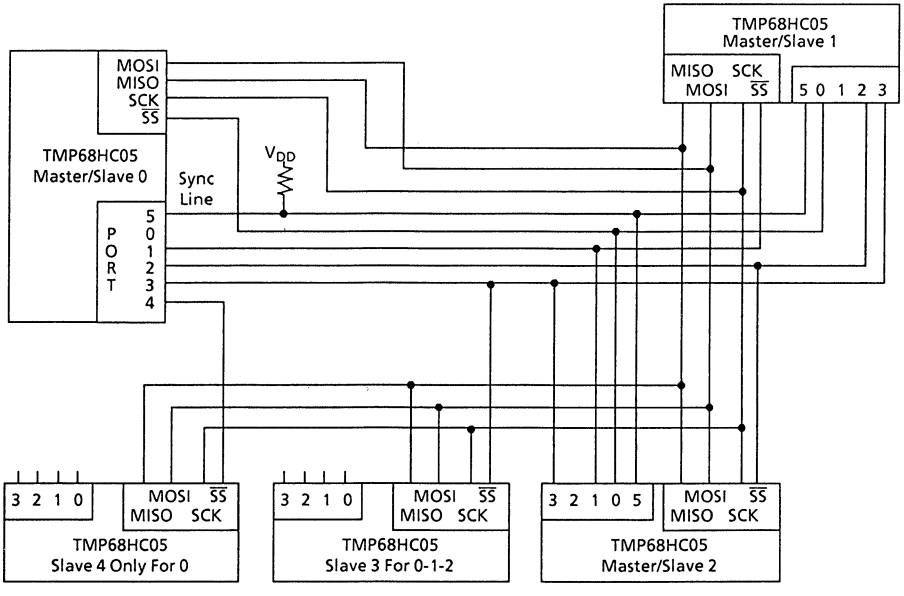
The four basic signals (MOSI, MISO, SCK, and $\overline{SS}$) discussed above are described in the following paragraphs. Each signal function is described for both the master and slave mode.

6.2.1 Master Out Slave In (MOSI)

The MOSI pin is configured as a data output in a master (mode) device and as a data input in a slave (mode) device. In this manner data is transferred serially from a master to a slave on this line;most



a. Single Master, Four Slaves



b. Three Master/Slave, Two Slaves

Figure 6.1 Master-Slave System Configuration

significant bit first, least significant bit last. The timing diagrams of Figure 6.2 summarize the SPI timing diagram shown in Section 9, and show the relationship between data and clock (SCK). As shown in Figure 6.2, 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 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.

When the master device transmits data to a second (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) 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.

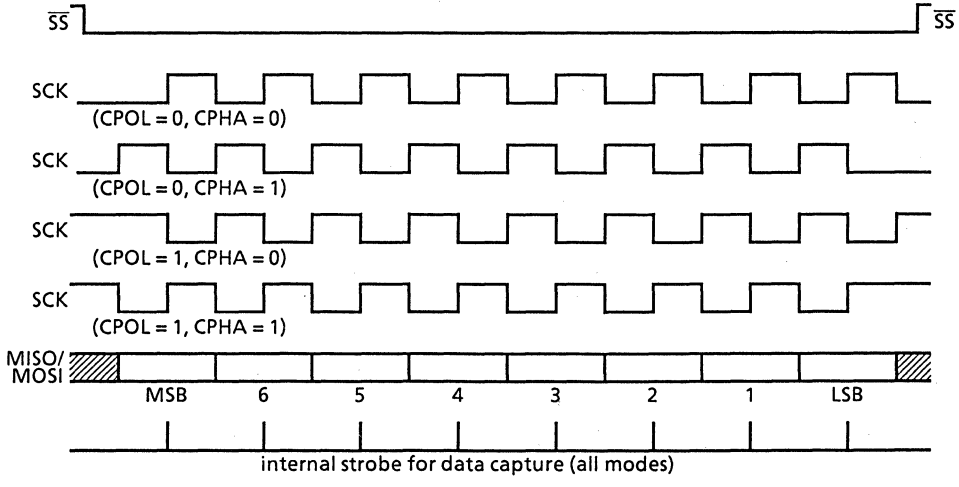


Figure 6.2 Data Clock Timing Diagram

6.2.2 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. In this manner data is transferred serially from a slave to a master on this line;most significant bit first, least significant bit last. 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 of Figure 6.2 shows the relationship between data and clock (SCK). As shown in Figure 6.2, 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 clock edge (SCK) in order for the slave device to latch the data.

Note : 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.

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.

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

The slave select ($\overline{SS}$) pin is a fixed input (PD5, pin 34), 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 SCK (system clock) and must remain low until after the last (eighth) SCK cycle. Figure 6.2 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 of 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 (master in slave out) line to the high-impedance state. Also, SCK and the MOSI (master out slave in) 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 which causes the serial peripheral interface (SPI) to be disabled (port D SPI pins become inputs). 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.

6.2.4 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 the 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) discussed below. Refer to Figure 6.2 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 6.2.

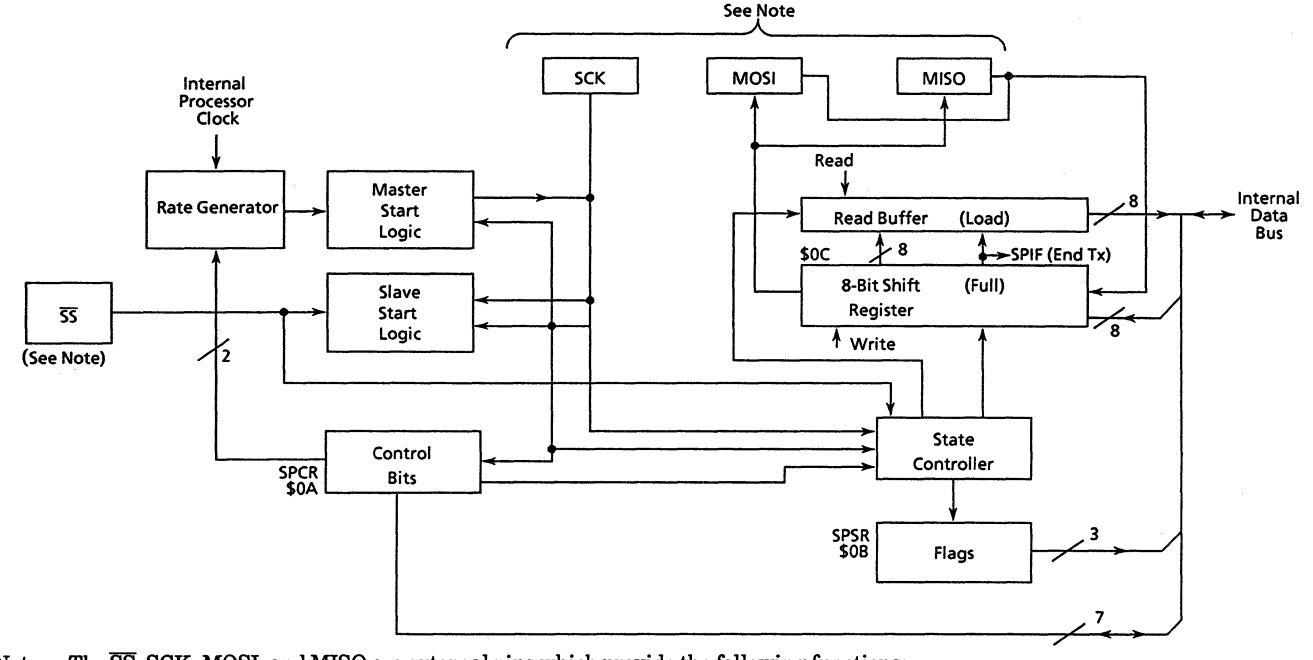
6.3 FUNCTIONAL DESCRIPTION

A block diagram of the serial peripheral interface (SPI) is shown in Figure 6.3. 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 system 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 system 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 register from the internal data bus and then shifted out serially to the MISO pin for application to the master device.

Figure 6.4 illustrates the MOSI, MISO, and SCK master-slave interconnections. Note that in Figure 6.4 the master $\overline{SS}$ pin is tied to a logic high and the slave $\overline{SS}$ pin is a logic low. Figure 6.1 provides a larger system connection for these same pins. Note that in Figure 6.1, 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.



Note : 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 6.3 Serial Peripheral Interface Block Diagram

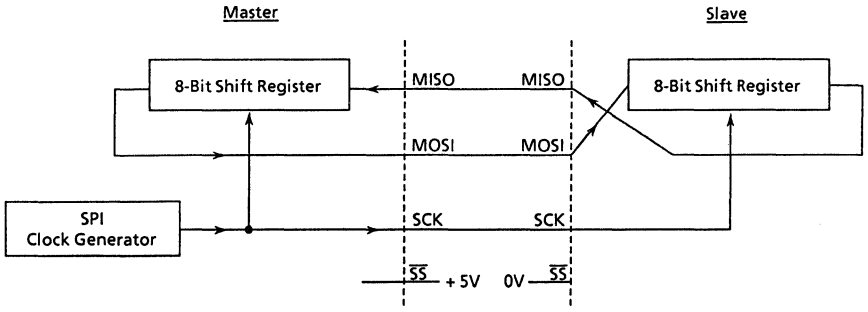


Figure 6.4 Serial Peripheral Interface Master-Slave Interconnection

6.4 REGISTERS

There are three registers in the serial parallel interface which provide control, status, and data storage functions. These registers which include 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 below.

6.4.1 Serial Peripheral Control Register (SPCR)

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

The serial peripheral control register bits are defined as follows:

B7, SPIE — 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.

B6, SPE — 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 SPE bit is cleared by reset, the SPI system is not connected to the external pins upon reset.

B4, MSTR — 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 MISO

and MOSI pins 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.

B3, CPOL — 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 6.2.

B2, CPHA — 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 6.2.

B1, SPR1
B0, SPR0 — 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

6.4.2 Serial Peripheral Status Register (SPSR)

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

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:

B7, SPIF 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 (SPI) 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 receive buffer and other bytes are losat.

The transfer 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.

B6, WCOL 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 a 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 msb onto the external MISO pin of the slave device. The $\overline{SS}$ pin low state enables the slave device but the drive onto the MISO 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.

A special case of WCOL 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 WCOL 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 cleaed by reset.

B4, MODF The function of the mode fault flag 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:

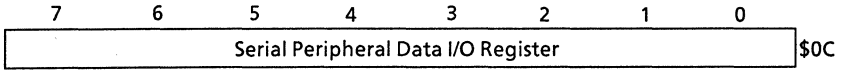
(1) MODF is set and SPI interrupt is generated if SPIE=1.

(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 cleated by reset.

6.4.3 Serial Peripheral Data I/O Register (SPDR)



The serial peripheral data I/O register (SPDR) 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 its 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 SPIF must 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.

6.5 SERIAL PERIPHERAL INTERFACE (SPI) SYSTEM CONSIDERATIONS

There are two types of SPI systems;single master system and multi-master systems. Figure 6.1 illustrates both of these systems and a discussion of each is provided below.

Figure 6.1 a illustrates how a typical single master system may be configured, using an M6805 HCMOS family device as the master and four M6805 HCMOS family devices as slaves. As shown, the MOSI, MISO, and SCK pins are all wired to equivalent pins on each of the five devices. The master device generates the SCK clock, the slave devices all receive it. 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 6.1b. 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.

7. EFFECTS OF STOP AND WAIT MODES ON THE TIMER AND SERIAL SYSTEMS

7.1 INTRODUCTION

The STOP and WAIT instructions have different effects on the programmable timer, serial communications interface (SCI), and serial peripheral interface (SPI) systems. These different effects are discussed separately below.

7.2 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 programmable timer, serial communications interface, and 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). The effects of the stop mode on each of the MCU systems (Timer, SCI, and SPI) are described separately.

7.2.1 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.

7.2.2 SCI During Stop Mode

When the MCU enters the stop mode, the baud rate generator which derives the receiver and transmitter is shut down. This essentially stops all SCI activity. The receiver is unable to receive and transmitter is unable to transmit. If the STOP instruction is executed during a transmitter transfer, that transfer is halted. When the stop mode is exited, that particular transmission resumes (if the exit is the result of a low input to the $\overline{IRQ}$ pin). Since the previous transmission resumes after an $\overline{IRQ}$ interrupt stop mode exit, the user should ensure that the SCI transmitter is in the idle state when the STOP instruction is executed. If the receiver is receiving data when the STOP instruction is executed, received data sampling is stopped (baud rate generator

stops) and the rest of the data is lost. For the above reasons, all SCI transactions should be in the idle state when the STOP instruction is executed.

7.2.3 SPI During Stop Mode

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.

At the end of a possible transmission with a slave SPI in the stop mode, no flags are set until a logic low $\overline{IRQ}$ input results in an MCU "wake up". Caution should be observed when operating the SPI (as a slave) during the stop mode because none of the protection circuity (write collision, mode fault, etc.) is active.

It should also be noted that when the MCU enters the stop mode all enabled output drivers (TDO, TCMP, MISO, MOSI, and SCK ports) remain active and any sourcing currents from these outputs will be part of the total supply current required by the device.

7.3 WAIT MODE

When the MCU enters the wait mode, the CPU clock is halted. All CPU action is suspended;however, the timer, SCI, and SPI systems remain active. In fact an interrupt from the timer, SCI, or SPI (in addition to a logic low on the $\overline{IRQ}$ or $\overline{RESET}$ pins) causes the processor to exit the wait mode. Since the three systems mentioned above operate as they do in the normal mode, only a general discussion of the wait mode is provided below.

The wait mode power consumption depends on how many systems are active. The power consumption will be highest when all the systems (timer, TCMP, SCI, and SPI) are active. The power consumption will be the least when the SCI and SPI systems are disabled (timer operation cannot be disabled in the wait mode). If a non-reset exit from the wait mode is performed (e.g., timer overflow interrupt exit), the state of the remaining systems will be unchanged. If a reset exit from the wait mode is performed all the systems revert to the disabled reset state.

8. INSTRUCTION SET AND ADDRESSING MODES

8.1 INSTRUCTION SET

The MCU 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.

All of the instructions used in the M146805 CMOS Family are used in the TMP 68HC05C4MCU, 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 high 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

8.1.1 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 8.1.

8.1.2 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 8.2.

Table 8.1 Register/Memory Instructions

Function	Mnem.	Addressing Modes: Immediate Op Code	Immediate # Bytes	Immediate # Cycles	Direct Op Code	Direct # Bytes	Direct # Cycles	Extended Op Code	Extended # Bytes	Extended # Cycles	Indexed (No Offset) Op Code	Indexed (No Offset) # Bytes	Indexed (No Offset) # Cycles	Indexed (8-Bit Offset) Op Code	Indexed (8-Bit Offset) # Bytes	Indexed (8-Bit Offset) # Cycles	Indexed (16-Bit Offset) Op Code	Indexed (16-Bit Offset) # Bytes	Indexed (16-Bit Offset) # 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 8.2 Read-Modify-Write Instructions

Function	Mnemonic	Addressing Modes: Inherent (A) Op Code	# Bytes	# Cycles	Inherent (X) Op Code	# Bytes	# Cycles	Direct Op Code	# Bytes	# Cycles	Indexed (No Offset) Op Code	# Bytes	# Cycles	Indexed (8-Bit Offset) 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
Nagate (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	–	–	–	–	–	–	–	–	–	–	–	–

8.1.3 Branch Instructions

Most branch instruction 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 8.3.

Table 8.3 Branch Instructions

Function	Reiative Addressing Mode			
	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

8.1.4 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), serial communications status register ($10), timer status register ($13), and timer input capture register ($14-$15). All port register, port DDRs, timer, two serial systems, on-chip RAM, and 48 bytes of 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 8.4.

Table 8.4 Bit Manipulation Instructions

Function	Mnemonic	Addressing Modes					
		Bit Set/Clear			Bit Test and Branch		
		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	–	–	–

8.1.5 Control Instructions

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to Table 8.5.

Table 8.5 Control Instructions

Function	Mnemonic	Inherent		
		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

8.1.6 Alphabetical Listing

The complete instruction set is given in alphabetical order in Table 8.6.

8.1.7 Opcode Map

Table 8.7 is an opcode map for the instructions used on the MCU.

8.2 ADDRESSING MODES

The MCU 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 8.7 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 below. 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. For additional details and graphical illustrations, refer to the M6805 HMOS Family Microcomputer/ Microprocessor User's Manual.

Table 8.6 Instruction Set (1/2)

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			●	●	●	●	●

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 8.6 Instruction Set (2/2)

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
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:

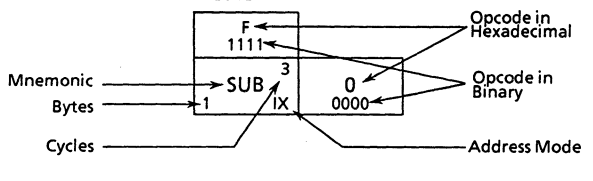
H	Half Carry (From Bit 3)	Λ	Test and Set if True Cleared Otherwise
I	Interrupt Mask	●	Not Affected
N	Negate (Sign Bit)	?	Load CC Register From Stack
Z	Zero	0	Cleared
C	Carry/Borrow	1	Set

Table 8.7 MC68HC05C4 HCMOS 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 DTR	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
F 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



8.2.1 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.

8.2.2 Immediate

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).

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

8.2.3 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.

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

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

8.2.4 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.

$$EA = (PC + 1):(PC + 2); PC \leftarrow PC + 3$$

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

8.2.5 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.

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

$$\text{Address Bus High} \leftarrow 0; \text{Address Bus Low} \leftarrow X$$

8.2.6 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 an 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 + 1)

8.2.7 Indexed, 16-Bit Offset

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)

8.2.8 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 within the span of the branch.

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

$$\text{otherwise, } EA = PC \leftarrow PC + 2$$

8.2.9 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.

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

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

8.2.10 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.

$$EA1=(PC+1)$$

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

$$EA2=PC+3+(PC+2);PC\leftarrow EA2 \text{ if branch taken;}$$

$$\text{otherwise, } PC\leftarrow PC+3$$

9. ELECTRICAL SPECIFICATIONS

9.1 INTRODUCTION

This section contains the electrical specifications and associated timing information for the TMP68HC05C4.

9.2 MAXIMUM RATINGS

(Voltages Referenced to V_{SS})

Ratings	Symbol	Value	Unit
Supply Voltage	V_{DD}	− 0.3 to + 7.0	V
Input Voltage	Vin	V_{SS} − 0.3 to V_{DD} + 0.3	V
Self-Check Mode ($\overline{IRQ}$ Pin Only)	Vin	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 TMP68HC05C4 (Standard)	T_A	T_L to T_H 0 to + 70	℃
Storage Temperature Range	T_{stg}	− 65 to + 150	℃

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 $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{DD}$. Reliability of operation is enhanced if unused inputs are connected to an appropriate logic voltage level (e.g., either V_{SS} or V_{DD}).

9.3 THERMAL CHARACTERISTICS

Characteristics	Symbol	Value	Unit
Thermal Resistance	θ_{JA}		℃/W
Ceramic		50	
Plastic		60	
Plastic Leaded Chip Carrier (PLCC)		70	

V_{DD} = 4.5V

Pins	R1	R2	C
PA0-PA7, PB0-PB7, PC0-PC7, PD6	3.26kΩ	2.38kΩ	50pF
PD1-PD4	1.9kΩ	2.26kΩ	200pF

V_{DD} = 3.0V

Pins	R1	R2	C
PA0-PA7, PB0-PB7, PC0-PC7, PD6	10.91kΩ	6.32kΩ	50pF
PD1-PD4	6kΩ	6kΩ	200pF

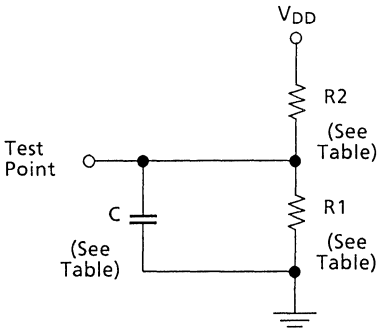


Figure 9.1 Equivalent Test Load

9.4 POWER CONSIDERATIONS

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

$P_D \equiv P_{INT} + P_{I/O}$

$P_{INT} \equiv I_{CC} \times V_{CC}$, Watts – Chip Internal Power

$P_{I/O}$ ≡ Power Dissipation on Input and Output Pins – User Determined

For most applications $P_{I/O} < P_{INT}$ and can be neglected. P_{PORT} may be significant if the device is configured to drive Darlington bases or sink LED loads.

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.

9.5 DC ELECTRICAL CHARACTERISTICS

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

Characteristic	Sym-bol	Min	Typ	Max	Unit
Output Voltage, I_{Load} = ≤10.0μA	V_{OL} V_{OH}	– V_{DD} – 0.1	– –	0.1 –	V V
Output High Voltage (I_{Load} = 0.8mA) PA0-PA7, PB0-PB7, PC0-PC7, TCMP (See Figure 9.2) (I_{Load} = 1.6mA) PD1-PD4 (See Figure 9.3)	V_{OH} V_{OH}	V_{DD} – 0.8 V_{DD} – 0.8	– –	– –	V V
Output Low Voltage (See Figure 9.4) (I_{Load} = 1.6mA) PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4, TCMP	V_{OL}	–	–	0.4	V
Input High Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IH}	0.7 × V_{DD}	–	V_{DD}	V
Input Low Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, TCAP, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IL}	V_{SS}	–	0.2 × V_{DD}	V
Data Retention Mode (0° to 70°C)	V_{RM}	2.0	–	–	V
Supply Current (See Notes) Run (See Figures 9.5 and 9.6) Wait (See Figures 9.5 and 9.6) Stop (See Figure 9.6) 25°C 0° to 70°C	 I_{DD} I_{DD} I_{DD} I_{DD}	 – – – –	 3.5 1.6 2.0 –	 7.0 4.0 50 140	 mA mA μA μA
I/O Ports Hi-Z Leakage Current PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4	I_{IL}	–	–	± 10	μA
Input Current $\overline{RESET}$, $\overline{IRQ}$, TCAP, OSC1, PD0, PD5, PD7	I_{in}	–	–	± 1	μA
Capacitance Ports (as Input or Output) $\overline{RESET}$, $\overline{IRQ}$, TCAP, OSC1, PD0-PD5, PD7	 C_{out} C_{in}	 – –	 – –	 12 8	 PF PF

Notes :

1. All values shown reflect average measurements.
2. Typical values at midpoint of voltage range, 25°C only.
3. Wait I_{DD}:Only timer system active (SPE = TE = RE = 0). If SPI, SCI active (SPE = TE = RE = 1) add 10% current draw.
4. Run (Operating) I_{DD}, Wait I_{DD}:Measured using external square wave clock source (fosc = 4.2 MHz), all inputs 0.2 V from rail, no DC loads, less than 50 pF on all outputs, C_L = 20 pF on OSC2.
5. Wait, 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. Wait I_{DD} is affected linearly by the OSC2 capacitance.

9.6 DC ELECTRICAL CHARACTERISTICS

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

Characteristic	Symbol	Min	Typ	Max	Unit
Output Voltage, $I_{Load} \leq 10.0\mu A$	V_{OL} V_{OH}	– $V_{DD}-0.1$	– –	0.1 –	V V
Output High Voltage (I_{Load} = 0.2mA) PA0-PA7, PB-PB7, PC0-PC7, TCMP (See Figure 9.2) (I_{Load} = 0.4mA) PD1-PD4 (See Figure 9.3)	V_{OH} V_{OH}	$V_{DD}-0.3$ $V_{DD}-0.3$	– –	– –	V V
Output Low Voltage (See Figure 9.4) (I_{Load} = 0.4mA) PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4, TCMP	V_{OL}	–	–	0.3	V
Input High Voltage PA0-PA7, PB0-PB7, PC0-PC7, PD0-PD5, PD7, 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-PD5, PD7, 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) Run (See Figures 9.5 and 9.7) Wait (See Figures 9.5 snd 9.7) Stop (See Figure 9.7) 25°C 0° to 70°C	 I_{DD} I_{DD} I_{DD} I_{DD}	 – – – –	 1.0 0.5 1.0 –	 2.5 1.4 30 80	 mA mA μA μA
I/O Ports Hi-Z Leakage Current PA0-PA7, PB0-PB7, PC0-PC7, PD1-PD4	I_{IL}	–	–	± 10	μA
Input Current $\overline{RESET}$, $\overline{IRQ}$, TCAP, OSC1, PD0, PD5, PD7	I_{in}	–	–	± 1	μA
Capacitance Ports (as Input or Output) $\overline{RESET}$, $\overline{IRQ}$, TCAP, OSC1, PD0-PD5, PD7	 C_{out} C_{in}	 – –	 – –	 12 8	 PF PF

Notes :

1. All values shown reflect average measurements.
2. Typical values at midpoint of voltage range, 25°C only.
3. Wait I_{DD}:Only timer system active (SPE = TE = RE = 0). If SPI, SCI active (SPE = TE = RE = 1) add 10% current draw.
4. Run (Operating) I_{DD}, Wait I_{DD}: Measured using external square wave clock source (fosc = 2.0 MHz), all input 0.2 V from rail, no DC loads, less than 50 pF on all outputs, C_L = 20 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. Wait I_{DD} is affected linearly by the OSC2 capacitance.

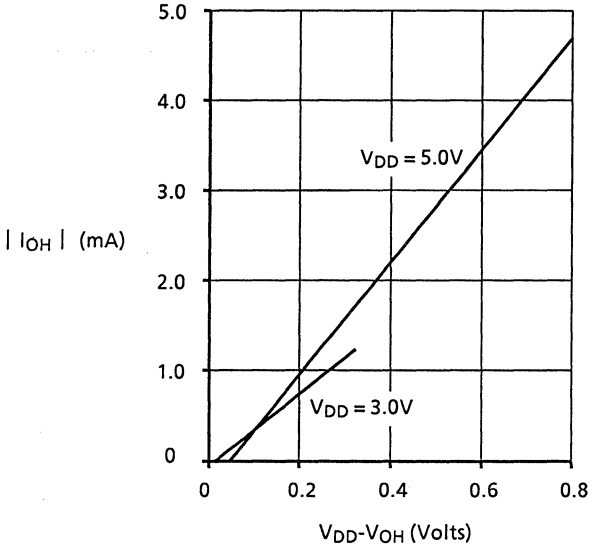


Figure 9.2 Typical V_{OH} vs I_{OH} for Ports A, B, C, and TCMP

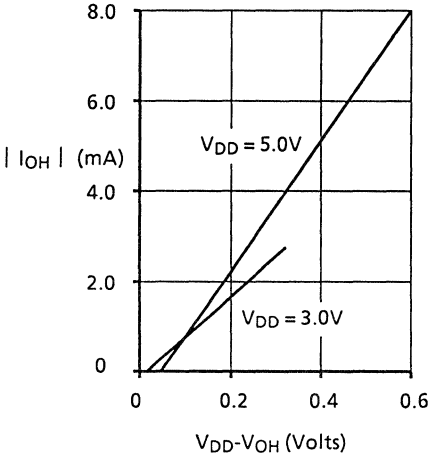


Figure 9.3 Typical V_{OH} vs I_{OH} for PD1-PD4

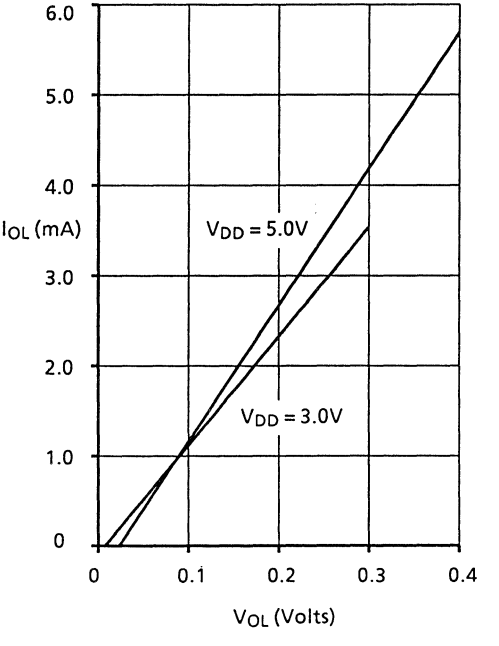


Figure 9.4 Typical V_{OL} vs I_{OL} for all Ports

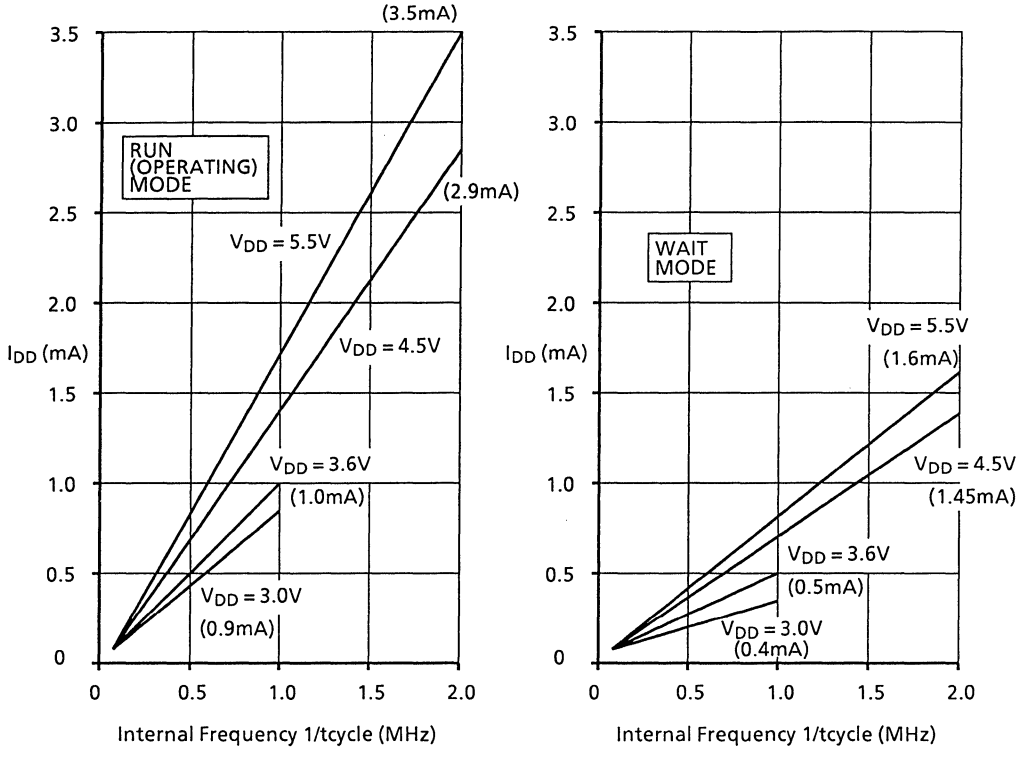


Figure 9.5 Typical Current vs Internal Frequency for Run and Wait Modes

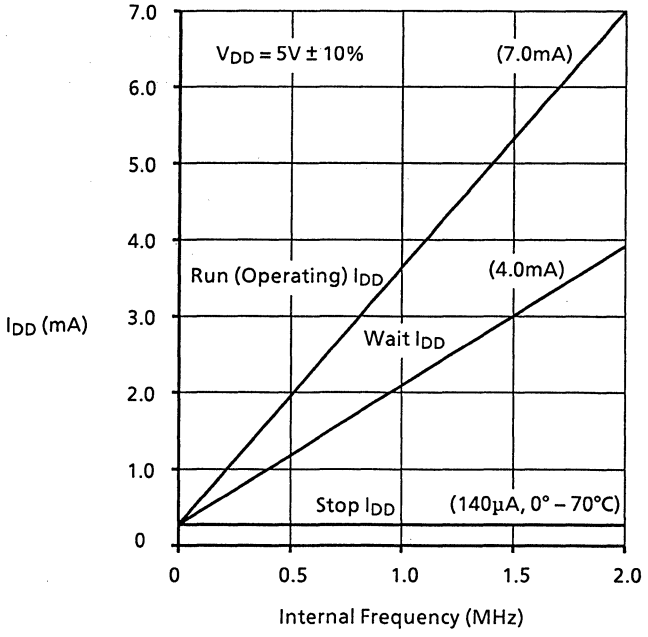


Figure 9.6 Maximum I_{DD} vs Frequency for V_{DD} = 5.0Vdc

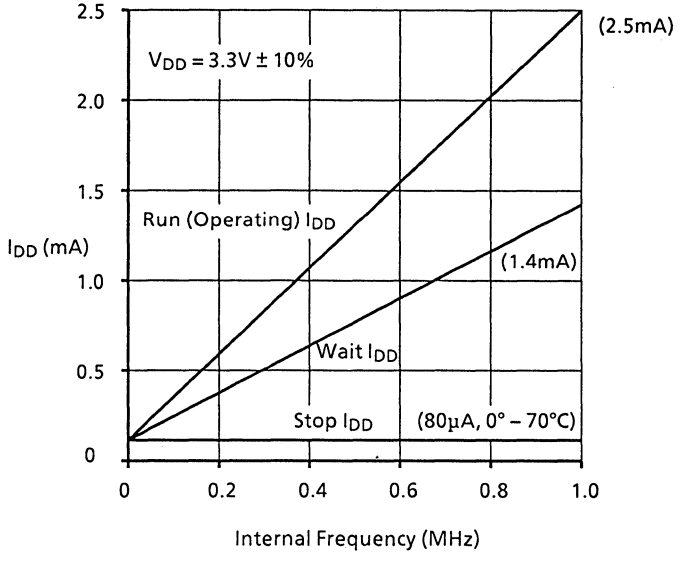


Figure 9.7 Maximum I_{DD} vs Frequency for V_{DD} = 3.3Vdc

9.7 CONTROL TIMING

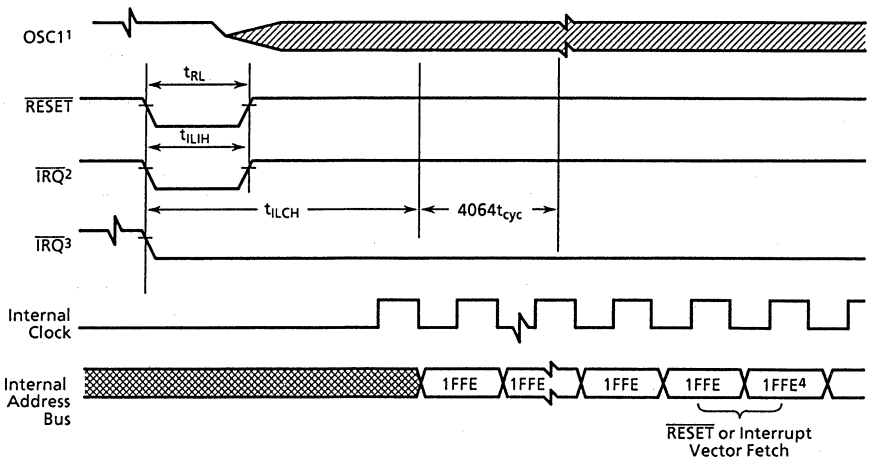
(V_{DD}=5.0V±10%, V_{SS}=0V, T_A=T_L to T_H)

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation Crystal Option External Clock Option	 f_{osc} f_{osc}	 – dc	 4.2 4.2	 MHz MHz
Internal Operating Frequency Crystal (f_{osc} ÷ 2) External Clock (f_{osc} ÷ 2)	 f_{op} f_{op}	 – dc	 2.1 2.1	 MHz MHz
Cycle Time (See Figure 3.1)	t_{cyc}	480	–	ns
Crystal Oscillator Startup Time (See Figure 3.1)	t_{OXOV}	–	100	ms
Stop Recovery Startup Time (Crystal Oscillator) (See Figure 9.8)	t_{ILCH}	–	100	ms
$\overline{RESET}$ Pulse Width (See Figure 3.1)	t_{RL}	1.5	–	t_{cyc}
Timer Resolution** Input Capture Pulse Width (See Figure 9.9) Input Capture Pulse Period (See Figure 9.9)	 t_{RESL} t_{TH}, t_{TL} t_{TLTL}	 4.0 125 ***	 – – –	 t_{cyc} ns t_{cyc}
Interrupt Pulse Width Low (Edge-Triggered) (See Figure 3.4)	t_{ILIH}	125	–	ns
Interrupt Pulse Period (See Figure 3.4)	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 2-bit prescaler in the timer must count four internal cycles (t_{cyc}), 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. $\overline{\text{IRQ}}$ pin edge-sensitive mask option.
3. $\overline{\text{IRQ}}$ pin level and edge-sensitive mask option.
4. $\overline{\text{RESET}}$ vector address shown for timing example.

Figure 9.8 Stop Recovery Timing Diagram

9.8 CONTROL TIMING

($V_{DD}=3.3V\pm10\%$, $V_{SS}=0V$, $T_A=T_L$ to T_H)

Characteristic	Symbol	Min	Max	Unit
Frequency of Operation Crystal Option External Clock Option	 f_{osc} f_{osc}	 – dc	 2.0 2.0	 MHz MHz
Internal Operating Frequency Crystal ($f_{osc}\div2$) External Clock ($f_{osc}\div2$)	 f_{op} f_{op}	 – dc	 1.0 1.0	 MHz MHz
Cycle Time (See Figure 3.1)	t_{cyc}	1000	–	ns
Crystal Oscillator Startup Time (See Figure 3.1)	t_{OXOV}	–	100	ms
Stop Recovery Startup Time (Crystal Oscillator) (See Figure 9.8)	t_{ILCH}	–	100	ms
$\overline{\text{RESET}}$ Pulse Width-Excluding Power-Up (See Figure 3.1)	t_{RL}	1.5	–	t_{cyc}
Timer Resolution** Input Capture Pulse Width (See Figure 9.9) Input Capture Pulse Period (See Figure 9.9)	 t_{RESL} t_{TH}, t_{TL} t_{TLTL}	 4.0 250 ***	 – – –	 t_{cyc} ns t_{cyc}
Interrupt Pulse Width Low (Edge-Triggered) (See Figure 3.4)	t_{ILIH}	250	–	ns
Interrupt Pulse Period (See Figure 3.4)	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 2-bit prescaler in the timer must count four internal cycles (t_{cyc}), 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}.

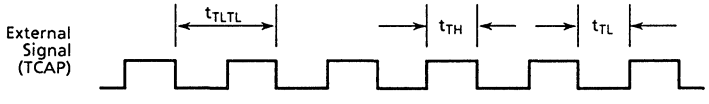


Figure 9.9 Timer Relationships

9.9 SERIAL PERIPHERAL INTERFACE (SPI) TIMING

(V_{DD}=5.0V±10%, V_{SS}=0V, 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 ns
5	Clock (SCK) Low Time Master Slave	 $t_{W\,(SCKL)\,m}$ $t_{W\,(SCKL)\,s}$	 340 190	 – –	 ns ns
6	Data Setup Time (Inputs) Master Slave	 $t_{su\,(m)}$ $t_{su\,(s)}$	 100 100	 – –	 ns ns
7	Data Hold Time (Inputs) Master Slave	 $t_{h\,(m)}$ $t_{h\,(s)}$	 100 100	 – –	 ns ns
8	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 = 200pF) SPI Outputs (SCK, MOSI, MISO) SPI Inputs (SCK, MOSI, MISO, $\overline{SS}$)	 t_{rm} t_{rs}	 – –	 100 2.0	 ns µs
13	Fall Time (70% V_{DD} to 20% V_{DD}, C_L = 200pF) SPI Outputs (SCK, MOSI, MISO) SPI Inputs (SCK, MOSI, MISO, $\overline{SS}$)	 t_{fm} t_{fs}	 – –	 100 2.0	 ns µs

* Signal production depends on software.

** Assumes 200 pF load on all SPI pins.

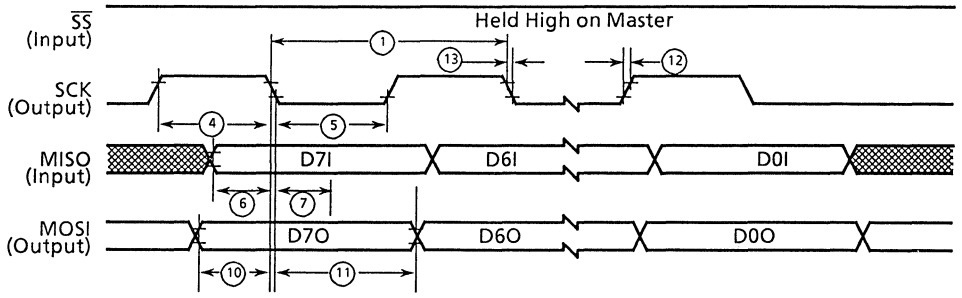
9.10 SERIAL PERIPHERAL INTERFACE (SPI) TIMING

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

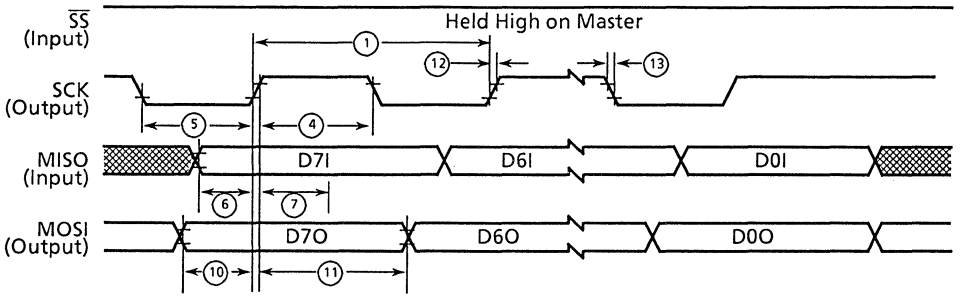
Num.	Characteristic	Symbol	Min	Max	Unit
	Operating Frequency Master Slave	 $f_{op\,(m)}$ $f_{op\,(s)}$	 dc dc	 0.5 1.0	 f_{op} MHz
1	Cycle Time Master Slave	 $t_{cyc\,(m)}$ $t_{cyc\,(s)}$	 2.0 1.0	 – –	 t_{cyc} μs
2	Enable Lead Time Master Slave	 $t_{lead\,(m)}$ $t_{lead\,(s)}$	 * 500	 – –	ns
3	Enable 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 ns
7	Data Hold Time (Inputs) Master Slave	 $t_{h\,(m)}$ $t_{h\,(s)}$	 200 200	 – –	 ns 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 = 200pF) SPI Outputs (SCK, MOSI, MISO) SPI Inputs (SCK, MOSI, MISO, $\overline{SS}$)	 t_{rm} t_{rs}	 – –	 200 2.0	 ns μs
13	Fall Time (70% V_{DD} to 20% V_{DD}, C_L = 200pF) SPI Outputs (SCK, MOSI, MISO) SPI Inputs (SCK, MOSI, MISO, $\overline{SS}$)	 t_{fm} t_{fs}	 – –	 200 2.0	 ns μs

* Signal production depends on software.

** Assumes 200 pF load on all SPI pins.

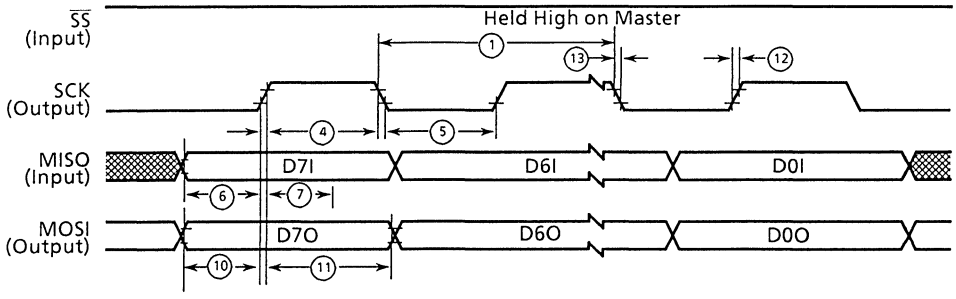


(a) SPI Master Timing CPOL = 0, CPHA = 1

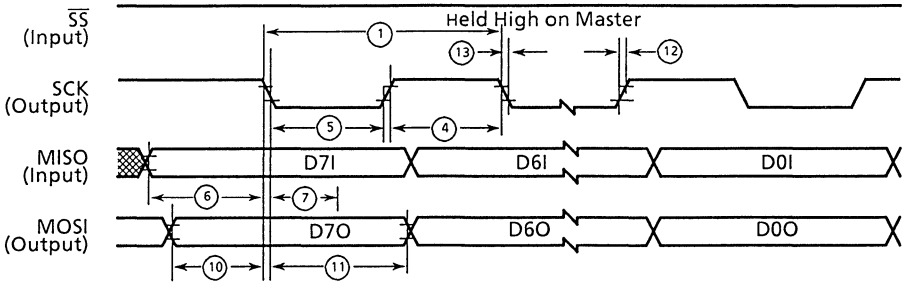


(b) SPI Master Timing CPOL = 1, CPHA = 1

Figure 9.10 SPI Timing Diagrams
(Sheet1 of 4)



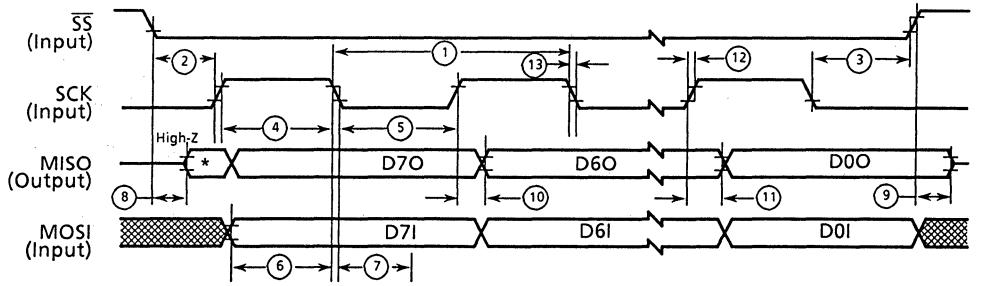
(C) SPI Master Timing CPOL = 0, CPHA = 0



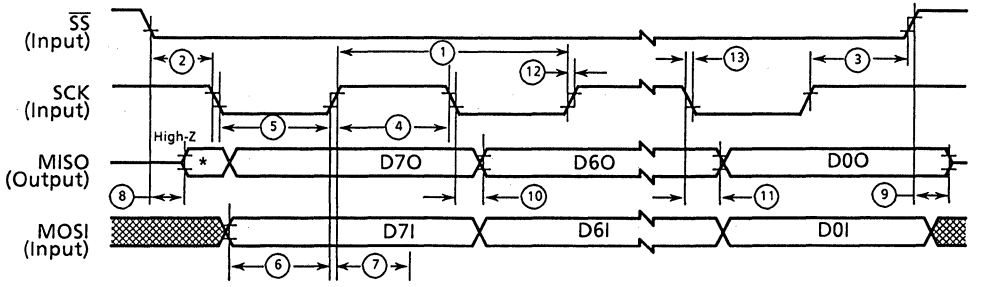
(d) SPI Master Timing CPOL = 1, CPHA = 0

Note : Measurement points are V_{OL}, V_{OH}, V_{IL}, and V_{IH}

Figure 9.10 SPI Timing Diagrams
(Sheet2 of 4)

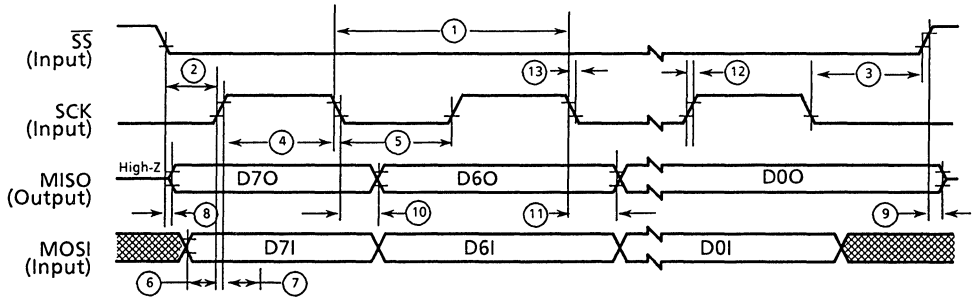


(e) SPI Slave Timing CPOL = 0, CPHA = 1

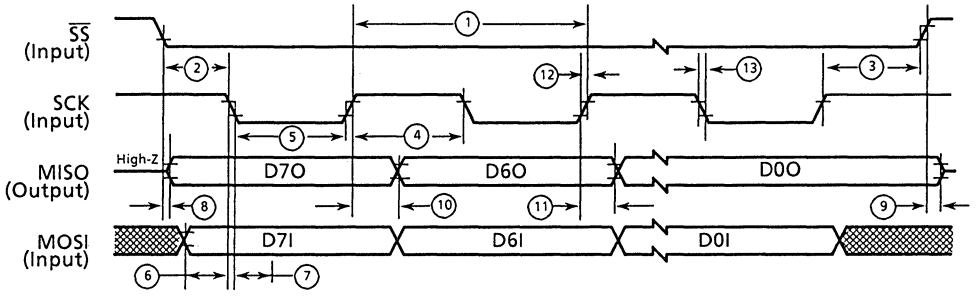


(f) SPI Slave Timing CPOL = 1, CPHA = 1

Figure 9.10 SPI Timing Diagrams
(Sheet3 of 4)



(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.

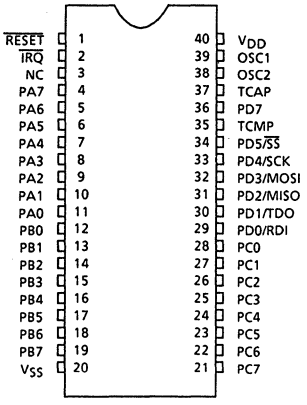
Figure 9.10 SPI Timing Diagrams
(Sheet4 of 4)

10. MECHANICAL DATA

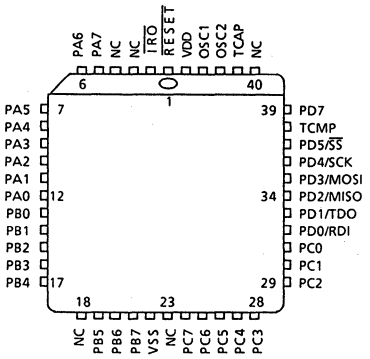
This section contains the pin assignment and package dimension diagrams for the TMP68HC05C4 microcomputer.

10.1 PIN ASSIGNMENT

P SUFFIX
40-Pin Dual-in-Line Package

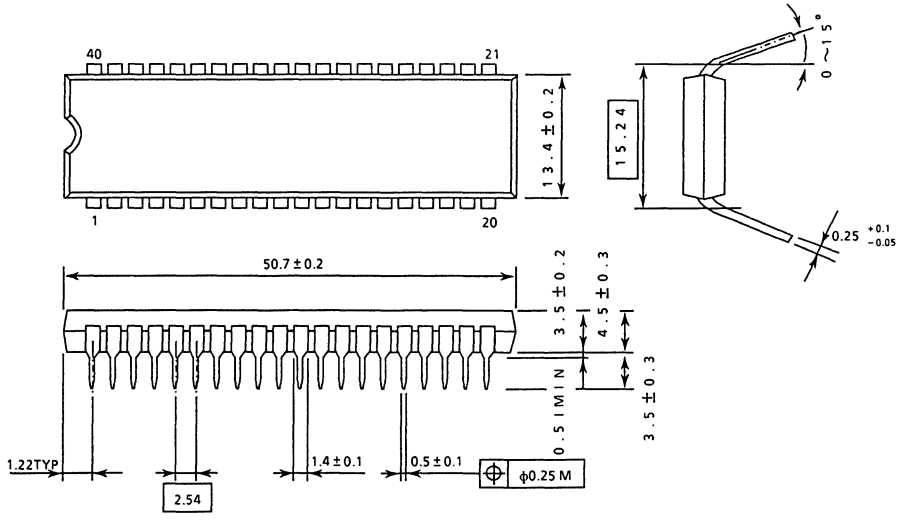


T SUFFIX
44-Lead PLCC Package

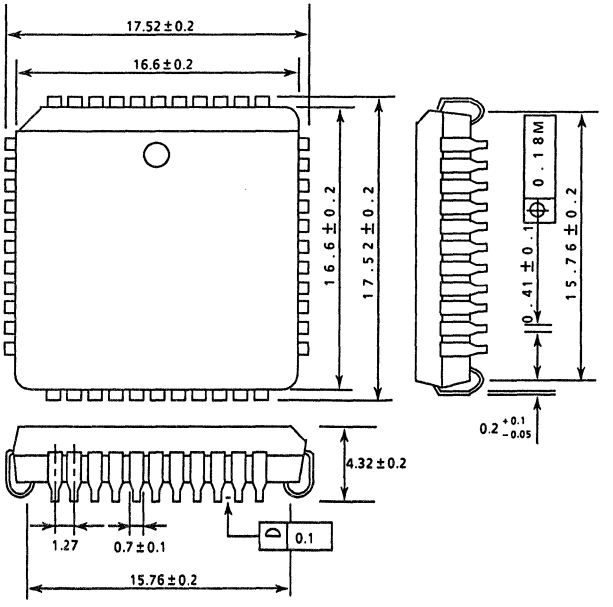


10.2 PACKAGE DIMENSIONS

P SUFFIX
DIP PLASTIC PACKAGE



T SUFFIX
PLCC PACKAGE



SECTION 2
TMP68HC11A8

1. This technical data may be controlled under U.S. Export Administration Regulations and may be subject to the approval of the U.S. Department of Commerce prior to export. Any export or re-export directly or indirectly, in contravention of the U.S. Export Administration Regulations is strictly prohibited.

2. LIFE SUPPORT POLICY

   Toshiba products described in this databook are not authorized for use as critical components in life support systems without the written consent of the appropriate officer of Toshiba America, Inc. Life support systems are either systems intended for surgical implant in the body or systems which sustain life.

   A critical component is any component of a life support system whose failure to perform may cause a malfunction or failure of the life support system, or may affect its safety or effectiveness.

3. The information in this databook has been carefully checked and is believed to be reliable, however, no responsibility can be assumed for inaccuracies that may not have been caught. All information in this databook is subject to change without prior notice. Furthermore, Toshiba cannot assume responsibility for the use of any license under the patent rights of Toshiba or any third parties.

4. No part of this manual may be transferred or reproduced without prior permission of Toshiba Corporation.

CONTENTS

1 INTRODUCTION

The TMP68HC11A8 is an advanced 8-bit microcontroller (MCU) with highly sophisticated on-chip peripheral capabilities. New design techniques were used to achieve a nominal bus speed of 2MHz. In addition, the fully static design allows operation at frequencies down to dc, further reducing power consumption.

1.1 FEATURES

The following are some of the hardware and software highlights.

Hardware Features

- 8K Bytes of ROM
- 512 Bytes of EEPROM
- 256 Bytes of RAM (All Saved During Standby) Relocatable to Any 4K Boundary
- Enhanced 16-Bit Timer System:
  - Four Stage Programmable Prescaler
  - Three Input Capture Functions
  - Five Output Compare Functions
- 8-Bit Pulse Accumulator Circuit
- Enhanced NRZ Serial Communications Interface (SCI)
- Serial Peripheral Interface (SPI)
- Eight Channel, 8-Bit Analog-to-Digital Converter
- Real Time Interrupt Circuit
- Computer Operating Properly (COP) Watchdog System
- Available in Dual-in-Line or Leaded Chip Carrier Packages

Software Features

- Enhanced M6800/M6801 Instruction Set
- 16×16 Integer and Fractional Divide Features
- Bit Manipulation
- WAIT Mode
- STOP Mode

1.2 GENERAL DESCRIPTION

The high-density CMOS technology used on the TMP68HC11A8 combines smaller size and higher speeds with the low power and high noise immunity of CMOS. On-chip memory systems include 8K bytes of ROM, 512 bytes of electrically eraseable programmable ROM (EEPROM), and 256 bytes of static RAM.

Major peripheral functions are provided on-chip. An eight channel analog-to-digital (A/D) converter is included with eight bits of resolution. An asynchronous serial communications interface (SCI) and a separate synchronous serial peripheral interface (SPI) are included. The main 16-bit free-running timer system has three input capture lines, five output compare lines, and a real-time interrupt function. An 8-bit pulse accumulator subsystem can count external events or measure external periods.

Self monitoring circuitry is included on-chip to protect against system errors. A computer operating properly (COP) watchdog system protects against software failures. A clock monitor system generates a system reset in case the clock is lost or runs too slow. An illegal opcode detection circuit provides a non-maskable interrupt if an illegal opcode is detected.

Two software controlled operating modes, WAIT and STOP, are available to conserve additional power.

A block diagram of the TMP68HC11A8 is shown in Figure 1.1

1.3 PROGRAMMER'S MODEL

In addition to being able to execute all M6800 and M6801 instructions, the TMP68HC11A8 allows execution of 91 new opcodes. Figure 1.2 shows the seven CPU registers which are available to the programmer.

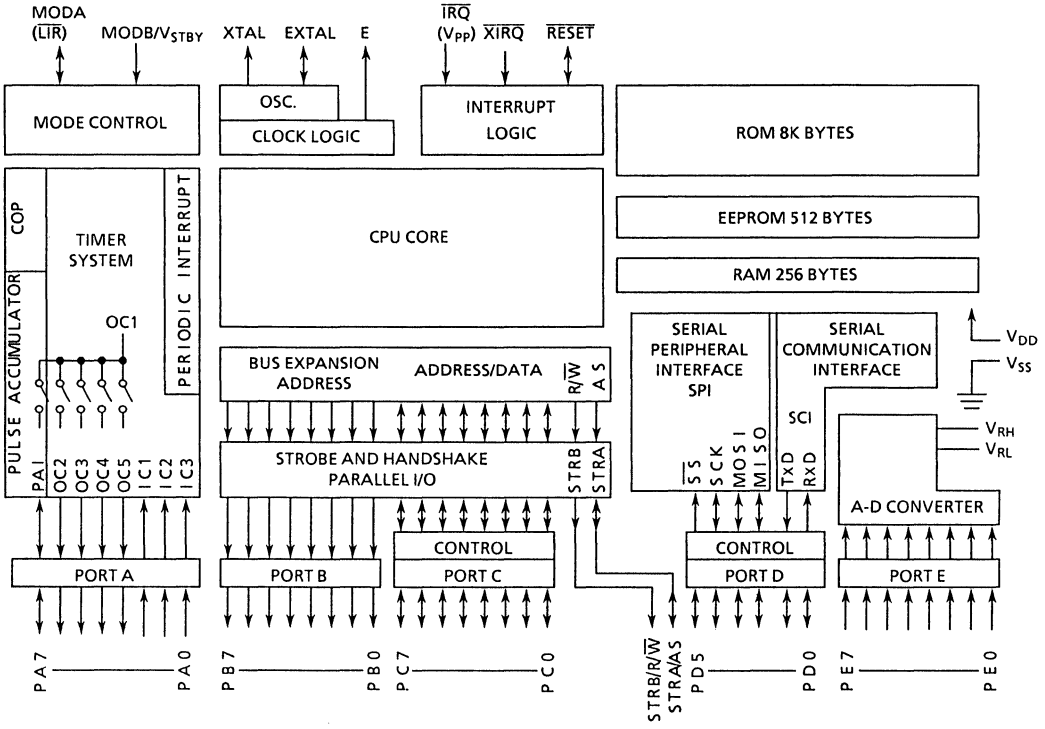


Figure 1.1 Block Diagram

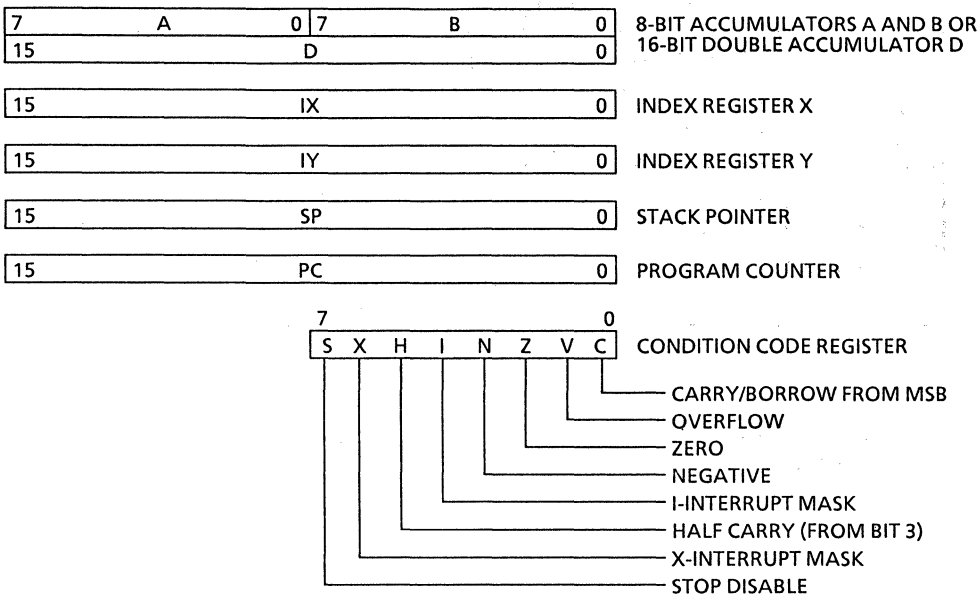


Figure 1.2 Programming Model

1.4 SUMMARY OF TMP68HC11 FAMILY

Table 1.1 and the following paragraphs summarize the current members of the TMP68HC11 Family. This data sheet describes the TMP68HC11A8 version and is to be used as a primary reference for other versions. Family members differ mainly in the types and amounts of memory. The A series parts ('A8, 'A1, and 'A0) are the foundation of the TMP68HC11 Family.

Notice that each major derivative has an x8, x1, and x0 variation. These variations all use identical die. A configuration (CONFIG) register is implemented with EEPROM cells and is used to semipermanently disable the ROM of x1 variations. The ROM and EEPROM are disabled on x0 variations.

The E series was developed for applications requiring four input capture functions for the timer, more ROM, or more RAM. These parts are modified to allow the former output compare five function to be configured as either an output compare or as a fourth input capture function. The amount of RAM was also increased to 512 bytes and the amount of ROM was increased to 12K bytes.

All A series parts are available in 64-pin plastic shrink dual-in-line (S-DIP) packages and 52-pin plastic lend chip carrier (PLCC) packages.

Table 1.1 TMP68HC11 Family Members

Device Number	ROM	EEPROM	RAM	CONFIG[1]	Comments
TMP68HC11A8	8K	512	256	$0F	Family Built Around this Device
TMP68HC11A1	0	512	256	$0D	Same Die as ' A8 but ROM Disabled
TMP68HC11A0	0	0	256	$0C	Same Die as ' A8 but ROM and EEPROM Disabled
TMP68HC11E9	12K	512	512	$0F	Four Input Captures and Bigger RAM and 12K ROM
TMP68HC11E1	0	512	512	$0D	' E9 with ROM Disabled
TMP68HC11E0	0	0	512	$0C	' E9 with ROM and EEPROM Disabled[4]

Notes:

1. CONFIG register values in this table reflect the value programmed prior to shipment from TOSHIBA.

2. SIGNAL DESCRIPTIONS AND OPERATING MODES

The signal descriptions and operating modes are presented in this section. When the microcontroller is in an expanded multiplexed operating mode, 18 pins change function to support a multiplexed address/data bus.

2.1 SIGNAL PIN DESCRIPTIONS

The following paragraphs provide a description of the input/output signals. Reference is made, where applicable, to other sections that contain more detail about the function being performed.

2.1.1 Input Power (V_{DD}) and Ground (V_{SS})

Power is supplied to the microcontroller using these pins. V_{DD} is the positive power input and V_{SS} is ground. Although the TMP68HC11A8 is a CMOS device, very fast signal transitions are present on many of its pins. Short rise and fall times are present even when the microcontroller is operating at slow clock rates. Special care must be taken to provide good power supply bypassing at the MCU. Recommended bypassing would include a 0.1 μF ceramic capacitor between the V_{DD} and V_{SS} pins and physically adjacent to one of the two pins. A bulk capacitance, whose size depends on the other circuity in the system, should also be present on the circuit board.

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

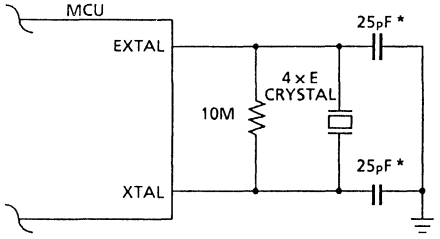
This active low bidirectional control signal is used as an input to initialize the TMP68HC11A8 to a known startup state, and as an open-drain output to indicate that an internal failure has been detected in either the clock monitor or computer operating properly (COP) watchdog circuit. This reset signal is significantly different from the reset signal used on other Motorola MCUs. Please refer to SECTION 9 RESETS, INTERRUPTS, AND LOW POWER MODES before designing circuitry to generate or monitor this signal.

2.1.3 Crystal Driver and External Clock Input (XTAL, EXTAL)

These two pins provide the interface for either a crystal or a CMOS compatible clock to control the internal clock generator circuitry. The frequency applied to these pins shall be four times higher than the desired E clock rate. The XTAL pin is normally left unterminated when using an external CMOS compatible clock input to the EXTAL pin. However, a 10K to 100K load resistor to ground may be used to reduce RFI noise emission. The XTAL output is normally intended to drive only a crystal.

The XTAL output may be buffered with a high input impedance buffer such as the 74HC04, or it may be used to drive the EXTAL input of another TMP68HC11.

In all cases take extra care in the circuit board layout around the oscillator pins. Load capacitances shown in the oscillator circuits include all stray layout capacitances. Refer to Figures 2.1, 2.2, and 2.3 for diagrams of oscillator circuits.



* This value includes all stray capacitances.

Figure 2.1 Common Crystal Connections

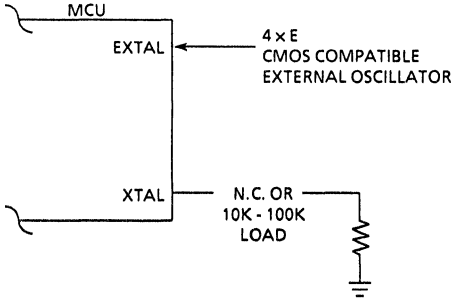


Figure 2.2 External Oscillator Connections

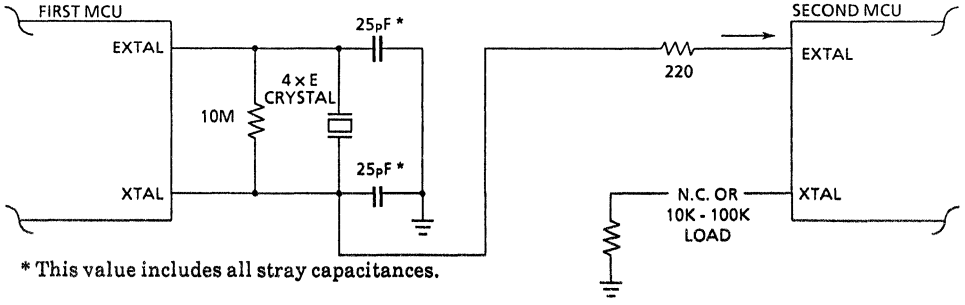


Figure 2.3 One Crystal Driving 2 MCUs

2.1.4 E Clock Output (E)

This is the output connection for the internally generated E clock which can be used as a timing reference. The frequency of the E clock output is actually one fourth that of the input frequency at the XTAL and EXTAL pins. When the E clock output is low an internal process is taking place and, when high, data is being accessed. The E clock signal is halted when the MCU is in STOP mode.

2.1.5 Interrupt Request ($\overline{IRQ}$)

The $\overline{IRQ}$ input provides a means for requesting asynchronous interrupts to the TMP68HC11A8. It is program selectable (OPTION register) with a choice of either negative edge-sensitive or level-sensitive triggering, and is always configured to level-sensitive triggering by reset. The $\overline{IRQ}$ pin requires an external pullup resistor to V_{DD} (typically 4.7K ohm).

During factory testing, this pin is also used as a bulk V_{PP} power supply-input. This allows for parallel programming of as many as half of the bytes in the EEPROM in a single programming operation.

2.1.6 Non-Maskable Interrupt ($\overline{XIRQ}$)

This input provides a means for requesting a non-maskable interrupt, after reset initialization. During reset, the X bit in the condition code register is set and any interrupt is masked until MCU software enables it. The $\overline{XIRQ}$ input is level sensitive and requires an external pullup resistor to V_{DD}.

2.1.7 Mode A/Load Instruction Register and Mode B/Standby Voltage (MODA/$\overline{LIR}$, MODB/V_{STBY})

During reset, MODA and MODB are used to select one of the four operating modes. Refer to Table 2.1 Paragraph 2.2 OPERATING MODES provides additional information.

After the operating mode has been selected, the $\overline{LIR}$ pin provides an open-drain output to indicate that an instruction is starting. All instructions are made up of a series of E clock cycles. The $\overline{LIR}$ signal goes low during the first E clock cycle of each instruction (opcode fetch). This output is provided as an aid in program debugging.

Table 2.1 Operating Modes Versus MODA and MODB

MODB	MODA	Mode Selected
1	0	Single Chip
1	1	Expanded Multiplexed
0	0	Special Bootstrap
0	1	Special Test

The V_{STBY} signal is used as the input for RAM standby power. When the voltage on this pin is more than one MOS threshold (about 0.7 volts) above the V_{DD} voltage, the internal 256-byte RAM and part of the reset logic are powered from this signal rather than the V_{DD} input. This allows RAM contents to be retained without V_{DD} power applied to the MCU. Reset must be driven low before V_{DD} is removed and must remain low until V_{DD} has been restored to a valid level.

2.1.8 A/D Converter Reference Voltages (V_{RL}, V_{RH})

These two inputs provide the reference voltages for the analog-to-digital converter circuitry.

2.1.9 Strobe B and Read/Write (STRB/R/$\overline{W}$)

This signal acts as a strobe B output or as a data bus direction indicator depending on the operating mode.

In single-chip operating mode, the STRB output acts as a programmable strobe for handshake with other parallel I/O devices. Refer to SECTION 4 PARALLEL I/O for additional information.

In expanded multiplexed operating mode, R/$\overline{W}$ is used to control the direction of transfers on the external data bus. A low on the R/$\overline{W}$ signal indicates data is being written to the external data bus. A high on this signal indicates that a read cycle is in progress. R/$\overline{W}$ will stay low during consecutive data bus write cycles, such as in a double-byte store. The NAND of inverted R/$\overline{W}$ with the E clock should be used as the write enable signal for an external static RAM.

2.1.10 Strobe A and Address Strobe (STRA/AS)

This signal acts as an edge detecting strobe A input or as an address strobe bus control output depending on the operating mode.

In single-chip operating mode, the STRA input acts as a programmable strobe for handshake with other parallel I/O devices. Refer to SECTION 4 PARALLEL I/O for additional information.

In expanded multiplexed operating mode, the AS output is used to demultiplex the address and data signals at port C. Refer to 2.2.2 Expanded Multiplexed Operating Mode for additional information.

2.1.11 Port Signals

Ports A, D, and E signals are independent of the operating mode. Port B provides eight general purpose output signals in single-chip operating modes and provides eight high-order address signals when the microcontroller is in expanded multiplexed operating modes. Port C provides eight general purpose input/output signals when the microcontroller is in single-chip operating modes. When the microcontroller is in

expanded multiplexed operating modes, port C is used for a multiplexed address/data bus. Table 2.2 shows a summary of the 40 port signals as they relate to the operating modes. Unused inputs and I/O pins configured as inputs should be terminated high or low.

Table 2.2 Port Signal Summary

Port-Bit	Single-Chip and Bootstrap Mode	Expanded Multiplexed and Special Test Mode
A-0	PA0/IC3	PA0/IC3
A-1	PA1/IC2	PA1/IC2
A-2	PA2/IC1	PA2/IC1
A-3	PA3/OC5/and-or OC1	PA3/OC5/and-or OC1
A-4	PA4/OC4/and-or OC1	PA4/OC4/and-or OC1
A-5	PA5/OC3/and-or OC1	PA5/OC3/and-or OC1
A-6	PA6/OC2/and-or OC1	PA6/OC2/and-or OC1
A-7	PA7/PAI/and-or OC1	PA7/PAI/and-or OC1
B-0	PB0	A8
B-1	PB1	A9
B-2	PB2	A10
B-3	PB3	A11
B-4	PB4	A12
B-5	PB5	A13
B-6	PB6	A14
B-7	PB7	A15
C-0	PC0	A0/D0
C-1	PC1	A1/D1
C-2	PC2	A2/D2
C-3	PC3	A3/D3
C-4	PC4	A4/D4
C-5	PC5	A5/D5
C-6	PC6	A6/D6
C-7	PC7	A7/D7
D-0	PD0/RxD	PD0/RxD
D-1	PD1/TxD	PD1/TxD
D-2	PD2/MISO	PD2/MISO
D-3	PD3/MOSI	PD3/MOSI
D-4	PD4/SCK	PD4/SCK
D-5	PD5/$\overline{SS}$	PD5/$\overline{SS}$
D-6	STRA	AS
D-7	STRB	R/$\overline{W}$
E-0	PE0/AN0	PE0/AN0
E-1	PE1/AN1	PE1/AN1
E-2	PE2/AN2	PE2/AN2
E-3	PE3/AN3	PE3/AN3
E-4	PE4/AN4	PE4/AN4
E-5	PE5/AN5	PE5/AN5
E-6	PE6/AN6	PE6/AN6
E-7	PE7/AN7	PE7/AN7

2.1.11.1 Port A.

Port A may be configured for: three input capture functions (IC1, IC2, IC3), four output compare functions (OC2, OC3, OC4, OC5) , and either a pulse accumulator input (PAI) or a fifth output compare function (OC1) . Refer to 8.1 PROGRAMMABLE TIMER for additional information.

Any port A pin that is not used for its alternate timer function may be used as a general-purpose input or output line.

2.1.11.2 Port B.

While in single-chip operating modes, all of the port B pins are general-purpose output pins. During MCU reads of this port, the level sensed at the input side of the port B output drivers is read. Port B may also be use in a simple strobed output mode where an output pulse appears at the STRB signal each time data is written to port B.

When in expanded multiplexed operating modes, all of the port B pins act as high order address output signals. During each MCU cycle, bits 8 through 15 of the address are output on the PB0-PB7 lines respectively.

2.1.11.3 Port C.

While in single-chip operating modes, all port C pins are general-purpose input/output pins. Port C inputs can be latched by providing an input transition to the STRA signal. Port C may also be used in full handshake modes of parallel I/O where the STRA input and STRB output act as handshake control lines.

When in expanded multiplexed operating modes, all port C pins are configured as multiplexed address/data signals. During the address portion of each MCU cycle, bits 0 through 7 of the address are output on the PC0-PC7 lines. During the data portion of each MCU cycle (E high), pins 0 through 7 are bidirectional data signals (D0-D7). The direction of data at the port C pins is indicated by the R/$\overline{W}$ signal.

2.1.11.4 Port D.

Port D pins 0-5 may be used for general purpose I/O signals. Port D pins alternately serve as the serial communications interface (SCI) and serial peripheral interface (SPI) signals when those subsystems are enabled.

Pin PD0 is the receive data input (RxD) signal for the serial communication interface (SCI).

Pin PD1 is the transmit data output (TxD) signal for the SCI.

Pins PD2 through PD5 are dedicated to the SPI. PD2 is the master-in-slave-out (MISO) signal. PD3 is the master-out-slave in (MOSI) signal. PD4 is the serial clock (SCK) signal and PD5 is the slave select ($\overline{SS}$) input.

2.1.11.5 Port E.

Port E is used for general-purpose inputs and/or analog-to-digital (A/D) input channels. Reading port E during the sampling portion of an A/D conversion could cause very small disturbances and affect the accuracy of that result. If very high accuracy is required, avoid reading port E during conversions.

2.2 OPERATING MODES

There are four operating modes for the TMP68HC11A8:single-chip operating mode, expanded multiplexed operating mode, special bootstrap operating mode, and special test operating mode.

Table 2.1 shows how the operating mode is selected. The following paragraphs describe these operating modes.

2.2.1 Single-Chip Operating Mode

In single-chip operating mode, the TMP68HC11A8 functions as a monolithic microcontroller without external address or data buses. Port B, port C, strobe A, and strobe B function as general purpose I/O and handshake signals. Refer to SECTION 4 PARALLEL I/O for additional information.

2.2.2 Expanded Multiplexed Operating Mode

In expanded multiplexed operating mode, the TMP68HC11A8 has the capability of accessing a 64K byte address space. This total address space includes the same on-chip memory addresses used for single-chip operating mode plus external peripheral and memory devices. The expansion bus is made up of port B and port C, and control signals AS and R/$\overline{W}$. Figure 2.4 shows a recommended way of demultiplexing low order addresses from data at port C. The address, R/$\overline{W}$, and AS signals are active and valid for all bus cycles including accesses to internal memory locations.

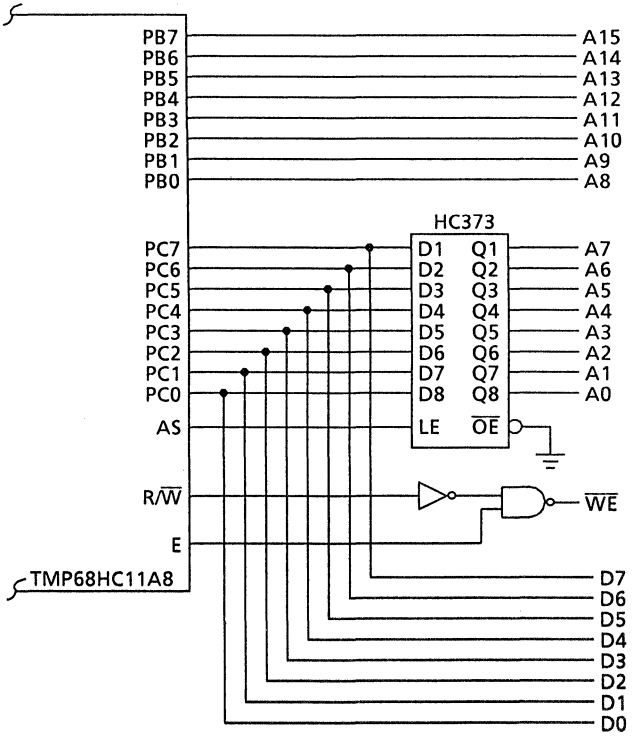


Figure 2.4. Address/Data Demultiplexing

2.2.3 Special Bootstrap Operating Mode

The bootstrap mode is considered a special operating mode as distinguished from the normal single-chip operating mode. This is a very versatile operating mode since there are essentially no limitations on the special purpose program that can be loaded into the internal RAM. The boot loader program is contained in the 192 byte bootstrap ROM. This ROM is enabled only if the MCU is reset in special bootstrap operating mode, and appears as internal memory space at locations $BF40-$BFFF. The boot loader program will use the SCI to read a 256 byte program into on-chip RAM at locations $0000-$00FF.

After the character for address $00FF is received, control is automatically passed to that program at location $0000.

The TMP68HC11A8 communicates through the SCI port. After reset in special bootstrap operating mode, the SCI is running at E clock/16 (7812 baud for E clock equal 2 MHz). If the security feature was specified and the security bit is set, $FF is output by the SCI transmitter. The EEPROM is then erased. If erasure is unsuccessful, $FF is

output again and erasure is attempted again. Upon successful erasure of the EEPROM, all internal RAM is written over with $FF. The CONFIG register is then erased. The boot loader program now proceeds as though the part had not been in security mode.

If the part is not in security mode (or has completed the above erase sequence), a break character is output by the SCI transmitter. For normal use of the boot loader program, the user sends $FF to the SCI receiver at either E clock/16 (7812 baud for E clock = 2MHz) or E clock/104 (1200 baud for E clock = 2MHz).

Note : This $FF is not echoed through the SCI transmitter.

Now the user must download 256 bytes of program data to be put into RAM starting at location $0000. These characters are echoed through the transmitter. When loading is complete, the program jumps to location $0000 and begins executing that code.

If the SCI transmitter pin is to be used, an external pullup resistor is required because port D pins are configured for wire-OR operation.

In special bootstrap operating mode the interrupt vectors are directed to RAM as shown in Table 2.3. This allows the user to use interrupts by way of a jump table. For example: to use the SWI interrupt, a jump instruction would be placed in RAM at locations $00F4, $00F5, and $00F6. When an SWI is encountered, the vector (which is in the boot loader ROM program) will direct program control to location $00F4 in RAM which in turn contains a JUMP instruction to the interrupt service routine.

Table 2.3 Bootstrap Mode Interrupt Vectors

Address	Vector
00C4	SCI
00C7	SPI
00CA	Pulse Accumulator Input Edge
00CD	Pulse Accumulator Overflow
00D0	Timer Overflow
00D3	Timer Output Compare 5
00D6	Timer Output Compare 4
00D9	Timer Output Compare 3
00DC	Timer Output Compare 2
00DF	Timer Output Compare 1
00E2	Timer Input Capture 3
00E5	Timer Input Capture 2
00E8	Timer Input Capture 1
00EB	Real Time Interrupt
00EE	IRQ
00F1	XIRQ
00F4	SWI
00F7	Illegal Opcode
00FA	COP Fail
00FD	Clock Monitor
BF40 (Boot)	Reset

2.2.4 Additional Boot Loader Program Options

The user may transmit a $55 (only at E clock/16) as the first character rather than the normal $FF. This will cause the program to jump directly to location $0000, skipping the download.

The user may tie the receiver to the transmitter (with an external pull-up resistor). This will cause the program to jump directly to the beginning of EEPROM ($B600). Another way to cause the program to jump directly to EEPROM is to transmit either a break or $00 as the first character rather than the normal $FF.

Note that none of these options bypass the security check and so do not compromise those customers using security.

Keep in mind that upon entry to the downloaded program at location $0000, some registers have been changed from their reset states. The SCI transmitter and receiver are enabled which cause port D pins 0 and 1 to be dedicated to SCI use. Also port D is configured for wired-OR operation. It may be necessary for the user to write to the SCCR2 and SPCR registers to disable the SCI and/or port D wire-OR operation.

2.2.5 Special Test Operating Mode

The test mode is a special operating mode intended primarily for factory testing. This mode is very similar to the expanded multiplexed operating mode. In special test operating mode, the reset and interrupt vectors are fetched from external memory locations $BFC0-BFFF rather than $FFC0-$FFFF. There are no time limits for protection of the TMSK2, OPTION, and INIT registers, so these registers may be written repeatedly. Also a special TEST1 register is enabled which allows several factory test functions to be invoked.

The special test operating mode is not recommended for use by an end user because of the reduced system security;however, an end user may wish to come out of reset in special test operating mode. Then, after some initialization, the SMOD and MDA bits could be rewritten to select a normal operating mode to re-enable the protection features.

3. ON-CHIP MEMORIES

This section describes the on-chip ROM, RAM, and EEPROM memories. The memory maps for each mode of operation are shown and the RAM and I/O mapping register (INIT) is described. The INIT register allows the on-chip RAM and the 64 control registers to be moved to suit the needs of a particular application.

3.1 MEMORY MAPS

Composite memory maps for each mode of operation are shown in Figure 3.1. Memory locations are shown in the shaded areas and the contents of these shaded areas are shown to the right. These modes include single-chip, expanded multiplexed, special bootstrap, and special test.

Single-chip operating modes do not generate external addresses. Refer to Table 3.1 for a full list of the registers.

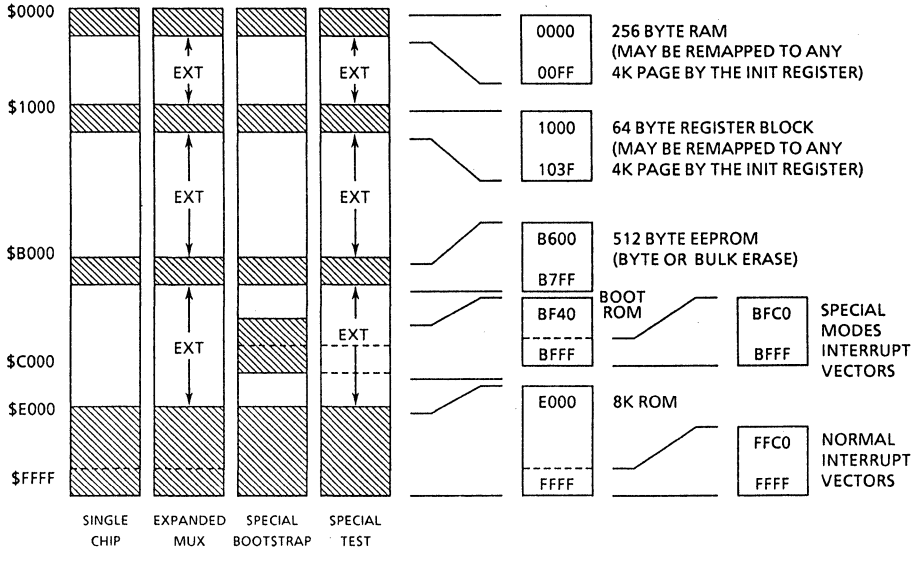


Figure 3.1 Memory Maps

Table 3.1 Register and Control Bit Assignments (Sheet 1 of 2)

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0		
$1000	Bit7	–	–	–	–	–	–	Bit0	PORTA	I/O Port A
$1001									Reserved	
$1002	STAF	STAI	CWOM	HNDS	OIN	PLS	EGA	INVB	PIOC	Parallel I/O Control Register
$1003	Bit7	–	–	–	–	–	–	Bit0	PORTC	I/O Port C
$1004	Bit7	–	–	–	–	–	–	Bit0	PORTB	I/O Port B
$1005	Bit7	–	–	–	–	–	–	Bit0	PORTCL	Alternate Latched Port C
$1006									Reserved	
$1007	Bit7	–	–	–	–	–	–	Bit0	DDRC	Data Direction for Port C
$1008			Bit5	–	–	–	–	Bit0	PORTD	I/O Port D
$1009			Bit5	–	–	–	–	Bit0	DDRD	Data Direction for Port D
$100A	Bit7	–	–	–	–	–	–	Bit0	PORTE	Input Port E
$100B	FOC1	FOC2	FOC3	FOC4	FOC5				CFORC	Compare Force Register
$100C	OC1M7	OC1M6	OC1M5	OC1M4	OC1M3				OC1M	OC1 Action Mask Register
$100D	OC1D7	OC1D6	OC1D5	OC1D4	OC1D3				OC1D	OC1 Action Data Register
$100E	Bit15	–	–	–	–	–	–	Bit8	TCNT	Timer Counter Register
$100F	Bit7	–	–	–	–	–	–	Bit0		
$1010	Bit15	–	–	–	–	–	–	Bit8	TIC1	Input Capture 1 Register
$1011	Bit7	–	–	–	–	–	–	Bit0		
$1012	Bit15	–	–	–	–	–	–	Bit8	TIC2	Input Capture 2 Register
$1013	Bit7	–	–	–	–	–	–	Bit0		
$1014	Bit15	–	–	–	–	–	–	Bit8	TIC3	Input Capture 3 Register
$1015	Bit7	–	–	–	–	–	–	Bit0		
$1016	Bit15	–	–	–	–	–	–	Bit8	TOC1	Output Compare 1 Register
$1017	Bit7	–	–	–	–	–	–	Bit0		
$1018	Bit15	–	–	–	–	–	–	Bit8	TOC2	Output Compare 2 Register
$1019	Bit7	–	–	–	–	–	–	Bit0		
$101A	Bit15	–	–	–	–	–	–	Bit8	TOC3	Output Compare 3 Register
$101B	Bit7	–	–	–	–	–	–	Bit0		
$101C	Bit15	–	–	–	–	–	–	Bit8	TOC4	Output Compare 4 Register
$101D	Bit7	–	–	–	–	–	–	Bit0		
$101E	Bit15	–	–	–	–	–	–	Bit8	TOC5	Output Compare 5 Register
$101F	Bit7	–	–	–	–	–	–	Bit0		

Table 3.1 Register and Control Bit Assignments (Sheet 2 of 2)

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0		
$1020	OM2	OL2	OM3	OL3	OM4	OL4	OM5	OL5	TCTL1	Timer Control Register 1
$1021			EDG1B	EDG1A	EDG2B	EDG2A	EDG3B	EDG3A	TCTL2	Timer Control Register 2
$1022	OC1I	OC2I	OC3I	OC4I	OC5I	IC1I	IC2I	IC3I	TMSK1	Timer Interrupt Mask Register 1
$1023	OC1F	OC2F	OC3F	OC4F	OC5F	IC1F	IC2F	IC3F	TFLG1	Timer Interrupt Flag Register 1
$1024	TOI	RTII	PAOVI	PAII			PR1	PR0	TMSK2	Timer Interrupt Mask Register 2
$1025	TOF	RTIF	PAOVF	PAIF					TFLG2	Timer Interrupt Flag Register 2
$1026	DDRA7	PAEN	PAMOD	PEDGE			RTR1	RTR0	PACTL	Pulse Accumulator Control Register
$1027	Bit7	–	–	–	–	–	–	Bit0	PACNT	Pulse Accumulator Count Register
$1028	SPIE	SPE	DWOM	MSTR	CPOL	CPHA	SPR1	SPR0	SPCR	SPI Control Register
$1029	SPIF	WCOL		MODF					SPSR	SPI Status Register
$102A	Bit7	–	–	–	–	–	–	Bit0	SPDR	SPI Data Register
$102B	TCLR		SCP1	SCP0	RCKB	SCR2	SCR1	SCR0	BAUD	SCI Baud Rate Control
$102C	R8	T8		M	WAKE				SCCR1	SCI Control Register 1
$102D	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK	SCCR2	SCI Control Register 2
$102E	TDRE	TC	RDRF	IDLE	OR	NF	FE		SCSR	SCI Status Register
$102F	Bit7	–	–	–	–	–	–	Bit0	SCDR	SCI Data (Read RDR, Write TDR)
$1030	CCF		SCAN	MULT	CD	CC	CB	CA	ADCTL	A/D Control Register
$1031	Bit7	–	–	–	–	–	–	Bit0	ADR1	A/D Result Register 1
$1032	Bit7	–	–	–	–	–	–	Bit0	ADR2	A/D Result Register 2
$1033	Bit7	–	–	–	–	–	–	Bit0	ADR3	A/D Result Register 3
$1034	Bit7	–	–	–	–	–	–	Bit0	ADR4	A/D Result Register 4
$1035 Thru $1038									Reserved	
$1039	ADPU	CSEL	IRQE	DLY	CME		CR1	CR0	OPTION	System Configuration Options
$103A	Bit7	–	–	–	–	–	–	Bit0	COPRST	Arm/Reset COP Timer Circuitry
$103B	ODD	EVEN		BYTE	ROW	ERASE	EELAT	EEPGM	PPROG	EEPROM Programming Control Register
$103C	RBOOT	SMOD	MDA	IRV	PSEL3	PSEL2	PSEL1	PSEL0	HPRIO	Highest Priority I-Bit Int and Misc
$103D	RAM3	RAM2	RAM1	PAM0	REG3	REG2	REG1	REG0	INIT	RAM and I/O Mapping Register
$103E	TILOP		OCCR	CBYP	DISR	FCM	FCOP	TCON	TEST1	Factory TEST Control Register
$103F	–	–	–	–	NOSEC	NOCOP	ROMON	EEON	CONFIG	COP, ROM, & EEPROM Enables

In expanded multiplexed operating modes, memory locations are basically the same as the single-chip operating modes; however, the locations between the shaded areas (designated EXT) are for externally addressed memory and I/O. If an external memory or I/O device is located to overlap an enabled internal resource, the internal resource will take priority. For reads of such an address the data (if any) driving the port C data inputs is ignored and will not result in any harmful conflict with the internal read. For writes to such an address data is driven out of the port C data pins as well as to the internal location. No external devices should drive port C during write accesses to internal locations; however, there is normally no conflict since the external address decode and/or data direction control should incorporate the R/$\overline{W}$ signal in their development. The R/$\overline{W}$, AS, address, and write data signals are valid for all accesses including accesses to internal memory and registers.

The special bootstrap operating mode memory locations are similar to the single-chip operating mode memory locations except that a bootstrap program at memory locations $BF40 through $BFFF is enabled. The reset and interrupt vectors are addressed at $BFC0-$BFFF while in the special bootstrap operating mode. These vector addresses are within the 192 byte memory used for the bootstrap program.

The special test operating mode memory map is the same as the expanded multiplexed operating mode memory map except that the reset and interrupt vectors are located at external memory locations $BFC0-$BFFFF.

3.2 RAM AND I/O MAPPING REGISTER (INIT)

There are 64 internal registers which are used to control the operation of the MCU. These registers can be relocated on 4K boundaries within the memory space, using the INIT register. Refer to Table 3.1 (found on a foldout page at the back of this document) for a complete list of the registers. The registers and control bits are explained throughout this document.

The INIT register is a special-purpose 8-bit register which may be used during initialization to change the default locations of RAM and control registers within the MCU memory map. It may be written to only once within the initial 64E clock cycles after a reset and thereafter becomes a read-only register.

	7	6	5	4	3	2	1	0	
$103D	RAM3	RAM2	RAM1	RAM0	REG3	REG2	REG1	REG0	INIT
RESET	0	0	0	0	0	0	0	1	

The default starting address for internal RAM is $0000 and the default starting address for the 64 control registers is $1000 (the INIT register is set to $01 by reset). The upper four bits of the INIT register specify the starting address for the 256 byte RAM

and the lower four bits of INIT specify the starting address for the 64 control registers. These four bits are matched to the upper four bits of the 16-bit address.

Throughout this document, the control register addresses will be displayed with the high-order digit shown as a bold "1" to indicate that the register block may be relocated to some 4K memory page other than its default position of $1000-$103F.

Note that if the RAM is relocated to either $E000 or $F000, which is in conflict with the internal ROM, (no conflict if the ROMON bit in the configuration register is zero), RAM will take priority and the conflicting ROM will become inaccessible. Also, if the 64 control registers are relocated so that they conflict with the RAM and/or ROM, then the 64 control registers take priority and the RAM and/or ROM at those locations become inaccessible. No harmful conflicts result, the lower priority resources simply become inaccessible. Similarly, if an internal resource conflicts with an external device no harmful conflict results. Data from the external device will not be applied to the internal data bus and cannot interfere with the internal read.

Note : that there are unused register locations in the 64 byte control register block. Reads of these unused registers will return data from the undriven internal data bus and not from another resource that happens to be located at the same address.

3.3 ROM

The internal 8K ROM occupies the highest 8K of the memory map ($E000-$FFFF). This ROM is disabled when the ROMON bit in the CONFIG register is clear. The ROMON bit is implemented with an EEPROM cell and is programmed using the same procedures for programming the on-chip EEPROM. For further information refer to 3.5.3 System Configuration Register (CONFIG).

In the single-chip operating mode, internal ROM is enabled regardless of the state of the ROMON bit.

There is also a 192 byte mask programmed boot ROM in the TMP68HC11A8. This bootstrap program ROM controls the operation of the special bootstrap operating mode and is only enabled following reset in the special bootstrap operating mode. For more information refer to 2.2.3 Special Bootstrap Operating Mode.

3.4 RAM

The 256 byte internal RAM may be relocated during initialization by writing to the INIT register.

The reset default position is $0000 through $00FF. This RAM is implemented with static cells and retains its contents during the WAIT and STOP modes.

The contents of the 256-byte RAM can also be retained by supplying a low current backup power source to the MODB/V_{STBY} pin. When using a standby power source, V_{DD}

may be removed;however, reset must go low before V_{DD} is removed and remain low until V_{DD} has been restored.

3.5 EEPROM

The 512 bytes of EEPROM are located at $B600 through $B7FF and have the same read cycle time as the internal ROM. The write (or programming) mechanism for the EEPROM is controlled by the PPROG register. The EEPROM is disabled when the EEON bit in the CONFIG register is zero. The EEON bit is implemented with an EEPROM cell.

The erased state of an EEPROM byte is $FF. Programming changes ones to zeros. If any bit in a location needs to be changed from a zero to a one, the byte must be erased in a separat operation before it is reprogrammed. If a new data byte has no ones in bit positions which were already programmed to zero, it is acceptable to program the new data without erasing the EEPROM byte first. For example, programming $50 to a location which was already $55 would change the location to $50.

Programming and erasure of the EEPROM relies on an internal high-voltage charge pump. At E clock frequencies below 2 MHz the efficiency of this charge pump decreases which increases the time required to program or erase a location. The recommended program and erase time is 10 milliseconds when the E clock is 2 MHz and should be increased to as much as 20 milliseconds when E is between 1 MHz and 2MHz. When the E clock is below 1 MHz, the clock source for the charge pump should be switched from the system clock to an on-chip R-C oscillator clock. This is done by setting the CSEL bit in the OPTION register. A 10 millisecond period should be allowed after setting the CSEL bit to allow the charge pump to stabilize. Note that the CSEL bit also controls a clock to the analog to digital converter subsystem.

3.5.1 EEPROM Programming Control Register (PPROG)

This 8-bit register is used to control programming and erasure of the 512-byte EEPROM. Reset clears this register so the EEPROM is configured for normal reads.

	7	6	5	4	3	2	1	0	
$103B	ODD	EVEN	0	BYTE	ROW	ERASE	EELAT	EEPGM	PPROG
RESET	0	0	0	0	0	0	0	0	

ODD - Program Odd Rows (TEST)

EVEN - Program Even Rows (TEST)

Bit5 - Not implemented.

This bit always reads zero.

BYTE - Byte Erase Select

This bit overrides the ROW bit.

0 = Row or Bulk Erase

1 = Erase Only One Byte

ROW - Row Erase Select

If the BYTE bit is 1, ROW has no meaning.

0 = Bulk Erase

1 = Row Erase

ERASE - Erase Mode Select

0 = Normal Read or Program

1 = Erase Mode

EELAT - EEPROM Latch Control

0 = EEPROM Address and Data Configured for Read Mode

1 = EEPROM Address and Data Configured for Programming/Erasing

EEPGM - EEPROM Programming Voltage Enable

0 = Programming Voltage Switched Off

1 = Programming Voltage Turned On

If an attempt is made to set both the EELAT and EEPGM bits in the same write cycle, neither will be set. If a write to an EEPROM address is performed while the EEPGM bit is set, the write is ignored and the programming operation currently in progress is not disturbed. These two safe-guards were included to prevent accidental EEPROM changes in cases of program runaway. Mask sets A38P, A49N, and date codes before 86xx did not have these safeguards.

3.5.2 Programming/Erasing Internal EEPROM

The EEPROM programming and erasure process is controlled by the PPROG register. The following paragraphs describe the various operations performed on the EEPROM and include example program segments to demonstrate programming and erase operations.

These program segments are intended to be simple straightforward examples of the sequences needed for basic program and erase operations. There are no special restrictions on the address modes used and bit manipulation instructions may be used. Other MCU operations can continue to be peformed during EEPROM programming and erasure provided these operations do not include reads of data from EEPROM (the EEPROM is disconnected from the read data bus during EEPROM program and erase

operations). The subroutine DLY10 used in these program segments is not shown but can be any set of instructions which takes ten milliseconds.

3.5.2.1 Read.

For the read operation the EELAT bit in the PPROG register must be clear. When this bit is cleared, the remaining bits in the PPROG register have no meaning or effect, and the EEPROM may be read as if it were a normal ROM.

3.5.2.2 Programming.

During EEPROM programming, the ROW and BYTE bits are not used. If the E clock frequency is 1 MHz or less, the CSEL bit in the OPTION register must be set. Recall that in this EEPROM, zeros must be erased by a separate erase operation before programming. The following program segment demonstrates how to program an EEPROM byte.

*On entry, A = data to be programmed and X = an EEPROM address

```
        •
        •
        •
PROG    LDAB    #$02
        STAB    $103B    Set EELAT Bit (EEPGM=0)
        STAA    0, X     Store Data to EEPROM Address
        LDAB    #$03
        STAB    $103B    Set EEPGM Bit (EELAT=1)
        JSR     DLY10    Delay 10 ms
        CLR     $103B    Turn Off High Voltage and Set to READ Mode
        •
        •
        •
```

3.5.2.3 Bulk Erase.

The following program segment demonstrates how to bulk erase the 512-byte EEPROM. The CONFIG register is not affected in this example.

```
        •
        •
        •
BULKE   LDAB    #$06
        STAB    #103B    Set to Bulk Erase Mode
        STAB    $B600    Write any Data to any EEPROM Address
        LDAB    #$07
```

```
        STAB    $103B    Turn On Programming Voltage
        JSR     DLY10    Delay 10 ms
        CLR     $103B    Turn Off High Voltage and Set to READ Mode
        •
        •
        •
```

3.5.2.4 Row Erase.

The following program segment demonstrates the row erase function. A 'row' is sixteen bytes ($B600-$B60F, $B610-$B61F...$B7F0-$B7FF). This type of erase operation saves time compared to byte erase when large sections of EEPROM are to be erased.

*On entry X = any address in the row to be erased

```
        •
        •
        •
ROWE    LDAB    #$0E
        STAB    $103B    Set to Row Erase Mode
        STAB    0, X     Write any Data to any Address in Row
        LDAB    #$0F
        STAB    $103B    Turn on High Voltage
        JSR     DLY10    Delay 10 ms
        CLR     $103B    Turn Off High Voltage and Set to Read Mode
        •
        •
        •
```

3.5.2.5 Byte Erase. The following program segment shows the byte erase function.

*On entry, X = address of byte to be erased

```
        •
        •
        •
BYTEE   LDAB    #$16
        STAB    $103B    Set to Byte Erase Mode
        STAB    0, X     Write any Data to the Address to Erase
        LDAB    #$17
        STAB    $103B    Turn on High Voltage
        JSR     DLY10    Delay 10 ms
```

```
CLR     $103B     Turn Off High Voltage and Set to Read Mode
 •
 •
 •
```

3.5.3 System Configuration Register (CONFIG)

The TMP68HC11A8 can be configured to specific system requirements through the use of hardwired options such as the mode select pins, semi-permanent EEPROM control bit specifications (CONFIG register), or by use of control registers. The configuration control register (CONFIG) is implemented in EEPROM cells and controls the presence of ROM and EEPROM in the memory map, as well as enabling the COP watchdog system A security feature to protect data in the EEPROM and RAM is also available on mask programmed TMP68HC11A8s.

	7	6	5	4	3	2	1	0	
$103F	0	0	0	0	NOSEC	NOCOP	ROMON	EEON	CONFIG
RESET	(see 3.5.3.2 OPERATION OF THE CONFIG MECHANISM)								

Bits 7, 6, 5, and 4-Not Implemented

These bits are always read as zero.

NOSEC - Security Mode Disable Bit

This bit is only implemented if it is specifically requested at the time mask ROM information is requested. When this bit is not implemented it always reads one.

When RAM and EEPROM security are required, the NOSEC bit can be programmed to zero to enable a software anti-theft mechanism. When clear, the NOSEC bit prevents the selection of expanded multiplexed operating modes. If the MCU is reset in the special bootstrap operating mode while NOSEC is zero, EEPROM, RAM, and CONFIG are erased before the loading process continues.

0 = Enable Security Mode

1 = Disable Security Mode

NOCOP - COP System Disable

0 = COP Watchdog System Enabled

1 = COP Watchdog System Disabled

ROMON - Enable On-Chip ROM

When this bit is clear, the 8K ROM is disabled, and that memory space becomes externally accessed space. In the single-chip operating mode, the internal 8K ROM is enabled regardless of the state of the ROMON bit.

EEON - Enable On-Chip EEPROM

When this bit is clear, the 512-byte EEPROM is disabled, and that memory space becomes externally accessed space.

3.5.3.1 Programming and Erasure of the Config Register.

Since the CONFIG register is implemented with EEPROM cells, special provisions must be made to erase and program this register. The normal EEPROM control bits in the PPROG register are used for this purpose. Programming follows the same procedure as programming a byte in the 512-byte EEPROM except the CONFIG register address is used. Erase also follows the same procedure as that used for the EEPROM except that only bulk erase can be used on the CONFIG register. When the CONFIG register is erased, the 512-byte EEPROM array is also erased. Be sure to check the Technical Summary for the particular TMP68HC11 Family member if you are using a version other than TMP68HC11A8, TMP68HC11A1, or TMP68HC11A0.

On mask set B96D and newer, the CONFIG register may only be programmed or erased while the MCU is operating in the test mode or the bootstrap mode. This interlock was added to help prevent accidental changes to the CONFIG register.

The following program segment demonstrates how to program the CONFIG register. This program assumes that the CONFIG register was previously erased.

*On entry, A = data to be programmed into CONFIG

```
        •
        •
        •
PROGC   LDAB    #$02
        STAB    $103B    Set EELAT Bit (EEPGM = 0)
        STAA    $103F    Store Data to CONFIG Address
        LDAB    #$03
        STAB    $103B    Turn on Programming Voltage
        JSR     DLY10    Delay 10 ms
        CLR     $103B    Turn Off High Voltage and Set to READ Mode
        •
        •
        •
```

The following program segment demonstrates the erase procedure for the CONFIG register.

```
        •
        •
        •
```

```
BULKC   LDAB    #$06
        STAB    $103B   Set Bulk Erase Mode
        STAB    $103F   Write any Data to CONFIG
        LDAB    #$07
        STAB    $103B   Turn on Programming Voltage
        JSR     DLY10   Delay 10 ms
        CLR     $103B   Turn Off High Voltage and Set to READ Mode
        •
        •
        •
```

3.5.3.2 Operation of the Configuration Mechanism.

The CONFIG register consists of an EEPROM byte and static working latches. This register controls the startup configuration of the MCU. The contents of the EEPROM CONFIG byte are transferred into static working latches during any reset sequence. The operation of the MCU is controlled directly by these latches and not the actual EEPROM byte. Changes to the EEPROM byte do not affect operation of the MCU until after the next reset sequence. When programming the CONFIG register, the EEPROM byte is being accessed. When the CONFIG register is being read, the static latches are being accessed.

To change the value in the CONFIG register proceed as follows:

1) Erase the CONFIG register.

Note : Do not issue a reset at this time.

2) Program the new value to the CONFIG register.

3) Issue a reset so the new configuration will take effect.

4. PARALLEL I/O

The TMP68HC11A8 has 40 I/O pins arranged as five 8-bit ports. All of these pins serve multiple functions depending on the operating mode and data in the control register. This section explains the operation of these pins only when they are used for parallel I/O.

Ports C and D are used as general purpose input and/or output pins under direct control of their respective data direction registers. Ports A, B, and E, with the exception of port A pin 7, are fixed direction inputs or outputs and therefore do not have data direction registers. Port B, port C, the STRA pin, and the STRB pin are used for strobed and/or handshake modes of parallel I/O, as well as general purpose I/O.

4.1 GENERAL PURPOSE I/O (PORTS C AND D)

Each port I/O line has an associated bit in a specific port data register and port data direction register. The data direction register bits are used to specify the primary direction of data for each I/O line. When an output line is read, the value at the input to the pin to the pin driver is returned. When a line is configured as an input, that pin becomes a high-impedance input. If a write is executed to an input line, the value does not affect the I/O pin, but is stored in an internal latch. When theline becomes an output, this value appears at the I/O pin. Data direction register bits are cleared by reset to configure I/O pins as inputs.

The AS and R/$\overline{W}$ pins are dedicated to bus control while in the expanded multiplexed operating modes, or parallel I/O strobes (STRA and STRB) while in the single chip operating modes.

4.2 FIXED DIRECTION I/O (PORTS A, B, AND E)

The lines for ports A, B, and E (except for port A bit 7) have fixed data directions. When port A is being used for general purpose I/O, bits 0, 1, and 2 are configured as input only and writes to these lines have no effect. Bits 3, 4, 5, and 6 of port A are configured as output only and reads of these lines return the levels sensed at the input to the line drivers. Port A bit 7 can be configured as either a general-purpose input or output using the DDRA7 bit in the pulse accumulator control register. When port B is being used for general purpose output, it is configured as output only and reads of these lines will return the levels sensed at the input of the pin drivers. Port E contains the eight A/D channel inputs, but these lines may also be used as general purpose digital inputs.Writes to the port E address have no effect.

4.3 SIMPLE STROBED I/O

The simple strobed mode of parallel I/O is invoked and controlled by the parallel I/O control register (PIOC). This mode is selected when the handshake bit (HNDS) in the PIOC register is clear.Port C becomes a strobed input port with the STRA line as the

edge-detecting latch command input. Also, port B becomes a strobed output port with the STRB line as the output strobe. The logic sense of the STRB output is selected by the invert strobe B bit (INVB) in the PIOC register.

4.3.1 Strobed Input Port C

In this mode, there are two addresses where port C may be read, the PORTC data register and the alternate latched port C register (PORTCL). The data direction register still controls the data direction of all port C lines. Even when the strobed input mode is selected, any or all of the port C lines may still be used for general purpose I/O.

The STRA line is used as an edge-detecting input, and the edge-select for strobe A (EGA) bit in the PIOC register defines either falling or rising edge as the significant edge. Whenever the selected edge is detected at the STRA pin, the current logic levels at port C lines are latched into the PORTCL register and the strobe A flag (STAF) in the PIOC register is set. If the strobe A interrupt enable (STAI) bit in PIOC is also set, an internal interrupt sequence is requested. The strobe A flag (STAF) is automatically cleared by reading the PIOC register (with STAF set) followed by a read of the PORTCL register. Data is latched in the PORTCL register whether or not the STAF flag was previously clear.

4.3.2 Strobed Output Port B

In this mode, the STRB pin is a strobe output which is pulsed for two E clock periods each time there is a write to port B. The INVB bit in the PIOC register controls the polarity of the pulse on the STRB line.

4.4 FULL HANDSHAKE I/O

The full handshake modes of parallel I/O involve port C and the STRA and STRB lines. There are two basic modes (input and output) and an additional variation on the output handshake mode that allows three-stated operation of port C. In all handshake modes, STRA is an edge-detecting input, and STRB is a handshake output line.

When full input handshake protocol is specified, both general purpose input and/or general purpose output can coexist at port C. When full output handshake protocol is specified, general purpose output can coexist with the handshake outputs at port C, but the three-state feature of the output handshake mode interferes with general purpose input in two ways. First, in full output handshake, the port C lines are outputs whenever STRA is at its active level regardless of the data direction register bits. This potentially conflicts with any external device trying to drive port C unless that external device has an open-drain type output driver. Second, the value returned on reads of port C is the state of the outputs of an internal port C output latch regardless of the states of the data direction register bits, so that the data written for output handshake can be read even if the pins are in a three-state condition.

4.4.1 Input Handshake Protocol

In the input handshake protocol, port C is a latching input port, STRA is an edge-sensitive latch command from the external system that is driving port C, and STRB is a "ready" output line controlled by logic in the MCU.

When a "ready" condition is recognized, the external device places data on the port C lines, then pulses the STRA line. The active edge on the STRA line latches the port C data into the PORTCL register, sets the STAF flag (optionally causing an interrupt), and deasserts the STRB line. Deassertion of the STRB line automatically inhibits the external device from strobing new data into port C. Reading the PORTCL latch register (independent of clearing the STAF flag) asserts the STRB line, indicating that new data may now be applied to port C.

The STRB line can be configured (with the PLS control bit) to be a pulse output (pulse mode) or a static output (interlocked mode).

The port C data direction register bits should be cleared for each line that is to be used as a latched input line. However, some port C lines can be used as latched inputs with the input handshake protocol while, at the same time, using some port C lines as static inputs, and some port C lines as static outputs. The input handshake protocol has no effect on the use of port C lines as static inputs or as static outputs. Reads of the PORTC data register always return the static logic level at the port C lines (for lines configured as inputs). Writes to either the PORTC data register or the alternate latched port C register (PORTCL) send information to the same port C output register without affecting the input handshake strobes.

4.4.2 Output Handshake Protocol

In the output handshake protocol, port C is an output port, STRB is a "ready" output, and STRA is an edge-sensitive acknowledge input signal, used to indicate to the MCU that the output data has been accepted by the external device. In a variation of this output handshake protocol, STRA is also used as an output-enable input, as well as an edge-sensitive acknowledge input.

The MCU places data on the port C output lines and then indicates stable data is available by asserting the STRB line. The external device then processes the available data and pulses the STRA line to indicate that new data may be placed on the port C output lines. The active edge on the STRA line causes the STRB line to be deasserted and the STAF status flag to be set. In response to the STAF bit being set, the program transfers new data out of port C as required. Writing data to the PORTCL register causes the data to appear on port C lines and asserts the STRB line.

There is a variation to the output handshake protocol that allows three-state operation on port C. It is possible to directly connect this 8-bit parallel port to other three-state devices with no additional parts.

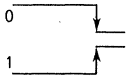
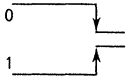
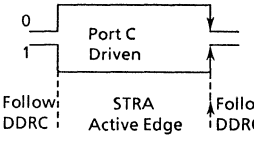
While the STRA input line is inactive, all port C lines obey the data direction specified by the data direction register so that lines which are configured as inputs are high impedance. When the STRA line is activated, all port C lines are forced to outputs regardless of the data in the data direction register. Note that in output handshake protocol, reads of port C always return the value sensed at the input to the output buffer regardless of the state of the data direction register bits because the lines would not necessarily have meaningful data on them in the three-state variation of this protocol. This operation makes it inpracticle to use some port C lines as static inputs, while using others as handshake output, but does not interfere with the use of some port C lines as static outputs. Port C lines intended as static outputs or normal handshake outputs should have their corresponding data direction register bits set, and lines intended as three-state handshake outputs should have their corresponding data direction bits clear.

4.5 PARALLEL I/O CONTROL REGISTER (PIOC)

The parallel handshake I/O functions are available only in the single-chip operating mode. The PIOC is a read/write register except for bit 7 which is read only. Table 4.1 shows a summary of handshake I/O operations.

Table 4.1 Handshake I/O Opertions Summary

	STAI	CWOM	INVB
0	STAF Interrupts Inhibited	Port C Outputs Normal	STRB Active Low
1	STAF Interrupts Enabled	Port C Outputs Open-Drain	STRB Active High

	STAF Clearing Sequence[1]	HNDS	OIN	PLS	EGA	Port C	Port B
Simple Strobe Mode	Read PIOC with STAF = 1 then Read PORTCL	0	×	×		Inputs latched into PORTCL on any active edge on STRA.	STRB pulses on writes to port B.
Full Input Handshake	Read PIOC with STAF = 1 then Read PORTCL	1	0	0 = STRB Active Level 1 = STRB Active Pulse		Inputs latched into PORTCL on any active edge on STRA.	Normal output port. Unaffected in handshake modes.
Full Output Handshake	Read PIOC with STAF = 1 then Write to PORTCL	1	1	0 = STRB Active Level 1 = STRB Active Pulse		Driven as outputs if STRA at active level. Follows DDRC if STRA not at active level.	Normal output port. Unaffected in handshake modes.

Note :

1. Set by active edge on STRA.

	7	6	5	4	3	2	1	0	
$1002	STAF	STAI	CWOM	HNDS	OIN	PLS	EGA	INVB	PIOC
RESET	0	0	0	0	0	U	1	1	

STAF - Strobe A Interrupt Status Flag

This bit is set when a selected edge occurs on strobe A. Clearing it depends on the state of HNDS and OIN bits. In simple strobed mode or in full input handshake mode, STAF is cleared by reading the PIOC register with STAF set followed by reading the PORTCL register. In output handshake, STAF is cleared by reading the PIOC register with STAF set followed by writing to the PORTCL register.

STAI - Strobe A Interrupt Enable Mask

When the 1 bit in the condition code register is clear and STAI is set, STAF (when set) will request an interrupt.

CWOM - Port C Write-OR Mode

CWOM affects all eight port C pins to together

0 = Port C outputs are normal CMOS outputs

1 = Port C outputs act as open-drain outputs

HNDS - Handshake Mode

When clear, strobe A acts as a simple input strobe to latch data into PORTCL, and strobe B acts as a simple output strobe which pulses after a write to port B. When set, a handshake protocol involving port C, STRA, and STRB is selected (see the definition for the ON bit).

0 = Simple strobe mode

1 = Full input or output handshake mode

OIN - Output or Input Handshaking

This bit has no meaning when HNDS = 0.

0 = Input handshake

1 = Output handshake

PLS - Pulse/Interlocked Handshake Operation

This bit has no meaning if HNDS = 0. When interlocked handshake operation is selected, strobe B, once activated, stays active until the selected edge of strobe A is detected. When pulsed handshake operation is selected, strobe B is pulsed for two E cycles.

0 = Interlocked handshake selected

1 = Pulsed handshake selected

EGA - Active Edge for Strobe A

0 = Falling edge of STRA is selected. When output handshake is selected, port C lines obey the data direction register while STRA is low, but port C is forced to output when STRA is high.

1 = Rising edge of STRA is selected. When output handshake is selected, port C lines obey the data direction register while STRA is high, but port C is forced to output when STRA is low.

INVB - Invert Strobe B

0 = Active level is logic zero

1 = Active level is logic one

5. SERIAL COMMUNICATIONS INTERFACE (SCI)

This section contains a description of the serial communication interface (SCI).

5.1 OVERVIEW AND FEATURES

A full-duplex asynchronous Serial Communications Interface (SCI) is provided with a standard NRZ format (one start bit, eight or nine data bits, and one stop bit) and a variety of baud rates. The SCI transmitter and receiver are functionally independent, but use the same data format and bit rate. "Baud" and "bit rate" are used synonymously in the following description.

SCI Two-Wire System Features

- Standard NRZ (mark/space) format.
- Advanced error detection method includes noise detection for noise duration of up to 1/16 bit time.
- Full-duplex operation.
- Software programmable for one of 32 different baud rates.
- Software selectable word length (eight or nine bit words).
- Separate transmitter and receiver enable bits.
- Capable of being interrupt driven.
- Four separate enable bits available for interrupt control.

SCI Receiver Features

- Receiver wake-up function (idle or address bit).
- Idle line detect.
- Framing error detect.
- Noise detect.
- Overrun detect.
- Receiver data register full flag.

SCI Transmitter Features

- Transmit data register empty flag.
- Transmit complete flag.
- Send break.

5.2 DATA FORMAT

Receive data or transmit data is the serial data which is transferred to the internal data bus from the receive data input pin (RxD), or from the internal bus to the transmit data output pin (TxD).

The non-return-to-zero (NRZ) data format shown in Figure 5.1 is used and must meet the following criteria:

(1) The idle line is brought to a logic one state prior to transmission/reception of a character.

(2) A start bit (logic zero) is used to indicate the start of a frame.

(3) The data is transmitted and received least-significant-bit first.

(4) A stop bit (logic one) is used to indicate the end of a frame. A frame consists of a start bit, a character of eight or nine data bits, and a stop bit.

(5) A break is defined as the transmission or reception of a low (logic zero) for at least one complete frame time.

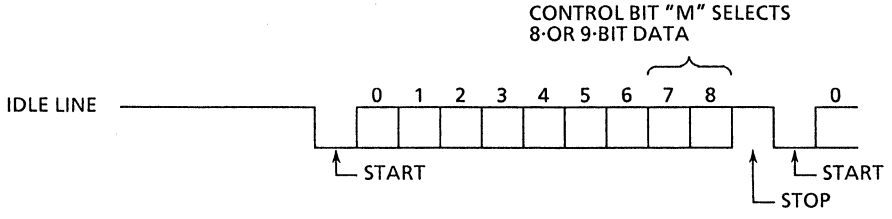


Figure 5.1 Data Format

5.3 WAKE-UP FEATURE

The receiver wake-up feature reduces SCI service overhead in multiple receiver systems. Software in each receiver evaluates the first character(s) of each message. If the message is intended for a different receiver, the SCI can be placed in a sleep mode so that the rest of the message will not generate requests for service. Whenever a new message is started, logic in the sleeping receivers causes them to wake up so they can evaluate the initial character(s) of the new message.

A sleeping SCI receiver can be configured (using the WAKE control bit in serial communications control register 1 (SCCR1)) to wake up using either of two methods: idle line wake up or address mark wake up.

In idle line wake up, a sleeping receiver wakes up as soon as the RxD line becomes idle. Idle is defined as a continuous logic high on the RxD line for ten (or eleven) full bit times. Systems using this type of wake up must provide at least one character time of

idle between messages to wake up sleeping receivers but must not allow any idle time between characters within a message.

In address mark wake up, the most significant bit (MSB) in a character is used to indicate that the character is an address (1) or a data (0) character. Sleeping receivers will wake up whenever an address character is receive. System using this method for wake up would set the MSB of the first character in each message and leave it clear for all other characters in the message. Idle periods may be present within messages and no idle time is required between message for this wake up method.

5.4 RECEIVE DATA (RxD)

Receive data is the serial data which is applied through the input line and the serial communications interface to the internal bus. The receiver circuitry clocks the input at a rate equal to 16 times the baud rate and this time is referred to as the RT clock.

Once a valid start bit is detected, the start bit, each data bit, and the stop bit are sampled three times at RT intervals 8 RT, 9 RT, and 10 RT (1 RT is the position where the bit is expected to start), as shown in Figure 5.2. The value of the bit is determined by voting logic which takes the value of the majority of samples.

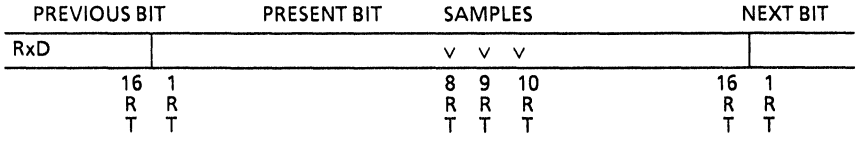


Figure 5.2 Sampling Technique Used on All Bits

5.5 START BIT DETECTION

When the RxD input is detected low, it is tested for three more sample times (referred to as the start edge verification samples in Figure 5.3). If at least two of these three verification samples detect a logic zero, a valid start bit has been detected, otherwise the line is assumed to be idle. A noise flag is set if all three verification samples do not detect a logic zero. A valid start bit could be assumed with a set noise flag present.

If there has been a framing error without detection of a break (10 zeros for 8-bit format or 11 zeros for 9-bit format), the circuit continues to operate as if there actually was a stop bit and the start edge will be placed artificially. The last bit received in the data shift register is inverted to a logic one, and the three logic one start qualifiers (shown in Figure 5.3) are forced into the sample shift register during the interval when detection of a start bit is anticipated (see Figure 5.4); therefore, the start bit will be accepted no sooner than it is anticipated.

If the receiver detects that a break produced the framing error, the start bit will not be artificially induced and the receiver must actually detect a logic one before the start bit can be recognized. See Figure 5.5.

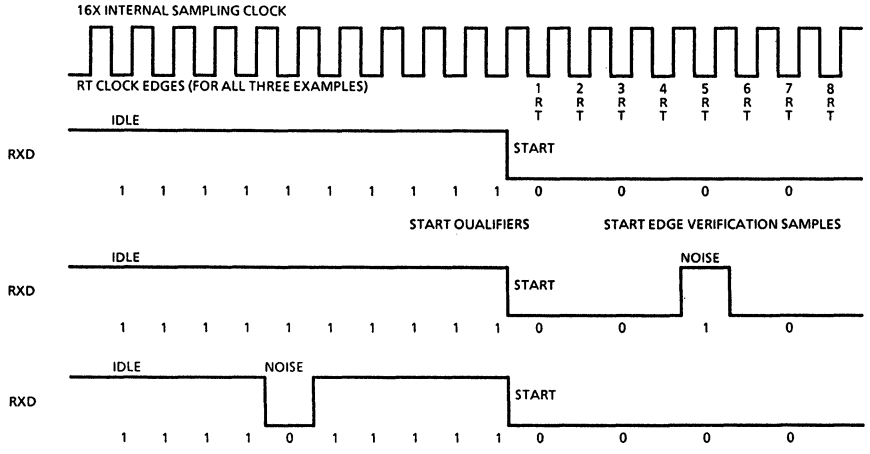
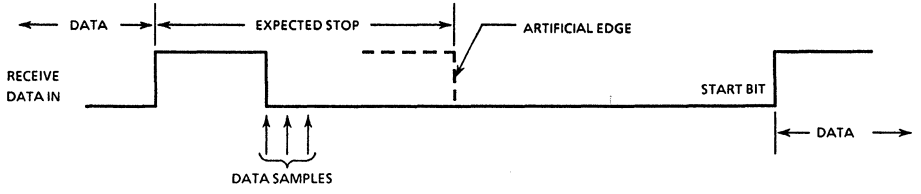
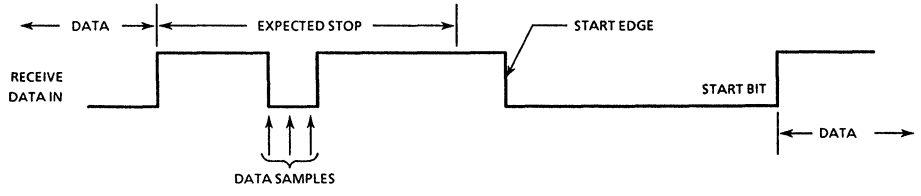


Figure 5.3 Examples of Start Bit Sampling Techniques



(a) Case1, Receive Line Low During Artificial Edge



(b) Case2, Receive Line High During Expected Start Edge

Figure 5.4 SCI Artifical Start Following a Framing Error

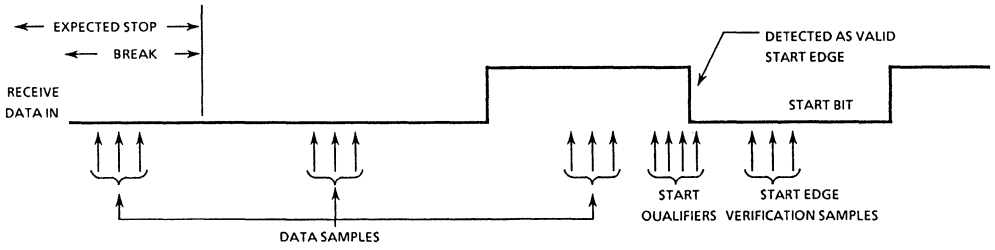


Figure 5.5 SCI Start Bit Following a Break

5.6 TRANSMIT DATA (TxD)

Transmit data is the serial data from the internal data bus which is applied through the serial communications interface to the output line. The transmitter generates a bit time by using a derivative of the RT clock, thus producing a transmission rate equal to 1/16 that of the receiver sample clock.

5.7 FUNCTIONAL DESCRIPTION

A block diagram of the SCI is shown in Figure 5.6. The user has option bits in serial communications control register 1 (SCCR1) to determine the "Wake-up" method (WAKE bit) and data word length (M bit) of the SCI. Serial communications control register 2 (SCCR2) provides control bits which individually enable/disable the transmitter or receiver (TE and RE, respectively), enable system interrupts (TIE, TCIE, ILIE) and provide the wake-up enable bit (RWU) and the send break code bit (SBK). The baud rate register (BAUD) bits allow the user to select different baud rates which may be used as the rate control for the transmitter and receiver.

Data transmission is initiated by a write to the serial communications data register (SCDR). Provided the transmitter is enabled, data stored in the SCDR is transferred to the transmit data shift register. This transfer of data sets the TDRE bit of the SCI status register (SCSR) and may generate an interrupt if the transmit interrupt is enabled. The transfer of data to the transmit data shift register is synchronized with the bit rate clock (Figure 5.7). All data is transmitted LSB first. Upon completion of data transmission, the transmission complete (TC) bit of the SCSR is set (provided no pending data, preamble, or break is to be sent), and an interrupt may be generated if the transmit complete interrupt is enabled. If the transmitter is disabled, and the data, preamble, or break (in the transmit shift register) has been sent, the TC bit will also be set. This will also generate an interrupt if the TCIE bit is set. If the transmitter is disabled in the middle of a transmission, that character will be completed before the transmitter gives up control of the TxD pin.

When the SCDR is read, it contains the last data byte received, provided that the receiver is enabled. The RDRF bit of the SCSR is set to indicate that a data byte has been transferred from the input serial shift register to the SCDR, which can cause an interrupt if the receiver interrupt is enabled. The data transfer from the input serial shift register to the SCDR is synchronized by the receiver bit rate clock. The OR (overrun), NF (noise), FE (framing) error bits of the SCSR may be set if data reception errors occurred.

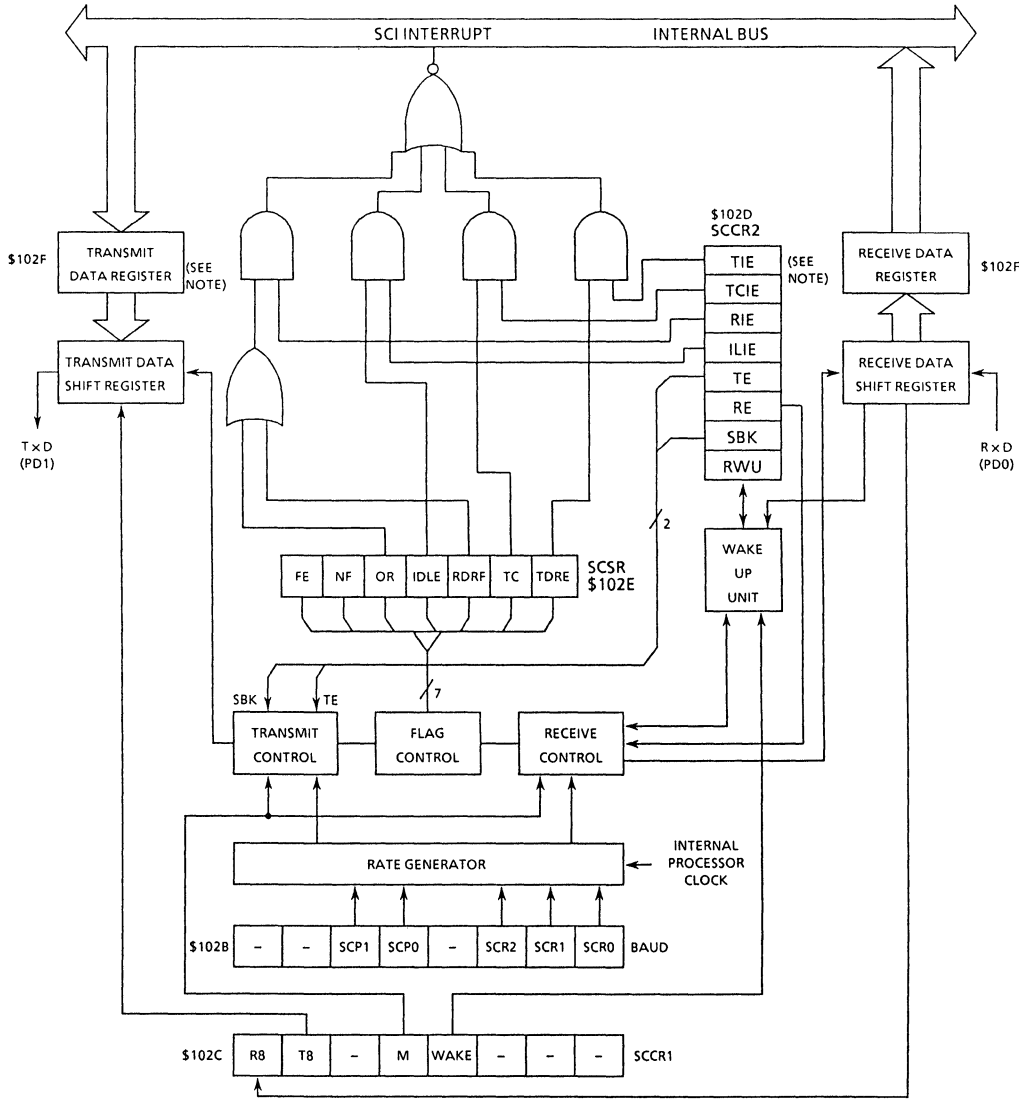
An idle line interrupt is generated if the idle line interrupt is enabled and the IDLE bit (which detects idle line transmission) of SCSR is set. This allows a receiver that is not in the wake-up mode to detect the end of a message, the preamble of a new message, or to resynchronize with the transmitter. A valid character must be received before the idle line condition or the IDLE bit will not be set and an idle interrupt will not be generated.

5.8 SCI REGISTERS

There are five registers used in the serial communications interface and the operation of these registers is discussed in the following paragraphs. Reference should be made to the block diagram shown in Figure 5.6.

5.8.1 Serial Communications Data Register (SCDR)

The serial communications data register performs two functions; i.e., it acts as the receive data register when it is read and as the transmit data register when it is written. Figure 5.6 shows this register as two separate registers, namely:the receive data register and the transmit data register.



Note : The Serial Communications Data Register (SCDR) is controlled by the internal R/$\overline{W}$ signal. It is the transmit data register when witten and receive data register when read.

Figure 5.6 Serial Communications Interface Block Diagram

5.8.2 Serial Communications Control Register 1 (SCCR1)

The serial communications control register 1 (SCCR1) provides the control bits which: (1) determine the word length, and (2) select the method used for the wake-up feature.

	7	6	5	4	3	2	1	0	
$102C	R8	T8	0	M	WAKE	0	0	0	SCCR1
RESET	U	U	0	0	0	0	0	0	

R8 - Receive Data Bit 8

If the M bit is set, this bit provides a storage location for the ninth bit in the receive data character.

T8 - Transmit Data Bit 8

If the M bit is set, this bit provides a storage location for the ninth bit in the transmit data character. It is not necessary to write to this bit for every character transmitted, only when the sense is to be different than that for the previous character.

Bit 5 - Not Implemented

This bit always reads zero.

M - SCI Character Length

0 = 1 start bit, 8 data bits, 1 stop bit

1 = 1 start bit, 9 data bits, 1 stop bit

WAKE - Wake Up Method Select

0 = Idle Line

1 = Address Mark

Bits 2-1 - Not Implemented

These bits always read zero.

5.8.3 Serial Communications Control Register 2 (SCCR2)

The serial communications control register 2 (SCCR2) provides the control bits which enable/disable individual SCI functions.

	7	6	5	4	3	2	1	0	
$102D	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK	SCCR2
RESET	0	0	0	0	0	0	0	0	

TIE - Transmit Interrupt Enable

0 = TDRE interrupts disabled

1 = SCI interrupt if TDRE = 1

TCIE - Transmit Complete Interrupt Enable

0 = TC interrupts disabled

1 = SCI interrupt if TC = 1

REI - Receive Interrupt Enable

0 = RDRF and OR interrupts disabled

1 = SCI interrupt if RDRF or OR = 1

ILIE - Idle Line Interrupt Enable

0 = IDLE interrupts disabled

1 = SCI interrupt if IDLE = 1

TE - Transmit Enable

When the transmit enable (TE) bit is set, the transmit shift register output is applied to the TxD line. Depending on the state of control bit M (SCCR1), a preamble of 10 (M = 0) or 11(M = 1) consecutive ones is transmitted when software sets the TE bit from a cleared state. After loading the last byte in the serial communications data register and receiving the TDRE flag, the user can clear TE bit. Transmission of the last byte will then be completed before the transmitter gives up control of the TxD pin. While the transmitter is active, the data direction register control for port D bit 1 is overridden and the line is forced to be an output.

RE - Receive Enable

When the receive enable (RE) bit is set, the receiver is enabled. When RE bit is clear, the receiver is disabled and all of the status bits associated with the receiver (RDRF, IDLE, OR, NF and FE) are inhibited. While the receiver is enabled, the data direction register control for port D bit 0 is overridden and the line is forced to be an input.

RWU - Receiver Wake UP

When the receiver wake-up (RWU) bit is set by the user's software, it puts the receiver to sleep and enables the "wake up" function. If the WAKE bit is cleared, RWU bit is cleared by the SCI logic after receiving 10 (M=0) or 11 (M=1) consecutive ones. If the WAKE bit is set, RWU bit is cleared by the SCI logic after receiving a data word whose MSB is set.

SBK - Send Break

If the send break (SBK) bit is toggled and cleared, the transmitter sends 10 (M=0) or 11 (M=1) zeros and then reverts to idle or sending data. If SBK bit remains set, the transmitter will continually send whole blocks of zeros (sets of 10 or 11) until cleared. At the completion of the break code, the transmitter sends at least one high bit to guarantee recognition of a valid start bit. If the transmitter is currently empty and idle, setting and clearing SBK bit is likely to queue two character times of break because the first break transfers almost immediatly to the shift register and the second is then queued into the parallel transmit buffer.

5.8.4 Serial Communications Status Register (SCSR)

The serial communications status register (SCSR) provides inputs to the interrupt logic circuits for generation of the SCI system interrupt.

	7	6	5	4	3	2	1	0	
$102E	TDRE	TC	RDRF	IDLE	OR	NF	FE	0	SCSR
RESET	0	0	0	0	0	0	0	0	

TDRE - Transmit Data Register Empty

The transmit data register empty (TDRE) bit is set to indicate that the content of the serial communications data register have been transferred to the transmit serial shift register. This bit is cleared by reading the SCSR register (with TDRE=1) followed by a write to the SCDR register.

TC - Transmit Complete

The transmit complete (TC) bit is set at the end of a data frame, preamble, or break condition if:

(1) TE=1, TDRE=1, and no pending data, preamble, or break is to be transmitter; or

(2) TE=0, and the data, preamble, or break in the transmit shift register has been transmitted.

The TC bit is a status flag which indicates that one of the above conditions have occurred.

The TC bit is cleared by reading the SCSR register (with TC set) followed by a write to the SCDR.

RDRF - Receive Data Register Full

The receive data register full (RDRF) bit is set when the receiver serial shift register is transferred to the SCDR register. The RDRF bit is cleared when the SCSR register is read (with RDRF set) followed by a read of the SCDR register.

IDLE - Idle Line Detect

The idle line detect (IDLE) bit, when set, indicates the receiver has detected an idle line. The IDLE bit is cleared by reading the SCSR register with IDLE bit set followed by reading SCDR register. Once the IDLE status flag is cleared, it will not be set again until after the RxD line has been active and becomes idle again.

OR - Overrun Error

The overrun error (OR) bit is set when the next byte is ready to be transferred from the receive shift register to the SCDR register which is already full (RDRF bit is set). When an overrun error occurs, the data which caused the overrun is lost and the data which was already in SCDR register is not disturbed. The OR bit is cleared when the SCSR register is read (with OR bit set), followed by a read of the SCDR register.

NF - Noise Flag

The noise flag (NF) bit is set if there is noise on any of the received bits, including the start and stop bits. The NF bit is not set until the RDRF flag is set. The NF bit is cleared when the SCSR register is read (with NF set), followed by a read of the SCDR register.

FE - Framing Error

The framing error (FE) bit is set when no stop bit was detected in the received data character. The FE bit is set at the same time as the RDRF bit is set. If the byte received causes both framing and overrun errors, the processor will only recognize the overrun error. The framing error flag inhibits further transfer of data into the SCDR register until it is cleared. The FE bit is cleared when the SCSR register is read (with FE bit equal to one) followed by a read of the SCDR register.

Bit 0 - Not Implemented

This bit always reads zero.

5.8.5 Baud Rate Register (BAUD)

The baud rate register selects the different baud rates which may be used as the rate control for the transmitter and receiver. The SCP0-SCP1 bits functions as a prescaler for the SCR0-SCR2 bits. Together, these five bits provide multiple baud rate combinations for a given crystal frequency.

	7	6	5	4	3	2	1	0	
$102B	TCLR	0	SCP1	SCP0	RCKB	SCR2	SCR1	SCR0	BAUD
RESET	0	0	0	0	0	U	U	U	

TCLR-Clear Baud Rate Counters (Test)

This bit is used to clear the baud rate counter chain during factory testing. TCLR bit is zero and cannot be set while in normal operating modes.

SCP1 and SCP0-SCI Baud Rate Prescaler Selects

The E clock is divided by the factors shown in Table 5.1. This prescaled output provides an input to a divider which is controlled by the SCR2-SCR0 bits.

Table5.1 Second Prescaler Stage

SCR1	SCR0	Internal Processor Clock Divide By
0	0	1
0	1	3
1	0	4
1	1	13

SCR2, SCR1, and SCR0-SCI Baud Rate Selects

These three bits select the baud rates for both the transmitter and the receiver. The prescaler output described above is further divided by the factors shown in Table 5.2.

Table 5.2 Second Prescaler Stage

SCR2	SCR1	SCR0	Prescaler output Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

RCKB-SCI Baud Rate Clock Check (Test)

This bit is used during factory testing to enable the exclusive-OR of the receiver clock and transmitter clock to be driven out the TxD pin. RCKB bit is zero and cannot be set while in normal operating modes.

The diagram shown in Figure 5.7 and the data given in Tables 5.3 and 5.4 illustrate the divider chain used to obtain the baud rate clock. Note that there is a fixed rate divide-by-16 between the receive clock (RT) and the transmit clock (Tx). The actual divider chain is controlled by the combined SCP0-SCP1 and SCR0-SCR2 bits in the baud rate register sa illustrated.

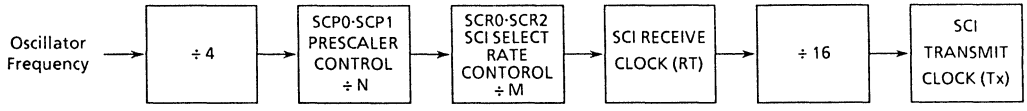


Figure 5.7 Rate Generator Division

Table 5.3 Prescaler Highest Baud Rate Frequency Output

SCP Bit		Clock* Divided By	Crystal Frequency (MHz)				
1	0		8.3886	8.0	4.9152	4.0	3.6864
0	0	1	131.072 K Baud	125.000 K Baud	76.80 K Baud	62.50 K Baud	57.60 K Baud
0	1	3	43.691 K Baud	41.666 K Baud	25.60 K Baud	20.833 K Baud	19.20 K Baud
1	0	4	32.768 K Baud	31.250 K Baud	19.20 K Baud	15.625 K Baud	14.40 K Baud
1	1	13	10.082 K Baud	9600 Baud	5.907 K Baud	4800 Baud	4430 Baud

* The clock in the "Clock Divided By" column is the internal processor clock.

Note : The divided frequencies shown in Table5.3 represent baud rates which are the highest transmit baud rate (Tx) that can be obtained by a specific crystal frequency and only using the prescaler division. Lower baud rates may be obtained by providing a further division using the SCI rate select bits as shown below for some representative prescaler outputs.

Table 5.4 Transmit Baud Rate Output for a Given Prescaler Output

SCR Bits			Divided By	Representative Highest Prescaler Baud Rate Output				
2	1	0		131.072 K Baud	32.768 K Baud	76.80 K Baud	19.20 K Baud	9600 Baud
0	0	0	1	131.072 K Baud	32.768 K Baud	76.80 K Baud	19.20 K Baud	9600 Baud
0	0	1	2	65.536 K Baud	16.384 K Baud	38.40 K Baud	9600 Baud	4800 Baud
0	1	0	4	32.768 K Baud	8.192 K Baud	19.20 K Baud	4800 Baud	2400 Baud
0	1	1	8	16.384 K Baud	4.096 K Baud	9600 Baud	2400 Baud	1200 Baud
1	0	0	16	8.192 K Baud	2.048 K Baud	4800 Baud	1200 Baud	600 Baud
1	0	1	32	4.096 K Baud	1.024 K Baud	2400 Baud	600 Baud	300 Baud
1	1	0	64	2.048 K Baud	512 Baud	1200 Baud	300 Baud	150 Baud
1	1	1	128	1.024 K Baud	256 Baud	600 Baud	150 Baud	75 Baud

Note : Table 5.4 illustrates how the SCI select bits can be used to provide lower transmitter baud rates by further dividing the prescaler output frequency. The five example are only representative samples. In all cases, the baud rates shown are transmit baud rates (transmit clock) and the receiver clock is 16 times higher in frequency than the actual baud rate.

6. SERIAL PERIPHERAL INTERFACE (SPI)

This section contains a description on the serial peripheral interface (SPI).

6.1 OVERVIEW AND FEATURES

The serial peripheral interface (SPI) is a synchronous interface which allows several SPI microcontrollers or SPI-type peripherals to be interconnected. 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. The TMP68HC11A8 SPI system may be configured either as a master or as a slave.

Features 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
- Easily Interfaces to Simple Expansion Parts (PLLs, D/As, Latches, Display Drivers, etc.)

6.2 SPI SIGNAL DESCRIPTIONS

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

Any SPI output line has to have its corresponding data direction register bit set. If this bit is clear, the line is disconnected from the SPI logic and becomes a general-purpose input line. Any SPI input line is forced to act as an input regardless of what is in the corresponding data direction register bit.

6.2.1 Master In Slave Out (MISO)

The MISO line is configured as an input in a master device and as an output in a slave device. It is one of the two lines that transfer serial data in one direction, with the most significant bit sent first. The MISO line of a slave device is placed in the high-impedance state if the slave is not selected.

6.2.2 Master Out Slave In (MOSI)

The MOSI line is configured as an output in a master device and an input in a slave device. It is one of the two lines that transfer serial data in one direction with the most significant bit sent first.

6.2.3 Serial Clock (SCK)

The serial clock is used to synchronize data movement both in and out of the device through its MOSI and MISO lines. The master and slave devices are capable of exchanging a byte of information during a sequence of eight clock cycles. Since SCK is generated by the master device, this line becomes an input on a slave device.

As shown in Figure 6.1, four possible timing relationships may be chosen by using control bits CPOL and CPHA in the serial peripheral control register (SPCR). Both master and slave devices must operation with the same timing. The master device always places data on the MOSI line a half-cycle before the clock edge (SCK), in order for the slave device to latch the data.

Two bits (SPR0 and SPR1) in the SPCR of the master device select the clock rate. In a slave device, SPR0 and SPR1 have no effect on the operation on the SPI.

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

The slave select ($\overline{SS}$) input line is used to select a slave device. It has to be low prior to data transactions and must stay low for the duration of transaction.

The $\overline{SS}$ line on the master must be tied high. If it goes low, a mode fault error flag (MODF) is set in the serial peripheral status register (SPSR). The $\overline{SS}$ pin can be selected to be a general-purpose output by writing a one in bit 5 of the port D data direction register, thus disabling the mode fault circuit. The other three SPI lines are dedicated to the SPI whenever the SPI is on.

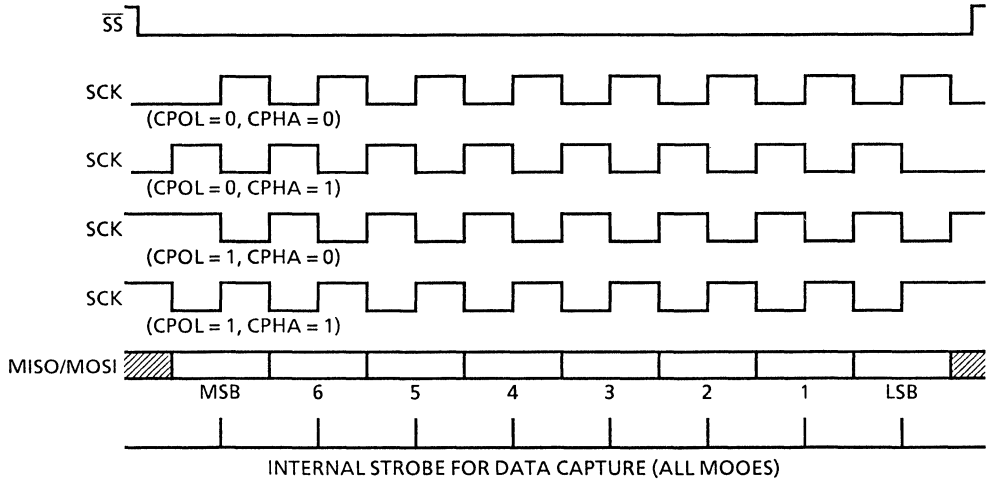


Figure 6.1 Data Clock Timing Diagram

When CPHA = 0, the shift clock is the OR of $\overline{SS}$ with SCK. In this clock phase mode, $\overline{SS}$ must go high between successive characters in an SPI message. When CPHA = 1, $\overline{SS}$ may be left low for several SPI characters. In cases where there is only one SPI slave MCU, its $\overline{SS}$ line could be tied to V_{SS} as long as CPHA = 1 clock modes are used.

6.3 FUNCTIONAL DESCRIPTION

Figure 6.2 shows a block diagram of the serial peripheral interface circuitry. When a master device transmits data to a slave device via the MISO line, the slave device responds by sending data to the master device via the master's MISO line. This implies full duplex transmission with both data out and data in synchronized with the same clock signal. 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) is used to signify that the I/O operation has been completed.

The SPI is double buffered on read, but not on write. If a write is performed during data transfer, the transfer occurs uninterrupted, and the write will be unsuccessful. This condition will cause the write collision (WCOL) status bit in the SPSR to be set. After a data byte is shifted, the SPIF flag of the SPSR is set.

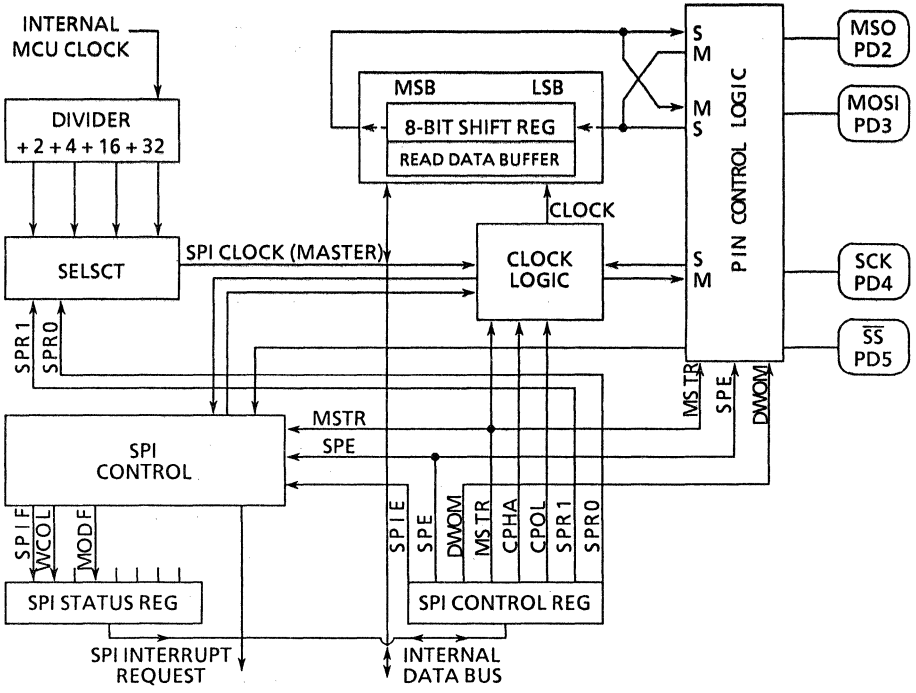


Figure 6.2 Serial Peripheral Interface Block Diagram

In the master mode, the SCK pin is an output. It idles high or low, depending on the CPOL bit in the SPCR, until data is written to the shift register, at which point eight clocks are generated to shift the eight bits of data and then SCK goes idle again.

In a slave mode, the slave start logic receives a logic low at the $\overline{SS}$ pin and a clock input at the SCK pin. Thus, the slave is synchronized with the master. Data from the master is received serially at the slave MOSI line and loads the 8-bit shift register. After the 8-bit shift register is loaded, its data is parallel transferred to the read buffer. During a write cycle, data is written into the shift register, then the slave waits for a clock train from the master to shift the data out on the slave's MISO line.

Figure 6.3 illustrates the MOSI, MISO, SCK and $\overline{SS}$ master-slave interconnections.

Due to data direction register control of SPI outputs and the port D write-OR mode (DWOM) option, the SPI system can be configured in a variety of ways. System with a single bidirectional data path rather than seperate MISO and MOSI paths can be accommodated. Since TMP68HC11A8 SPI slaves can selectively disable their MISO output, a broadcast message protocol is also possible.

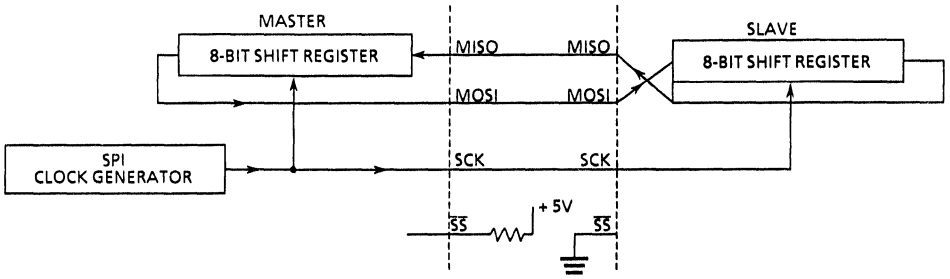


Figure 6.3 Serial Peripheral Interface Master-Slave Interconnection

6.4 SPI REGISTERS

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

6.4.1 Serial Peripheral Control Register (SPCR)

	7	6	5	4	3	2	1	0	
$1028	SPIE	SPE	DWOM	MSTR	CPOL	CPHA	SPR1	SPR0	SPCR
RESET	0	0	0	0	0	1	U	U	

SPIE - Serial Peripheral Interrupt Enable

0 = SPIF interrupts disabled

1 = SPI interrupt if SPIF = 1

SPE - Serial Peripheral System Enable

0 = SPI system off

1 = SPI system on

DWOM - Port D Write-OR Mode Option

DWOM affects all six port D pins together.

0 = Port D outputs are normal CMOS outputs

1 = Port D outputs act as open-drain outputs

MSTR - Master Mode Select

0 = Slave mode

1 = Master mode

CPOL - Clock Polarity

When the clock polarity (CPOL) bit is cleared and data is not being transferred, a steady state low value is produced at the SCK pin of the master device. Conversely, if this bit is set, the SCK pin will idle high. This bit is also used in conjunction with the clock phase control bit to produce the desired clock-data relationship between master and slave. See Figure 6.1.

CPHA - Clock Phase

The clock phase (CPHA) bit, in conjunction with the CPOL bit, controls the clock-data relationship between master and slave. The CPOL bit can be thought of as simply inserting an inverter in series with the SCK line. The CPHA bit selects one of two fundamentally different clocking protocols. When CPHA = 0, the shift clock is the OR of SCK with $\overline{SS}$. As soon as $\overline{SS}$ goes low the transaction begins and the first edge on SCK invokes the first data sample. When CPHA = 1, the $\overline{SS}$ pin may be thought of as a simple output enable control. Refer to Figure 6.1.

SPR1 and SPR0-SPI Clock Rate Selects

These two serial peripheral rate bits (SPR1, SPR2) select one of four baud rates (Table 6.1) to be used as SCK if the device is a master;however, they have no effect in the slave mode.

Table 6.1 Serial Peripheral Rate Selection

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

6.4.2 Serial Peripheral Status Register (SPSR)

	7	6	5	4	3	2	1	0	
$1029	SPIF	WCOL	0	MODF	0	0	0	0	SPSR
RESET	0	0	0	0	0	0	0	0	

SPIF - SPI Transfer Complete Flag

The serial peripheral data transfer flag (SPIF) bit is set upon completion of data transfer between the processor and external device. If SPIF bit goes high, and if SPIE bit is set, a serial peripheral interrupt is generated. Clearing the SPIF bit is accomplished by reading the SPSR register (with SPIF set) followed by an access of the SPDR register. Unless SPSR register is read (with SPIF set) first, attempts to write to SPDR register are inhibited.

WCOL - Write Collision

The write collision bit (WCOL) is set when an attempt is made to write to the serial peripheral data register while data transfer is taking place. If CPHA bit is zero a transfer is said to begin when $\overline{SS}$ goes low and the transfer ends when $\overline{SS}$ goes high after eight clock cycles on SCK. When CPHA bit is one a transfer is said to begin the first time SCK becomes active while $\overline{SS}$ is low and the transfer ends when the SPIF flag gets set. Clearing the WCOL bit is accomplished by reading the SPSR register (with WCOL set) followed by an access to SPDR register.

Bit 5 - Not Implemented

This bit always reads zero.

MODF - Mode Fault

The mode fault (MODF) flag indicates that there may 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 normally clear, and is set only when the master device has its $\overline{SS}$ pin pulled low. Setting the MODF bit affects the internal serial peripheral interface system in the following ways.

1) An SPI interrupt is generated if SPIE = 1.

2) The SPE bit is cleared. This disables the SPI.

3) The MSTR bit is cleared, thus forcing the device into the slave mode.

4) DDRD bits for the four SPI pins are forced to zeros.

Clearing the MODF bit is accomplished by reading the SPSR register (with MODF set), followed by a write to the SPCR register. Control bits SPE and MSTR may be restored by user software to their original state after the MODF bit has been cleared. It is also necessary to restore DDRD bit after a mode fault.

Bits 3-0 - Not Implemented

These bits always read zero.

6.4.3 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. At the completion of transmitting a byte of data, the SPIF status bit is set in both the master and slave devices.

When the user reads the serial peripheral data I/O register, a buffer is actually being read. The first SPIF must 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. In cases of overrun the byte which causes the overrun is lost.

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

7. ANALOG-TO-DIGITAL CONVERTER

The TMP68HC11A8 includes an 8-channel, multiplexed-input, successive approximation, analog-to-digital (A/D) converter with sample and hold to minimize conversion errors caused by rapidly changing input signals. Two dedicated lines (V_{RL}, V_{RH}) are provided for the reference voltage inputs. These pins may be connected to a separate or isolated power supply to ensure full accuracy of the A/D conversion. The 8-bit A/D converter has a total error of ± 1 LSB which includes $\pm 1/2$ LSB of quantization error and accepts analog inputs which range from V_{RL} to V_{RH}. Smaller analog input ranges can also be obtained by adjusting V_{RH} and V_{RL} to the desired upper and lower limits. Conversion is specified and tested for $V_{RL}=0V$ and $V_{RH}=5V \pm 10\%$; however, laboratory characterization over the full temperature range indicates little or no degradation with V_{RH}-V_{RL} as low as 2.5 to 3V. The A/D system can be operated with V_{RH} below V_{DD} and/or V_{RL} above V_{SS} as long as V_{RH} is above V_{RL} by enough to support the conversions (2.5 to 5.0V). Each conversion is accomplished in 32 MCU E clock cycles, provided the E clock rate is greater than 750 kHz. For systems which operate at clock rates less than 750 kHz, an internal R-C oscillator must be used to clock the A/D system. The internal R-C oscillator is selected by setting the CSEL bit in the OPTION register.

Note : Only four A/D input channels are available in the 48-pin version.

7.1 CONVERSION PROCESS

The A/D converter is ratiometric. An input voltage equal to V_{RL} converts to $00 and an input voltage equal to V_{RH} converts to $FF (full scale), with no overflow indication. For ratiometric conversions, the source of each analog input should use V_{RH} as the supply voltage and be referenced to V_{RL}.

Figure 7.1 shows the detailed sequence for a set of four conversions. This sequence begins one E clock cycle after a write to the A/D control/status register (ADCTL). Figure 7.2 shows a model of the port E A/D channel inputs. This model is useful for understanding the effects of external circuitry on the accuracy of A/D conversions.

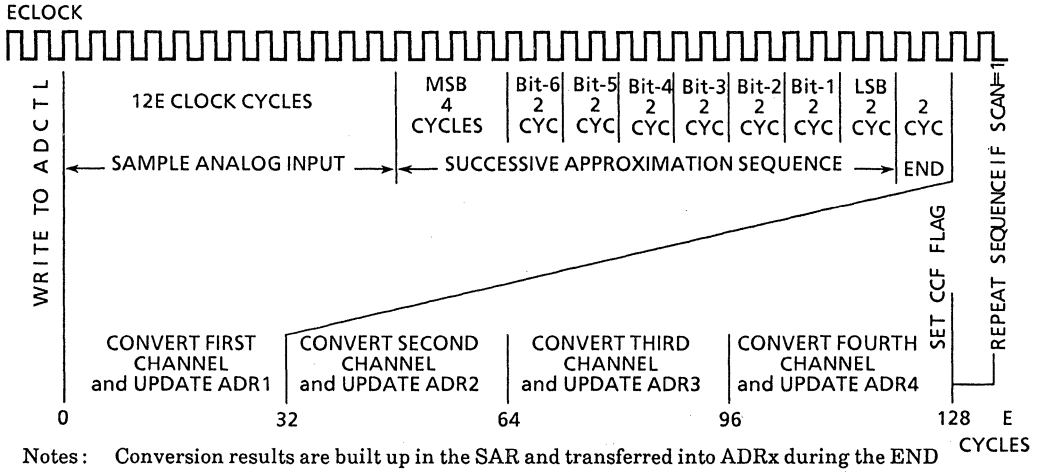
7.2 CHANNEL ASSIGNMENTS

A multiplexer allows the single A/D converter to select one of sixteen analog signals. Eight of these channels correspond to port E input lines to the MCU, four of the channels are for internal reference points or test functions, and four channels are reserved for future use. Table 7.1 shows the signals selected by the four channel select control bits.

7.3 SINGLE-CHANNEL OPERATION

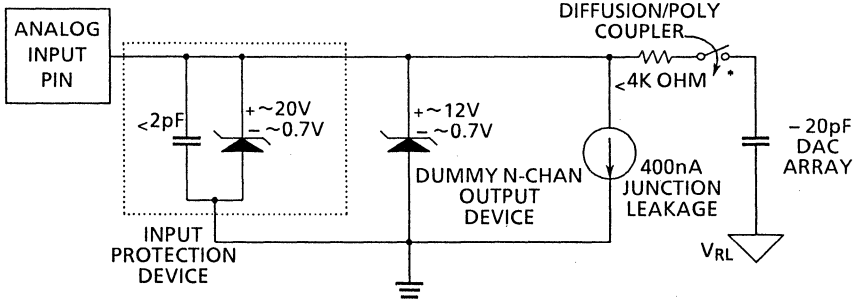
There are two variations of single-channel operation. In the first variation (SCAN = 0), the single selected channel is converted four consecutive times with the first result being stored in A/D result register 1 (ADR1) and the fourth result being stored in register ADR4. After the fourth conversion is complete, all conversion activity is halted

until a new conversion command is written to the ADCTL register. In the second variation (SCAN = 1), conversions continue to be performed on the selected channel with the fifth conversion being stored in register ADR1 (overwriting the first conversion result), the sixth conversion overwrites ADR2, and so on.



Notes: Conversion results are built up in the SAR and transferred into ADRx during the END period. The CCF status flag is set during the END period of the fourth conversion after a write to ADCTL. This figure assumes CSEL in the OPTION register is 0 so the E clock is acling as the conversion clock. If MULT = 0 all four conversions in the sequence are performed on the same analog channel.

Figure 7.1 A/D Conversion Sequence



* This analog switch is closed only during the 12 cycle sample time

Figure 7.2 A/D Pin Model

7.4 MULTIPLE-CHANNEL OPERATION

There are two variations in multiple-channel operation. In the first variation (SCAN = 0), the selected group of four channels are converted, one time each, with the

first result being stored in register ADR1 and the fourth result being stored in register ADR4. After the fourth conversion is complete,all conversion activity is halted until a new conversion command is written to the ADCTL register. In the second variation (SCAN = 1), conversions continue to be performed on the selected group of channels with the fifth conversion being stored in register ADR1 (replacing the earlier conversion result for the first channel in the group), the sixth conversion overwrites ADR2, and so on.

7.5 OPERATION IN STOP AND WAIT MODES

If a conversion sequence is still in process when either the STOP or WAIT mode is enered, the conversion of the current channel is suspended. When the MCU resumes normal operation, that channel will be re-sampled and the conversion sequence resumed. As the MCU exits the WAIT mode, the A/D circuits are stable and valid results can be obtained on the first conversion. However, in STOP mode, all analog bias currents are disabled and it becomes necessary to allow a stabilization period when leaving the STOP mode. If the STOP mode is exited with a delay, there will be enough time for these circuits to stabilize before the first conversion. If the STOP mode is exited with no delay (DLY bit in OPTION register equal to zero), sufficient time must be allowed for the A/D circuitry to stabilize to avoid invalid results (see 7.8 A/D POWER UP AND CLOCK SELECT).

7.6 A/D CONTROL/STATUS REGISTER (ADCTL)

All bits in this register may be read or written, except bit 7 which is a ready-only status indicator and bit 6 which always reads as a zero.

	7	6	5	4	3	2	1	0	
$1030	CCF	0	SCAN	MULT	CD	CC	CB	CA	ADCTL
RESET	0	0	U	U	U	U	U	U	

CCF - Conversions Complete Flag

This read-only status indicator is set when all four A/D result registers contain valid conversion results. Each time the ADCTL register is written, this bit is automatically cleared to zero and a conversion sequence is started. In the continuous modes, conversions continue in a round-robin fashion and the result registers continue to be updated with current data even though the CCF bit remains set.

Note : The user must write to register ADCTL to initiate conversion. To abort a conversion in progress, write to the ADCTL register and a new conversion sequence is initiated immediately.

Bit 6 - Not Implemented

This bit always reads zero.

SCAN - Continuous Scan Control

When this control bit is clear, the four requested conversions are performed once to fill the four result registers. When this control bit is set, conversions continue in a round-robin fashion with the result registers being updated as data becomes available.

MULT - Multiple-Channel/Single Channel Control

When this bit is clear, the A/D system is configured to perform four consecutive conversions on the single channel specified by the four channel select bits CD through CA (bits 3-0 of the ADCTL register). When this bit is set, the A/D system is configured to perform a conversion on each of four channels where each result register corresponds to one channel.

Note : When the multiple channel continuous scan mode is used, extra care is needed in the design of circuitry driving the A/D inputs. Refer to the A/D Pin Model and A/D Conversion Sequence figures in addition to the following discussion. The charge on the capacitive DAC array prior to the sample time is related to the voltage on the previously converted channel. A charge share situation exists between the internal DAC capacitance and the external circuit capacitance. Although the amount of charge involved is small the rate at which it is repeated is every 64 microseconds for an E clock of 2 MHz. The RC charging rate of the external circuit must be balanced against this charge sharing effect to avoid accuracy errors.

CD - Channel Select D

CC - Channel Select C

CB - Channel Select B

CA - Channel Select A

These four bits are used to select one of 16 A/D channels (see Table 7.1). When a multiple channel mode is selected (MULT=1), the two least-significant channel select bits (CB and CA) have no meaning and the CD and CC bits specify which group of four channels are to be converted. The channels selected by the four channel select control bits are shown in Table 7.1.

Table 7.1 Analog-to-Digital Channel Assignments

CD	CC	CB	CA	Channel Signal	Result in ADRx if MULT = 1
0	0	0	0	AN0	ADR1
0	0	0	1	AN1	ADR2
0	0	1	0	AN2	ADR3
0	0	1	1	AN3	ADR4
0	1	0	0	AN4*	ADR1
0	1	0	1	AN5*	ADR2
0	1	1	0	AN6*	ADR3
0	1	1	1	AN7*	ADR4
1	0	0	0	Reserved	ADR1
1	0	0	1	Reserved	ADR2
1	0	1	0	Reserved	ADR3
1	0	1	1	Reserved	ADR4
1	1	0	0	V_{RH} Pin**	ADR1
1	1	0	1	V_{RL} Pin**	ADR2
1	1	1	0	$(V_{RH})/2$**	ADR3
1	1	1	1	Reserved**	ADR4

*Not available in 48-pin package versions.
**This group of channels used during factory test.

7.7 A/D RESULT REGISTERS 1, 2, 3, AND 4 (ADR1, ADR2, ADR3, and ADR4)

The A/D result registers are ready-only registers used to hold an 8-bit conversion result. Writes to these registers have no effect. Data in the A/D result registers is valid when the CCF flag bit in the ADCTL register is set, indicating a conversion sequence is complete. If conversion results are needed sooner refer to Figure 7.1. For example the ADR1 result is valid 33 cycles after an ADCTL write. Refer to the A/D channelassignments in Table 7.1 for the relationship between the channels and the result registers.

7.8 A/D POWER UP AND CLOCK SELECT

A/D power up is controlled by bit 7 (ADPU) of the OPTION register. When ADPU is cleared, power to the A/D system is disabled. When ADPU is set, the A/D system is enabled. A delay of as much as 100 microseconds is required after turning on the A/D converter to allow the analog bias voltages to stabilize.

Clock select is controlled by bit 6 (CSEL) of the OPTION register. When CSEL is cleared, the A/D system uses the system E clock. When CSEL is set, the A/D system uses an internal R-C clock source, which runs at about 1.5MHz. The MCU E clock is not suitable to drive the A/D system if it is operating below 750kHz, in which case the R-C internal clock should be selected. A delay of 10 ms is required after changing CSEL from zero to one to allow the R-C oscillator to start and internal bias voltages to settle. Refer to 9.1.5 Configuration Options Register (OPTION) for additional information. Note that

the CSEL control bit also enables a separate R-C oscillator to drive the EEPROM charge pump.

When the A/D system is operating with the MCU E clock, all switching and comparator operations are synchronized to the MCU clocks. This allows the comparator results to be sampled at quiet clock times to minimize noise errors. The internal R-C oscillator is asynchronous to the MCU clock so noise will affect A/D results more while CSEL=1.

8. PROGRAMMABLE TIMER, REAL TIME INTERRUPT, AND PULSE ACCUMULATOR

This section describes the 16-bit programmable timer, the real time interrupt, and the pulse accumulator system.

8.1 PROGRAMMABLE TIMER

The timer has a single 16-bit free-running counter which is clocked by the output of a four-stage prescaler (divide by 1, 4, 8, or 16), which is in turn driven by the MCU E clock. Input functions are called input captures. These input captures record the count form the free-running counter in response to a detected edge on an input line. Output functions, called output compares, cause an output action when there is a match between a 16-bit output-compare register and the free-running counter. This timer system has three input capture registers and five output compare registers.

8.1.1 Counter

The key element in the timer system is a 16-bit free-running counter, or timer counter register. After reset, the MCU is configured to use the E clock as the input to the free-running counter. Initialization software may optionally reconfigure the system to use one of the three prescaler values. The prescaler control bits can only be written once during the first 64 cycles after a reset. Software can read the counter at any time without affecting its value because it is clocked and read during opposite phases of the E clock.

A counter read should first address the most significant byte. An MPU read of this address causes the least significant byte to be transferred to a buffer. This buffer is not affected by reset and is accessed when reading the least significant byte of the counter. For double byte read instructions, the two accesses occur on consecutive bus cycles.

The counter is cleared to $0000 during reset and is a read-only register with one exception. In test modes only, any MPU write to the most significant byte presets the counter to $FFF8 regardless of the value involved in the write.

When the count changes from $FFFF to $0000, the timer overflow flag (TOF) bit is set in timer interrupt flag register 2 (TFLG2). An interrupt can be enabled by setting the interrupt enable bit (TOI) in timer interrupt mask register 2 (TMSK2).

8.1.2 Input Capture

The input capture registers are 16-bit read-only registers which are not affected by reset and are used to latch the value of the counter when a defined transition is sensed by the corresponding input capture edge detector. The level transition which triggers counter transfer is defined by the corresponding input edge bits (EDGxB, EDGxA) in TCTL2.

The result obtained by an input capture corresponds to the value of the counter one E clock cycle after the transition which triggered the edge-detection logic. The selected edge transition sets the ICxF bit in timer interrupt flag register 1 (TFLG1) and can cause an interrupt if the corresponding ICxI bit(s) is (are) set in the timer interrupt mask register 1 (TMSK1). A read of the input capture register's most significant byte inhibits captures for one E cycle to allow a double-byte read of the full 16-bit register.

8.1.3 Output Compare

All output compare registers are 16-bit read/write registers which are initialized to SFFFF by reset. They can be used as output waveform controls or as elapsed time indicators. If an output compare register is not used, it may be used as a storage location.

All output compare registers have a separate dedicated comparator for comparing against the free-running counter. If a match is found, the corresponding output compare flag (OCxF) bit in TFLG1 is set and a specified action is automatically taken. For output compare functions two through five the automatic action is controlled by pairs of bits (OMx and OLx) in the timer control register 1 (TCTL1). Each pair of control bits are encoded to specify the output action to be taken as a result of a successful OCX compare. The output adition is taken on each succesful compare regardless of wheather or not the OCxF flag was previously clear.

An interrupt can also accompany a successful output compare, provided that the corresponding interrupt enable bit (OCxI), is set in TMSK1.

After an MPU write cycle to the most significant byte, output compares are inhibited for one E cycle in order to allow writing two consecutive bytes before making the next comparison. If both bytes of the register are to be changed, a double-byte write instruction should be used in order to take advantage of the compare inhibit feature.

MPU write can be made to either byte of the output compare register without affecting the other byte.

A write-only register, timer compare force (CFORC), allows forced compares. Five of the bit positions in the CFORC register correspond to the five output compares. To force a compare, or compares, a write is done to CFORC register withe the associated bits set for each output compare that is to be forced. The action taken as a result of a forced compare is the same as if there was a match between the OCx register and the free-running counter, except that the corresponding interrupt status flag bits are not set. Output actions are synchronized to the prescaled timer clock so there could be as much as 16 E clock cycles of delay between the write to CFORC and the output action.

8.1.4 Output Compare 1 I/O Pin Control

Unlike the other four output compares, output compare 1 can automatically affect any or all of the five output pins (bits 3-7) in port A as a result of a successful compare between the OC1 register and the 16-bit free-running counter. The two 5-bit registers used in conjunction with this function are the output compare 1 mask register (OC1M) and the output compare 1 data register (OC1D).

Register OC1M is used to specify the bits of port A (I/O and timer port) which are to be affected as a result of a successful OC1 compare. Register OC1D is used to specify the data which is to be stored to the affected bits of port A as the result of a successful OC1 compare. If an OC1 compare and another output compare occur during the same E cycle and both attempt to alter the same port A line, the OC1 compare prevails.

This function allows control of multiple I/O pins automatically with a single output compare.

Another intended use for the special I/O pin control on output compare 1 is to allow more than one output compare to control a single I/O pin. This allows pulses as short as one E clock cycle to be generated.

8.1.5 Timer Compare Force Register (CFORC)

The timer compare force register is used to force early output compare actions. The CFORC register is an 8-bit write-only register. Reads of this location have no meaning and always return logic zeros. Note that the compare force function si not generally recommended for use with the output toggle function because a normal compare occuring immediately before or after the force may result in undesirable operation.

	7	6	5	4	3	2	1	0	
$100B	FOC1	FOC2	FOC3	FOC4	FOC5	0	0	0	CFORC
RESET	0	0	0	0	0	0	0	0	

FOC1-FOC5 - Force Output Compare x Action

0 = Has no meaning

1 = Causes action programmed for output compare x, except the OCxF flag bit is not set.

Bits 2-0 - Not Implemented

These bits always read zero.

8.1.6 Output Compare 1 Mask Register (OC1M)

This register is used in conjunction with output compare 1 to specify the bits of port A which are affected as a result of a successful OC1 compare.

	7	6	5	4	3	2	1	0	
$100C	OC1M7	OC1M6	OC1M5	OC1M4	OC1M3	0	0	0	OC1M
RESET	0	0	0	0	0	0	0	0	

The bit of the OC1M register correspond bit-for-bit with the lines of port A (lines 7 through 3 only). For each bit that is affected by the successful compare, the corresponding bit in OC1M should be set to one.

Note that the pulse accumulator function shares line 7 of port A. If the DDRA7 bit in the pulse accumulator control register (PACTL) is set, then port A line 7 is configured as an output and OC1 can obtain access by setting OC1M bit 7. In this condition if the PAEN bit in the PACTL register is set, enabling the pulse accumulator input, then OC1 compares cause a write of OC1D bit 7 to an internal latch, and the output of that latch drives the pin and the pulse accumulator input. This action can then cause the pulse accumulator to take the appropriate action (pulse count or gate modes).

8.1.7 Output Compare 1 Data Register (OC1D)

This register used in conjunction with output compare 1 to specify the data which is to be stored to the affected bits of port A as the result of a successful OC1 compare.

	7	6	5	4	3	2	1	0	
$100D	OC1D7	OC1D6	OC1D5	OC1D4	OC1D3	0	0	0	OC1D
RESET	0	0	0	0	0	0	0	0	

The bits of the OC1D register correspond bit-for-bit with the lines of port A (lines 7 thru 3 only) . When a successful OC1 compare occurs, for each bit that is set in OC1M, the corresponding data bit in OC1D is stored in the corresponding bit of port A. If there is conflicting situation where an OC1 compare and another output compare function occur during the same E cycle with both attempting to alter the same port A line, the OC1 action prevails.

8.1.8 Timer Control Register 1 (TCTL1)

	7	6	5	4	3	2	1	0	
$1020	OM2	OL2	OM3	OL3	OM4	OL4	OM5	OL5	TCTL1
RESET	0	0	0	0	0	0	0	0	

OM2, OM3, OM4, and OM5 - Output Mode

OL2, OL3, OL4, and OL5 - Output Level

These two control bits (OMx and OLx) are encoded to specily the output action taken as a result of a successful OCx compare.

OMx	OLx	Timer disconnected from output pin logic
0	0	Toggle OCx output line
0	1	Toggle OCx output line
1	0	Clear OCx output line to zero
1	1	Set OCx output line to one

8.1.9 Timer Control Register 2 (TCTL2)

	7	6	5	4	3	2	1	0	
$1021	0	0	EDG1B	EDG1A	EDG2B	EDG2A	EDG3B	EDG3A	TCTL2
RESET	0	0	0	0	0	0	0	0	

Bits 7-6 - Not Implemented

These bits always read Zero.

EDGxB and EDGxA - Input Caputre x Edge Control.

These two bits (EDGxB and EDGxA) are cleared to zero by reset and are encoded to configure the input sensing logic for input capture x as follows:

EDGxB	EDGxA	Configuration
0	0	Capture disabled
0	1	Capture on rising edges only
1	0	Capture on falling edges only
1	1	Capture on any (rising or falling)edge

8.1.10 Timer Interrupt Mask Register 1 (TMSK1)

	7	6	5	4	3	2	1	0	
$1022	OC1I	OC2I	OC3I	OC4I	OC5I	IC1I	IC2I	IC3I	TMSK1
RESET	0	0	0	0	0	0	0	0	

OCxI - Output commpare x Interrupt

If the OCx enable bit is set when the OCx flag bit is set, a hardware interrupt sequence is requested.

ICxI - Input Capture x Interrupt

If the ICxI enable bit is set when the ICxF flag bit is set, a hardware interrupt sequence is requested.

8.1.11 Timer Interrupt Flag Register 1 (TFLG1)

Timer interrupt flag register 1 is used to indicate the occurrence of timer system events, and together with the TMSK1 register allows the timer subsystem to oprate in a polled or interrupt driven system. For each bit in TFLG1, there is a corresponding bit in TMSK1 in the same bit position. If the mask bit is set, each time the condition for the corresponding flag are met, a hardware interrupt sequence is requested as well as the flag bit being set.

These timer system status flags are cleared by writing a one to the bit positions corresponding to the flag (s) which are to be cleared. Bit manipulation instructions would be inappropriate for flag clearing because they are read-modify-write instruction. Even though the instruction mask implies that the programmer is only interested in some of the bits in the manipulated location, the entire location is actually read and rewritten which may clear other bits in the register.

	7	6	5	4	3	2	1	0	
$1023	OC1F	OC2F	OC3F	OC4F	OC5F	IC1F	IC2F	IC3F	TFLG1
RESET	0	0	0	0	0	0	0	0	

OCxF - Output Compare x Flag

This flag bit is set each time the timer counter matches the output compare register x value. A write of a zero does not affect this bit. A write of a one causes this bit to be cleared.

ICxF - Input Caputure x Flag

This flag is set each time a selected active edge is detected on the ICx input line. A write of a zero does not affect this bit. A write of a one causes this bit to be cleared.

8.1.12 Timer Interrupt Mask Register 2 (TMSK2)

Timer interrupt mask register 2 is used to control whether or not a hardware interrupt sequence is requested as a result of a status bit being set in timer interrupt flag register 2 . In addition, two timer prescaler bits are included in this register. For each of the four most significant bits in timer flag register 2, (TFLG2), there is a correponding bit in the timer mask register 2 (TMSK2) in the same bit position.

	7	6	5	4	3	2	1	0	
$1024	TOI	RTII	PAOVI	PAII	0	0	PR1	PR0	TMSK2
RESET	0	0	0	0	0	0	0	0	

TOI - Timer Overflow Interrupt Enable

0 = TOF interrupts disabled

1 = Interrupt requested when TOF = 1

RTII - RTI Interrupt Enable

0 = RTIF interrupts disabled

1 = Interrupt requested when RTIF = 1

PAOVI - Pulse Accumulator Overflow Interrupt Enable

0 = PAOVF interrupts disabled

1 = Interrupt requested when PAOVF = 1

PAII - Pulse Accumulator Input Interrupt Enable

0 = PAIF interrupts disabled

1 = Interrupt requested when PAIF = 1

Bits 3 and 2 - Not Implemented

These bits always read zero.

PR1 and PRO - Timer Prescaler Selects

These two bits may be read at any time but may only be written during initalization. Writes are disabled after the first write or after 64 E cycles out of reset. If the MCU is in special test or special bootstrap mode, then these two bits may be written any time.

These two bits specify the timer prescaler divide factor.

PR1	PR0	Divide-by-Factor
0	0	1
0	1	4
1	0	8
1	1	16

8.1.13 Timer Interrupt Flag Register 2 (TFLG2)

Timer interrupt flag register 2 is used to indicate the occurrence of timer system events and together with the TMSK2 register, allows the timer subsystems to operate in a polled or interrupt driven system. For each bit in timer flag register 2 (TFLG2), there is a corresponding bit in timer mask register 2 (TMSK2) in the same bit position. If the enable bit is set each time the conditions for the corresponding flag are met, a hardware interrupt sequence is requested as well as the flag bit being set.

The timer system status flags are cleared by writing a one to the bit positions corresponding to the flag(s) which are to be cleared. Bit manipulation instructions would be inappropriate for flag clearing because they are read-modify-write instructions. Even though the instruction mask implies that the programmer is only interested in some of the bits in the manipulated location, the entire location is actually read and rewritten which may clear other bits in the register.

	7	6	5	4	3	2	1	0	
$1025	TOF	RTIF	PAOVF	PAIF	0	0	0	0	TFLG2
RESET	0	0	0	0	0	0	0	0	

TOF - Timer Overflow

This bit is cleared by reset. It is set to one each time the 16-bit free-running counter advances from a value of $FFFF to $0000. This bit is cleared by a write to the TFLG2 register with bit 7 set.

RTIF - Real Time Interrupt Flag

This bit is set at each rising edge of the selected tap point. This bit is cleared by a write to the TFLG2 register with bit 6 set.

PAOVF - Pulse Accumulator Overflow Interrupt Flag

This bit is set when the count in the pulse accumulator rolls over from $FF to $00. This bit is cleared by a write to the TFLG2 register with bit 5 set.

PAIF - Pulse Accumulator Input Edge Interrupt Flag

This bit is set when an active edge is detected on the PAI input pin. This bit is cleared by a write to the TFLG2 register with bit 4 set.

Bits 3-0 - Not Implemented

These bits always read zero.

8.2 REAL TIME INTERRUPT

The real time interrupt feature on the MCU is configured and controlled by using two bits(RTR1 and RTR0) in the PACTL register to select one of four interrupt rates. The RTII bit in the TMSK2 register enables the interrupt capability. Every timeout causes the RTIF bit in TFLG2 to be set, and if RTII is set, and interrupt request is generated. After reset, one entire real time interrupt period elapses before the RTIF flag is set for the first time.

8.3 PULSE ACCUMULATOR

The pulse accumulator is an 8-bit read/write counter which can operate in either of two modes (exernal event counting or gated time accumulation) depending on the state of the PAMOD control bit in the PACTL register. In the event counting mode, the 8-bit counter is clocked to increasing values by an external pin. The maximum clocking rate for the external event counting mode is E clock divided by two. In the gated time accumulation mode, a free-running E clock/64 signal drives the 8-bit counter, but only while the external PAI input pin is activated.

The pulse accumulator uses port A bit 7 as its PAI input, but this pin also shares function as a general purpose I/O pin and as a timer output compare pin. Normally port A bit 7 would be configure as a input when being used for the pulse accumulator. Note that even when port A bit 7 is configured for output, this pin still drives the input to the pulse accumulator.

8.4 PULSE ACCUMULATOR CONTROL REGISTER (PACTL)

Four bits in this register are used to control an 8-bit pulse accumulator system and two other bits are used to select the rate for the real time interrupt system.

	7	6	5	4	3	2	1	0	
$1026	DDRA7	PAEN	PAMOD	PEDGE	0	0	RTR1	RTR0	PACTL
RESET	0	0	0	0	0	0	0	0	

DDRA7 - Data Direction for Port A Bit 7

0 = Input only

1 = Output

PAEN - Pulse Accumulator System Enable

0 = Pulse accumulator off

1 = Pulse accumulator on

PAMOD - Pulse Accumulator Mode

0 = External event counting

1 = Gated time accumulation

PEDGE - Pulse accumulator Edge Control

This bit has different meanings depending on the state of the PAMOD bit.

PAMOD	PEDGE	Action on Clock
0	1	PAI Falling Edge Increments the Counter
0	1	PAI Rising Edge Increments the Counter
1	0	A '0' on PAI Inhibits Counting
1	1	A '1' on PAI Inhibits Counting

Bits 3-2 - Not Implemented

These bits always read zero.

RTR1 and RTR0 - RTI Interrupt Rate Selects

These two bits select one of four rates for the real rime periodic interrupt circuit (see Table 8.1). Reset clears these two bits and after reset, a full RTI period elapses before the first RTI interrupt.

Table 8.1 Real Time Interrupt Rate versus RTR1 and RTR0

RTR1	RTR0	Divide E By	XTAL = 2^{23}	XTAL = 8.0 MHz	XTAL = 4.9152 MHz	XTAL = 4.0 MHz	XTAL = 3.6864 MHz
0	0	2^{13}	3.91 ms	4.10 ms	6.67 ms	8.19 ms	8.89 ms
0	1	2^{14}	7.81 ms	8.19 ms	13.33 ms	16.38 ms	17.78 ms
1	0	2^{15}	15.62 ms	16.38 ms	26.67 ms	32.77 ms	35.56 ms
1	1	2^{16}	31.25 ms	32.77 ms	53.33 ms	65.54 ms	71.11 ms
		E =	2.1 MHz	2.0 MHz	1.2288 MHz	1.0 MHz	921.6 MHz

9. RESETS, INTERRUPTS, AND LOW POWER MODES

This section provides a description of the resets, interrupts, and low power modes. The computer operating properly (COP) watchdog system and clock monitor are described as part of the reset system. The interrupt description includes a flowchart to illustrate how interrupts are executed.

9.1 RESETS

The MCU has four possible types of reset:an active low external reset pin ($\overline{\text{RESET}}$), a power-on reset, a computer operating properly (COP) watchdog timer reset, and a clock monitor reset.

9.1.1 External $\overline{\text{RESET}}$ pin

The $\overline{\text{RESET}}$ pin is used to reset the MCU and allow an orderly software startup procedure. When a reset condition is sensed, this pin is driven low by an internal device for four E clock cycles, then released, and two E clock, cycles later it is sampled. If the pin is still low, it means that an external reset has occurred. If the pin is high, it implies that the reset was initiated internally by either the watchdog timer (COP) or the clock monitor (refer to Figure 9.1). This method of differentiation between internal and external reset conditions assumes that the reset pin will rise to a logic one in less than two E clock cycles once it is released and that an externally generated reset should stay active for at least eight E clock cycles.

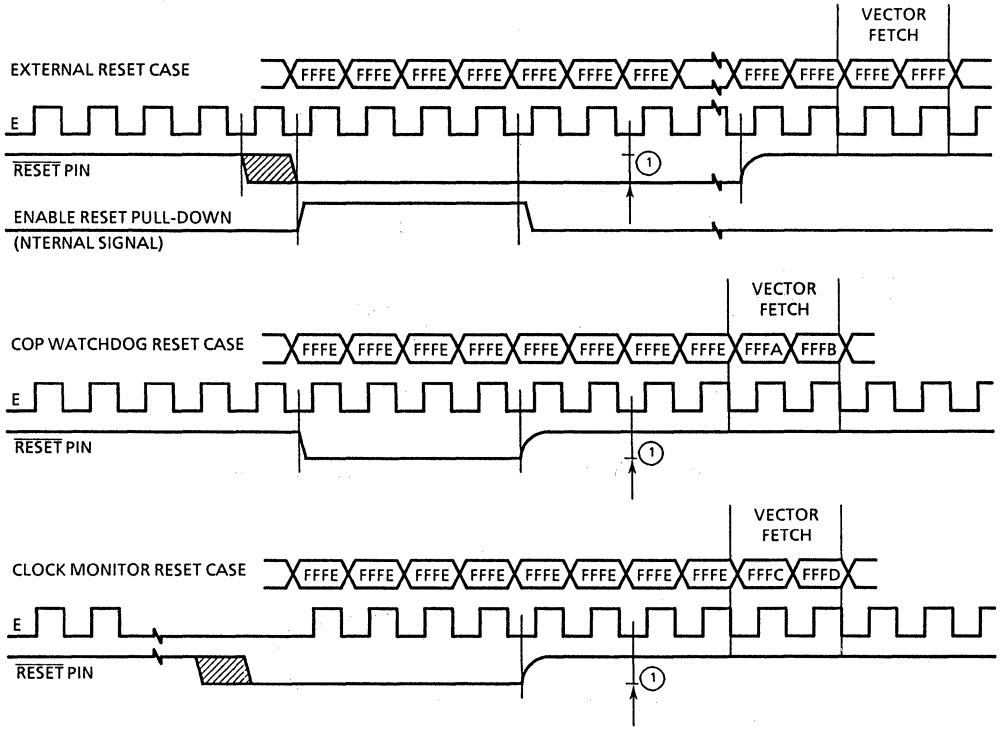
Since there is EEPROM on chip, it is very important to control reset during power transitions. If the reset line is not held low while V_{DD} is below its minimum operating level, the EEPROM contents could be corrupted. Corruption occurs due to improper instruction execution when there is not sufficient voltage to execute instructions correctly. Both EEPROM memories and the EEPROM based CONFIG register are subject to this potential problem.

A low voltage inhibit (LVI) circuit which holds reset low whenever V_{DD} is below its minimum operating level is required to protect against EEPROM corruption. Figures 9.2 and 9.3 show two examples of reset circuits with LVI capabilities. The best circuit for a particular application may be different from either of these suggested circuits.

The circuit in Figure 9.2 includes an R-C turn on delay. In older dynamic MCU designs this delay was required to allow the crystal oscillator to stabilize. Since the TMP68HC11A8 is a fully static CMOS design, it is capable of operating at clock rates down to DC and therefore a turn on delay is not required. Though not required for proper MCU operation, a turn on delay could be dictated by the system requirements of a particular application. For example, if incorrect time period measurements during the first few tens of milliseconds of operation cannot be tolerated, a turn on delay is probably needed.

9.1.2 Power-On Reset

The power-on reset occurs when a positive transition is detected on V_{DD}. The power-on reset is used strictly for power turn-on conditions and should not be used to detect any drops in power supply voltage. The power-on circuitry provides a 4064 cycle time delay from the time of the first oscillator operation. In a system where E=2MHz, power on reset lasts about 2 milliseconds. If the external $\overline{\text{RESET}}$ pin is low at the end of the power-on delay time, the MCU remains in the reset condition until the $\overline{\text{RESET}}$ pin goes high.



Note (1) $\overline{\text{RESET}}$ pin is sampled at this time. Low implies an extemal reset. High implies clock monitor or COP watchdog caused reset.

Figure 9.1 Reset Timing

9.1.2.1 CPU.

After reset, the CPU fetches the restart vector from locations $FFFE and $FFFF ($BFFE and $BFFF if in special bootstrap or special test operating mode) during the first three cycles, and begins executing instructions. The stack pointer and other CPU registers are indeterminate immediately after reset;however, the X and I

interrupt mask bits in the condition code register are set to mask any interrupt requests. Also, the S bit in the condition code register is set to disable the STOP mode.

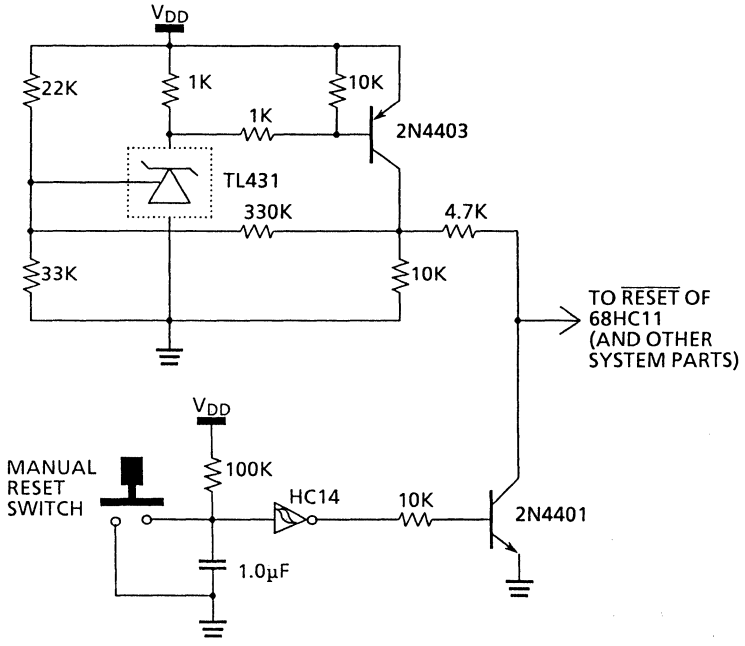


Figure 9.2 Reset Circuit with LVI and RC Delay

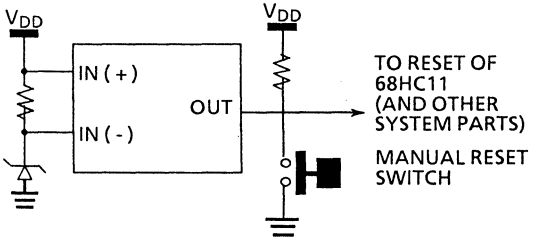


Figure 9.3 Simple LVI Reset Circuit

9.1.2.2 Memory Map.

After reset, the INIT register is initialized to $01, putting the 256 bytes of RAM at locations $0000 through $00FF and the control registers at locations $1000 through $103F. The 8K-bite ROM and/or the 512-byte EEPROM may or may not be present in the memory map because the two bits that enable them in the CONFIG register are EEPROM cells and are not affected by reset or power down.

9.1.2.3 Parallel I/O.

When a reset occurs in expanded multiplexed operating mode, the 18 pins used for parallel I/O are dedicated to the expansion bus. If a reset occurs in the single-chip operating mode, the STAF, STAI, and HNDS bits in the parallel input/output control register (PIOC) are cleared so that no interrupt is pending or enabled, and the simple strobed mode (rather than full handshake mode) of parallel I/O is selected. The CWON bit in PIOC is cleared so port C is not in wired-OR mode.

Port C is initialized as an input port (DDRC = $00), port B is general purpose output port with all bits cared. STRA is the edge-sensitive strobe A input and the active edge is initially configured to detect rising edges (EGA bit in the PIOC set), and SARB is the strobe B output and is initially a logic zero (the INVB bit in the PIOC is set). Port C, port D bits 0 through 5, port A bits 0, 1, 2, and 7, and port E are configured as general purpose high-impedance inputs. Port B and bits 3 through 6 of port A have their directions fixed as outputs and their reset state is a logic zero.

9.1.2.4 Timer.

During reset, the timer system is initialized to a count of $0000. The prescaler bits are cleared, and all output compare registers are initialized to $FFFF. All input capture registers are indeterminate after reset. The output compare 1 mask (OC1M) register is cleared so that successful OC1 compares do not affect any I/O pins. The other four output compares are configured to not affect any I/O pins on successful compares. All three input capture edge-detector circuits are configured for "capture disabled" operation. The timer overflow interrupt flag and all eight timer function interrupt flags are cleared. All nine timer interrupts are disabled since their mask bits are cleared.

9.1.2.5 Real Time Interrupt.

The real time interrupt flag is cleared and automatic handware interrupts are masked. The rate control bits are cleared after reset and may be initialized by software before the real time interrupt system is used.

9.1.2.6 Pulse Accumulator.

The pulse accumulator system is disabled at reset so that the PAI input pin defaults to being a general purpose input pin.

9.1.2.7 COP.

The COP watchdog system is enabled if the NOCOP control bit in the system configuration control register (EEPROM cell) is clear, and disabled if NOCOP is set. The COP rate is set for the shortest duration timeout.

9.1.2.8 SCI Serial I/O.

The reset condition of the SCI system is independent of the operating mode. At reset, the SCI baud rate is indeterminate and must be established by a software write to the BAUD register. All transmit and receive interrupts area masked and both the transmitter and receiver are disabled so the port pins default to being general purpose I/O lines. The SCI frame format is initialized to an 8-bit character size. The send break and receiver wake up functions are disabled. The TDRE and TC status bits in the SCI status register are both set, indicating that there is no transmit data in either the transmit data register of the transmit serial shift register. The RDRF, IDLE, OR NF, and FE receive-related status bits are all cleared.

Note that upon reset in special bootstrap mode execution begins in the 192 byte boot ROM. This firmware sets port D to wire-OR mode, establishes a baud rate, and enables the SCI receiver and transmitter.

9.1.2.9 SPI Serial I/O.

The SPI system is desabled by reset. The port pins associated with this function default to being general purpose I/O lines.

9.1.2.10 A/D Converter.

The A/D converter system configuration is indeterminate after reset. The conversion complete flag is cleared by reset. The ADPU bit is cleared by reet thus disabling the A/D system.

9.1.2.11 System.

The EEPROM programming controls are all disabled so the memory system is configured for normal read operation. The highest priority I interrupt defaults to being the external $\overline{IRQ}$ pin by PSEL3-PSEL0 equal to 0:1:0:1. The $\overline{IRQ}$ interrupt pin is configured for level sensitive operation (for wire-OR systems). The RBOOT, SMOD, and MDA bits in the HPRIO register reflect the status of the MODB and MODA inputs at the rising edge of reset. The DLY control bit is set to specify that an oscillator start-up delay is imposed upon recovery from STOP mode. The clock monitor system is disabled by CME equal zero.

9.1.3 Computer Operating Properly (COP) Reset

The MCU includes a computer operating properly watchdog system to help protect against software failues. To use a COP watchdog timer, a watchdog timer reset sequence must be executed on a regular periodic basis so that the watchdog timer is never allowed to time out.

The internal COP function includes special control bits which permit specification of one of four time out periods and even allows the function to be disabled completely. The COP system has a separate reset vector.

The NOCOP control bit, which determines whether or not a wathcdog timeout causes a system reset, is implemented in an EEPROM cell in the CONFIG register. Once programmed, this bit remains set (or cleared) even when no power is applied, and the COP function is enabled or disabled independent of resident software. The NOCOP control bit may be preempted while in special test modes to prevent the COP system from causing a hardware reset.

Two other control bits in the OPTION register select one of four timeout durations for the COP timer. The actual timeout period is dependent on the system E clock frequency, but for reference purposes, Table 9.1 shows the relationship between the CR1 and CR0 control bits and the COP timeout period for various system clock frequencies.

Table 9.1. COP Timeout Period versus CR1 and CR0

CR1	CR0	E/2^{15} Divided BY	XTAL = 2^{23} Timeout – 0/ + 15.6ms	XTAL = 8.0 MHZ Timeout – 0/ + 16.4ms	XTAL = 4.9152 MHZ Timeout – 0/ + 26.7ms	XTAL = 4.0 MHZ Timeout – 0/ + 32.8ms	XTAL = 3.6864 MHZ Timeout – 0/ + 35.6ms
0	0	1	15.625 ms	16.384 ms	26.667 ms	32.768 ms	35.556 ms
0	1	4	62.5 ms	65.536 ms	106.67 ms	131.07 ms	142.22 ms
1	0	16	250 ms	262.14 ms	426.67 ms	524.29 ms	568.89 ms
1	1	64	1 s	1.049 s	1.707 s	2.1 s	2.276 s
		E =	2.1MHz	2.0MHz	1.2288MHz	1.0MHz	921.6kHz

The default reset condition of the CR1 and CR0 bits is cleared which corresponds to the shortest timeout period.

The sequence required to reset the watchdog timer is:

(1) Write $55 to the COP reset register (COPRST) as $103A, followed by

(2) Write $AA to the same address.

Both writes must occur in correct order prior to timeout but, any number of instructions may be executed between the writes. The elapsed time between adjacent software reset sequence must never be greater than the COP time out period. Reading the COPRST register does not return meaningful data and does not affect the watchdog timer.

9.1.4 Clock Monitor Reset

The clock monitor function is enabled by the CME control bit in the OPTION register. When CME is clear, the monitor function is disabled. When the CME bit is set, the clock monitor function detects the absence of an E clock for more than a certain period of time. The timeout period is dependent on processing parameters and will be between 5 and

100 microseconds. This means that an E-clock rate of 200 kHz or more will never cause a clock monitor failure and an E-clock rate of 10 kHz or less will definitely cause a clock monitor failure. This implies that systems operating near or below an E-clock rate of 200 kHz should not use the clock monitor function.

Upon detection of a slow or absent clock, the clock monitor circuit will cause a system reset. This reset is issued to the external system via the bidirectional $\overline{\text{RESET}}$ pin. The clock monitor system has a separate reset vector.

Special considerations are needed when using a STOP function and clock monitor in the same system. Since the STOP function causes the clocks to be halted, the clock monitor function will generate a reset sequence if it is enabled at the time the STOP mode is entered.

The clock monitor is useful as a backup for the COP watchdog timer. Since the watchdog timer requires a clock to function, it will not indicate any failure if the system clocks fail. The clock monitor would detect such a failure and force the MCU to its reset state. Note that clocks are not required for the MCU to reach its reset configuration, although clocks are required to sequence through reset back to the run condition.

9.1.5 Configuration Options Register (OPTION)

This is a special purpose 8-bit register that is used (optionally) during initialization to configure internal system configuration options. With the exception of bits 7, 6, and 3 (ADPU, CSEL, and CME) which may be read or written at any time, this register may be written to only once after a reset and thereafter is a read-only register. If no write is performed to this location within 64 E-clock cycles after reset, then bits 5, 4, 1, and 0 (IRQE, DLY, CR1, and CR0) will become read-only to minimize the possibility of any accidential changes to the system configuration (writes will be ignored). While in special test modes, the protection mechanism on this register is preempted and all bits in the OPTION register may be written repeatedly.

	7	6	5	4	3	2	1	0	
$1039	ADPU	CSEL	IRQE	DLY	CME	0	CR1	CR0	OPTION
RESET	0	0	0	1	0	0	0	0	

ADPU - A/D Powerup

This bit controls operations of the on-chip analog-to-digital converter. When ADPU is clear, the A/D system is powered down and conversion requests will not return meaningful information. To use the A/D system, this bit should be set. A 100 microsecond delay is required after ADPU is turned on to allow the A/D system to stabilize.

CSEL - A/D/EE Charge Pump Clock Source Select

This bit determines the clocking source for the on-chip A/D and EEPROM charge pump. When this bit is zero, the MCU E clock drives the A/D system and the EEPROM charge pump. When CSEL is one, on-chip separate R-C oscillators are enabled and clock the systems at about 2MHz. When running with an E clock below 1MHz, CSEL must be high to program or erase EEPROM. When operating below 750KHz E clock rate, CSEL should be high for A/D conversions. A delay of 10 milliseconds is required after CSEL is turned on to allow the A/D system to stabilize.

IRQE - IRQ Edge/Level Sensitive

This bit may only be written under special circumstances as described above. When this bit is clear, the $\overline{IRQ}$ pin is configuredfor level sensitive wired-OR operation (low level) and when it is set, the $\overline{IRQ}$ pin is configured for edge-only sensitivity (falling edges).

DLY - STOP Exit Turn-On Delay

This bit may only be written under special circumstances as described above. This bit is set during reset and controls whether or not a relatively long turn-on delay will be imposed before processing can resume after a STOP period. If an external clock source is supplied this delay can be inhibited so that processing can resume within a few cycles of a wake up from STOP mode. When DLY is set, a 4064 E clock cycle delay is imposed to allow oscillator stabilization and when DLY is clear, this delay is bypassed.

CME - Clock Monitor Enable

This control bit may be read or written at any time and controls whether or not the internal clock monitor circuit will trigger a reset sequence when a slow or absent system clock is detected. When it is clear, the clock monitor circuit is disabled and when it is set, the clock monitor circuit is enabled. Systems operating at or below 200 kHz should not use the clock monitor function. Reset clears the CME bit.

Bit 2 - Not Implemented

This bit always reads zero.

CR1 and CR0 - COP Timer Rate Selects

These bits may only be written under special circumstances as described above. Refer to Table 9.1 for the relationship between CR1:CR0 and the COP timeout period.

9.2 INTERRUPTS

When an external or internal (hardware) interrupt occurs, the interrupt is not serviced until the current instruction being executed is completed. Until the current instruction is complete, the interrupt is considered pending. After completion of current instruction execution, unmasked interrupts may be serviced in accordance with an

established fixed hardware priority circuit; however, one I-bit related interrupt source may be dynamically elevated to the highest I bit priority position in the heirarchy (see 9.2.5 Highest Priority I Interrupt Register (HPRIO)).

Seventeen hardware interrupts and one software interrupt (excluding reset type interrupts) can be generated from all of the possible sources. The interrupts can be divided into two basic categories, maskable and non-maskable. In the TMP68HC11A8 fifteen of the interrupts can be masked using the condition code register I bit. In addition to being maskable by the I bit in the condition code register, all of the on-chip interrupt sources are individually maskable by local control bits.

The software interrupt (SWI instruction) is a non-maskable instruction rather than a maskable interrupt source. The illegal opcode interrupt is a non-maskable interrupt. The last interrupt source, external input to the $\overline{\text{XIRQ}}$ pin, is considered a non-maskable interrupt because once enabled, it cannot be masked by software;however, it is masked during reset and upon receipt of an interrupt at the $\overline{\text{XIRQ}}$ pin. Tables 9.2, 9.3, and 9.4 provide a list of each interrupt, its vector location in memory, and the actual condition code and control bits that mask it. A discussion of the various interrupts is provided below. Figure 9.4 shows the interrupt stacking order.

Table 9.2 Interrupt Vector Assignments

Vector Address	Interrupt Source	CC Register Mask	Local Mask
FFC0, C1	Reserved	–	–
•	•		
•	•		
FFD4, D5	Reserved	–	–
FFD6, D7	SCI Serial System	I Bit	See Table 9.3
FFD8, D9	SPI Serial Transfer Complete	I Bit	SPIE
FFDA, DB	Pulse Accumulator Input Edge	I Bit	PAII
FFDC, DD	Pulse Accumulator Overflow	I Bit	PAOVI
FFDE, DF	Timer Overflow	I Bit	TOI
FFE0, E1	Timer Output Compare 5	I Bit	OC5I
FFE2, E3	Timer Output Compare 4	I Bit	OC4I
FFE4, E5	Timer Output Compare 3	I Bit	OC3I
FFE6, E7	Timer Output Compare 2	I Bit	OC2I
FFE8, E9	Timer Output Compare 1	I Bit	OC1I
FFEA, EB	Timer Input Compare 3	I Bit	OC3I
FFEC, ED	Timer Input Compare 2	I Bit	OC2I
FFEE, EF	Timer Input Compare 1	I Bit	OC1I
FFF0, F1	Real Time Interrupt	I Bit	RTII
FFF2, F3	$\overline{IRQ}$ (External Pin or Parallel I/O)	I Bit	See Table 9.4
FFF4, F5	$\overline{XIRQ}$ Pin (Pesudo Non-Maskable Interrupt)	X Bit	None
FFF6, F7	SWI	None	None
FFF8, F9	Illegal Opcode Trap	None	None
FFFA, FB	COP Failure (Reset)	None	NOCOP
FFFC, FD	COP Clock Monitor Fail (Reset)	None	CME
FFFE, FF	$\overline{RESET}$	None	None

Table 9.3 SCI Serial System Interrupts

Interrupt Cause	Local Mask
Receive Data Register Full	RIE
Receive Overrun	RIE
Idle Line Detect	ILIE
Transmit Data Register Empty	TIE
Transmit Complete	TCIE

Table 9.4 IRQ Vector Interrupts

Interrupt Cause	Local Mask
External Pin	None
Parallel I/O Handshake	STAI

9.2.1 Software Interrupt (SWI)

The software interrupt is executed in the same manner as any other instruction and will take precedence over interrupts only if the other interrupts are masked (I and X bits in the condition code register set). The SWI instruction is executed in a manner similar to other maskable interrupts in that it sets the I bit, CPU registers are stacked, etc.

Note : The SWI instruction will not be fetched if another interrupt os pending. However, once an SWI instruction has begun, no other interrupt can be honored until the SWI vector has been fetched.

9.2.2 Illegal Opcode Trap

Since not all possible opcodes or opcode sequences are defined, an illegal opcode detection circuit has been included. When an illegal opcode is detected, an interrupt is requested to the illegal opcode vector. The illegal opcode vector should never be left uninitialized. It is a good idea to reinitialize the stack pointer as a result of an illegal opcode interrupt so repeated execution of illegal opcodes does not cause stack overruns.

9.2.3 Interrupt Mask Bits in Condition Code Register

Upon reset, both the X bit and the I bit are set to inhibit all maskable interrupts and $\overline{\text{XIRQ}}$. After minimum system initialization, software may clear the X bit by a TAP instruction, thus enabling $\overline{\text{XIRQ}}$ interrupts. Thereafter software cannot set the X bit so an $\overline{\text{XIRQ}}$ interrupt is effectively a nonmaskable interrupt. Since the operation of the I bit related interrupt structure has no effect on the X bit, the external $\overline{\text{XIRQ}}$ pin remains effectively non-masked. In the interrupt priority logic, the $\overline{\text{XIRQ}}$ interrupt is a higher priority than any source that is maskable by the I bit. All I bit reated interrupts operate normally with their own priority relationship. When an I bit related interrupt occurs, the I bit is automatically set by hardware after stacking the condition code register byte, but the X bit is not affected. When an X bit related interrupt occurs, both the X bit and the I bit are automatically set by hardware after stacking the condition code register. An RTI (return from interrupt) instruction restores the X and I bits to their pre-interrupt request state.

9.2.4 Priority Structure

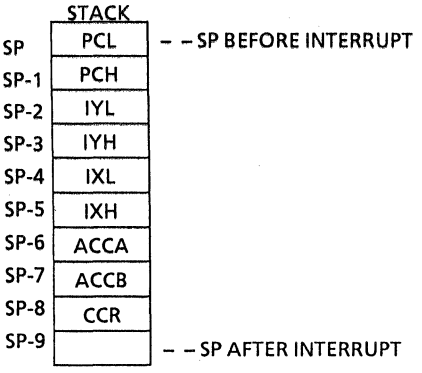


Figure 9.4 Interrupt Stacking Order

Interrupts obey a fixed hardware priority circuit to resolve simultaneous requests; however, one I bit related interrupt source may be elevated to the highest I bit priority position in the resolution circuit. The first six interrupt sources are not masked by the I bit in the condition code register and have the fixed priority interrupt relationship of:reset, clock monitor fail, COP fail, illegal opcode, and $\overline{\text{XIRQ}}$. (SWI is actually an instruction and has highest priority other than reset in the sense that once the SWI opcode is fetched, no other interrupt can be honored until the SWI vector has been fetched). Each of these sources is an input to the priority resolution circuit. The highest I bit masked priority input to the resolution circuit is assigned under software control (of the HPRIO register) to be connected to any one of the remaining I bit related interrupt sources. In order to avoid timing races, the HPRIO register may only be written while the I bit related interrupts are inhibited (I bit in condition code register is a logic one). An interrupt that is assigned to this high priority position is still subject to masking by any associated control bits or the I bit in the condition code register. The interrupt vector address is not affected by assigning a source to this higher priority position.

Figures 9.5, 9.6, and 9.7 illustrate the interrupt process as it relates to normal processing. Figure 9.5 shows how the CPU begins from a reset and how interrupt detection relates to normal opcode fetches. Figure 9.6 is an expansion of a block in Figure 9.5 and shows how interrupt priority is resolved. Figure 9.7 is an expansion of the SCI interrupt block in Figure 9.7 shows the resolution of interrupt sources within the SCI subsystem.

9.2.5 Highest Priority I Interrupt Register (HPRIO)

This register is used to select one of the I bit related interrupt sources to be elevated to the highest I bit masked position in the priority resolution circuit. In addition, four miscelleneous system control bits are included in this register.

	7	6	5	4	3	2	1	0	
$103C	RBOOT	SMOD	MDA	IRV	PSEL3	PSEL2	PSEL1	PSEL0	HPRIO
RESET	–	–	–	–	0	1	0	1	

RBOOT - Read Bootstrap ROM

The read bootstrap ROM bit only has meaning when the SMOD bit is a one (special bootstrap mode or special test mode). At all other times, this bit is clear and may not be written. When set, upon reset in bootstrap mode only, the small bootstrap loader program is enabled. When clear, by reset in the other three modes, this ROM is disabled and accesses to this area are treated as external accesses.

SMOD - Special Mode

The special mode bit reflects the inverse of the MODB input pin at the rising edge of reset. It is set if the MODB pin is low during reset. If MODB is high during reset, it is cleared. This bit may be cleared under software control from the special modes, thus, changing the operating mode of the MCU, but may never be set by software.

MDA - Mode Select A

The mode select A bit reflects the status of the MODA input pin at the rising edge of reset. While the SMOD bit is set (special bootstrap or special test mode in effect), the MDA bit may be written, thus, changing the operating mode of the MCU. When the SMOD bit is clear, the MODA bit is a read-only bit and the operating mode cannot be changed without going through a reset sequence. Table 9.5 summarizes the relationship between the SMOD and MDA bits and the MODB and MODA input pins at the rising edge of reset.

Table 9.5 Mode Bits Relationship

Inputs		Mode Description	Latched at Reset	
MODB	MODA		SMOD	MDA
1	0	Single Chip	0	0
1	1	Expanded Multiplexed	0	1
0	0	Special Bootstrap	1	0
0	1	Special Test	1	1

1 = Logic High 0 = Logic Low

IRV - Internal Read Visibility

The internal read visibility bit is used in the special modes (SMOD=1) to affect visibility of internal reads on the expansion data bus. IRV is writeable only if SMOD=1 and returns to zero if SMOD=0. If IRV is clear, visibility of internal reads is blocked. If the bit is set, internal reads are visible on the external bus.

PSEL3, PSEL2, PSEL1, and PSEL0 - Priority Select

These four priority select bits are used to specify one I bit related interrupt source which becomes the highest priority I bit related source (Table 9.6). These bits may be written only while the I bit in CCR=1 (interrupts masked).

Table 9.6 Highest Priority I Interrupt versus PSEL3-PSEL0

PSEL3	PSEL2	PSEL1	PSEL0	Interrupt Source Promoted
0	0	0	0	Timer Overflow
0	0	0	1	Pulse Accumulator Overflow
0	0	1	0	Pulse Accumulator Input Edge
0	0	1	1	SPI Serial Transfer Complete
0	1	0	0	SCI Serial System
0	1	0	1	Reserved (Default to $\overline{IRQ}$)
0	1	1	0	$\overline{IRQ}$ (External Pin or Parallel I / O)
0	1	1	1	Real Time Interrupt
1	0	0	0	Timer Input Capture 1
1	0	0	1	Timer Input Capture 2
1	0	1	0	Timer Input Capture 3
1	0	1	1	Timer Output Compare 1
1	1	0	0	Timer Output Compare 2
1	1	0	1	Timer Output Compare 3
1	1	1	0	Timer Output Compare 4
1	1	1	1	Timer Output Compare 5

Note: During reset, PSEL3, PSEL2, PSEL1, and PSEL0 are initialized to 0:1:0:1 which corre-sponds to "Reserved (default to $\overline{IRQ}$)" being the highest priority I bit related interupt source.

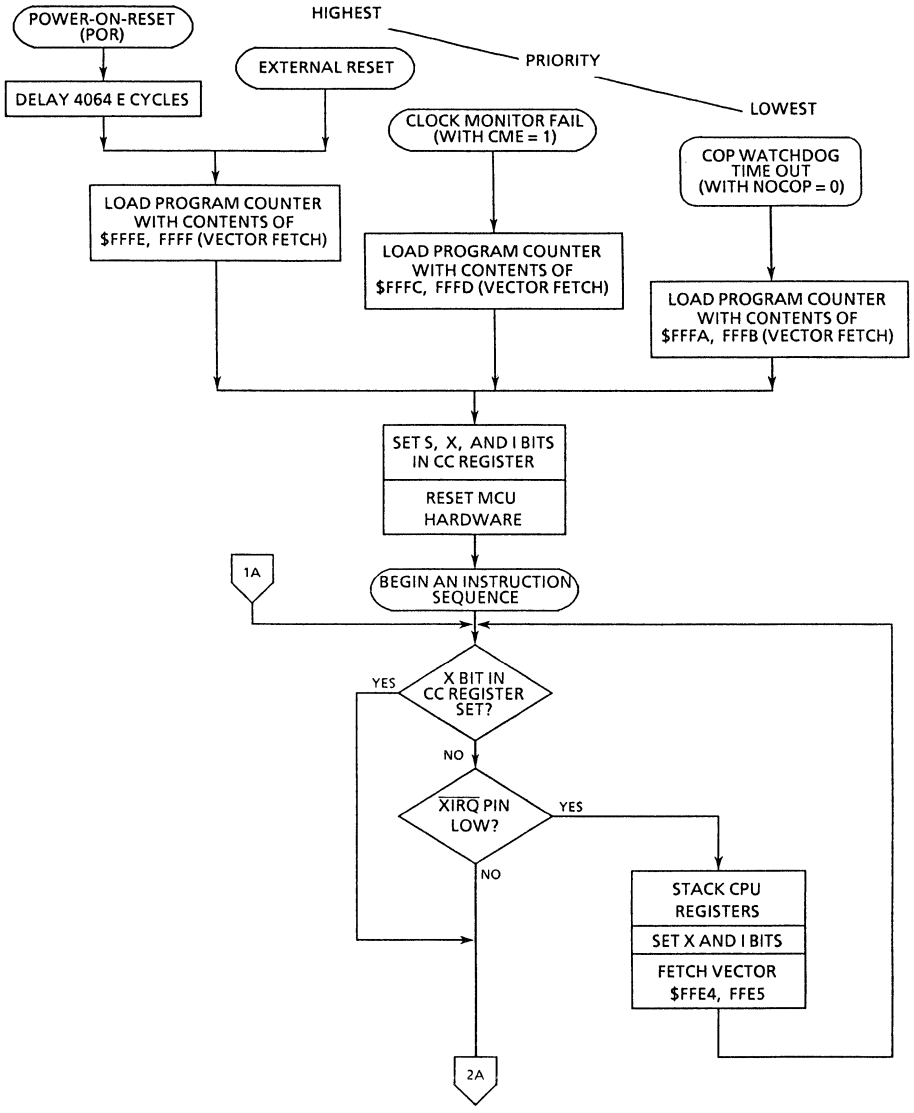


Figure 9.5 Processing Flow Out of Resets (Sheet 1of 2)

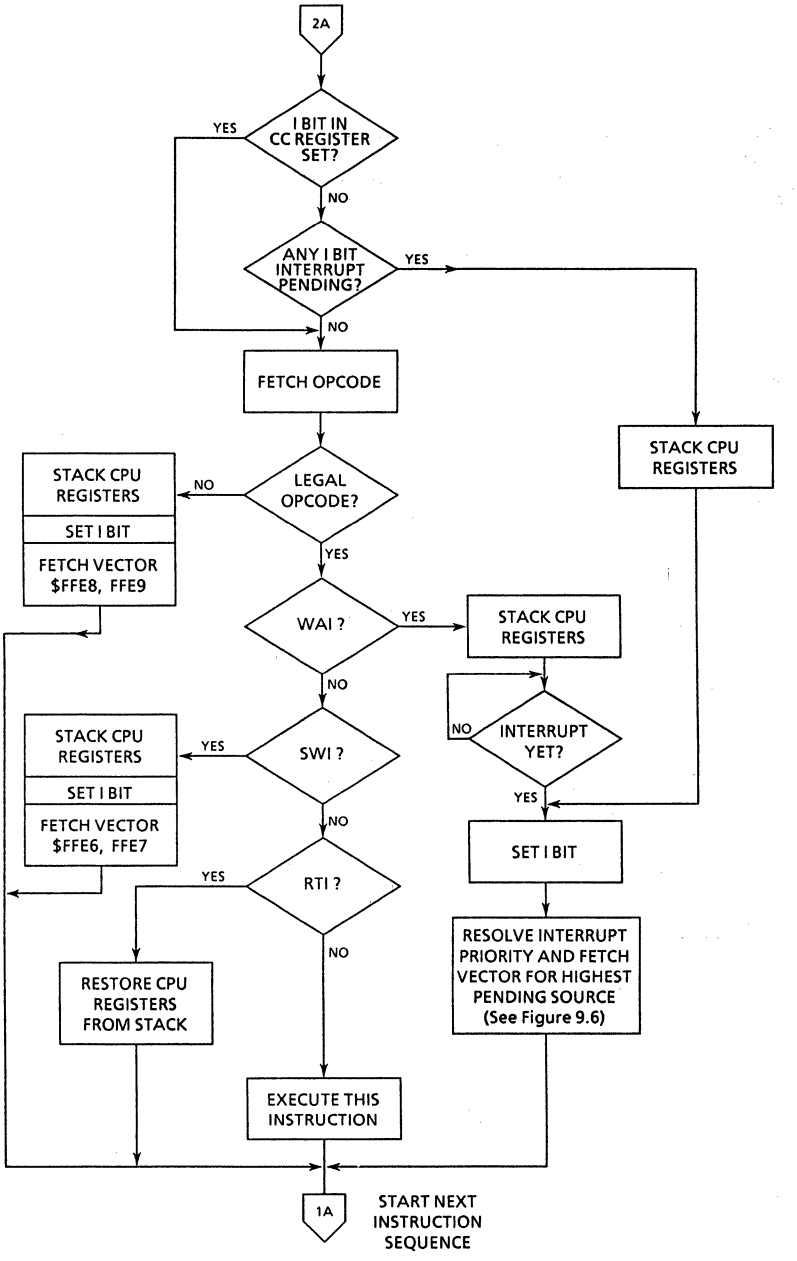


Figure 9.5 Processing Flow Out Resets (Sheet 2 of 2)

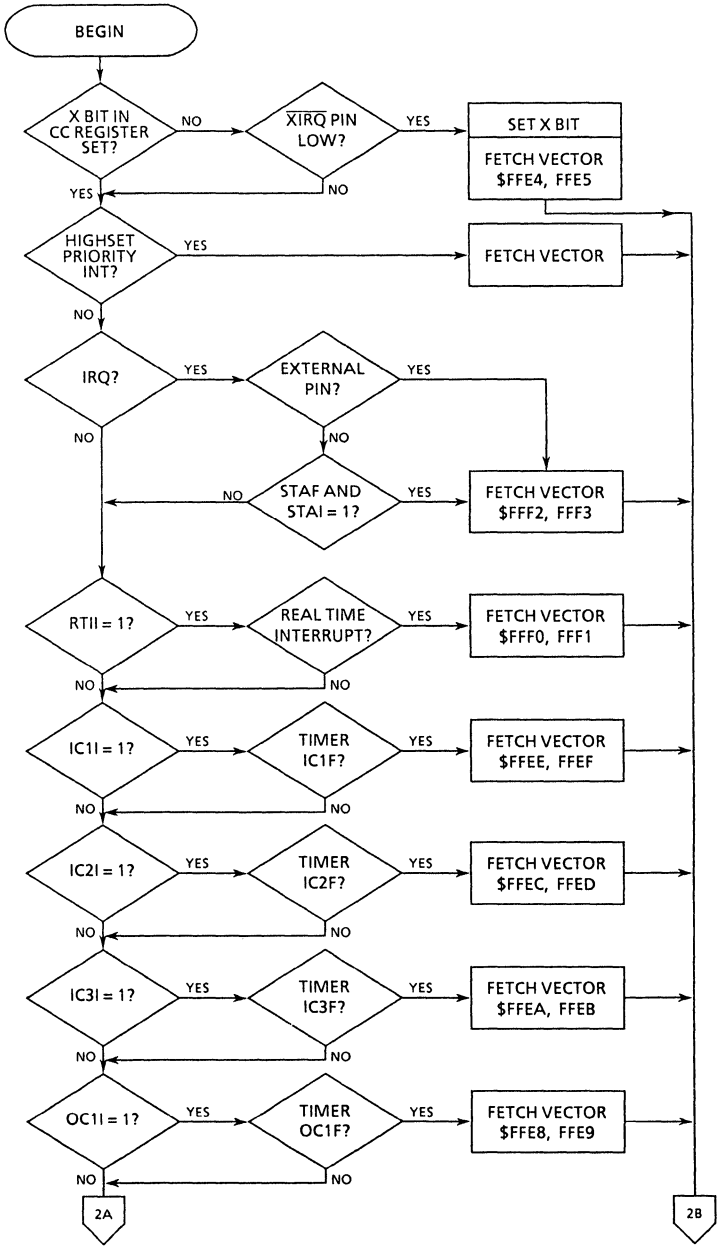
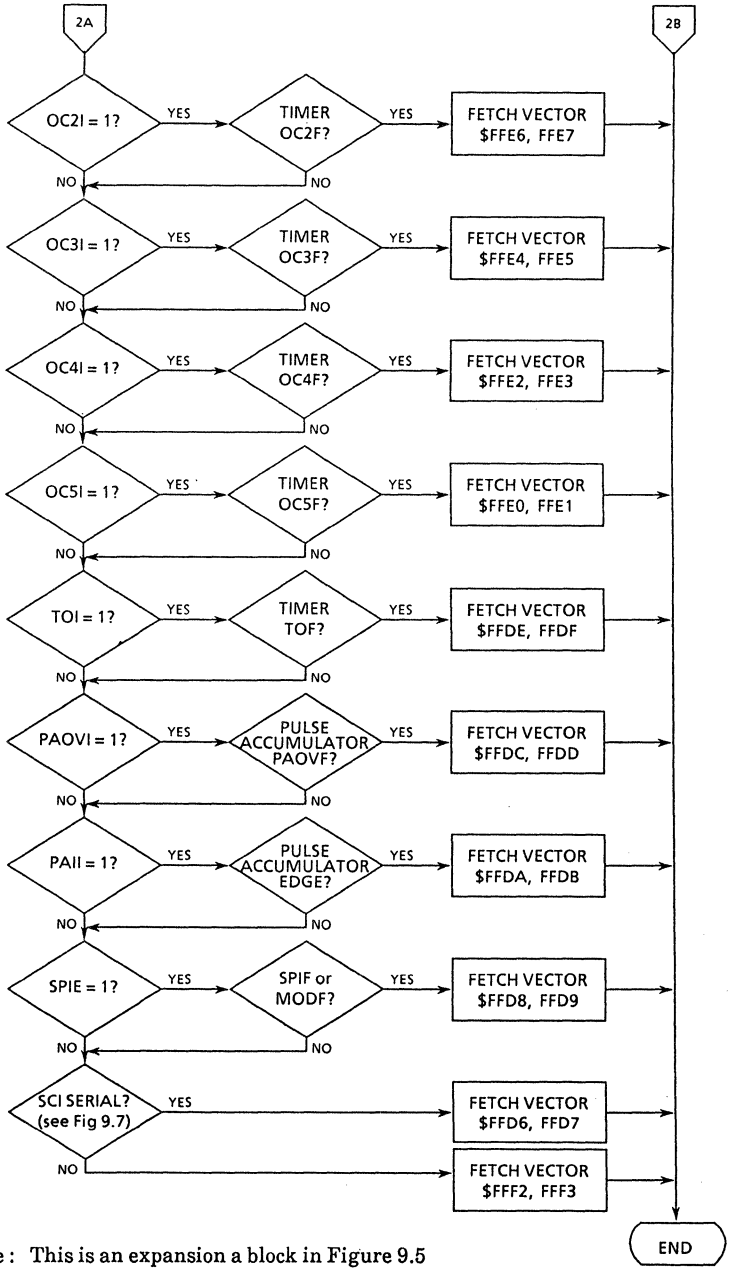
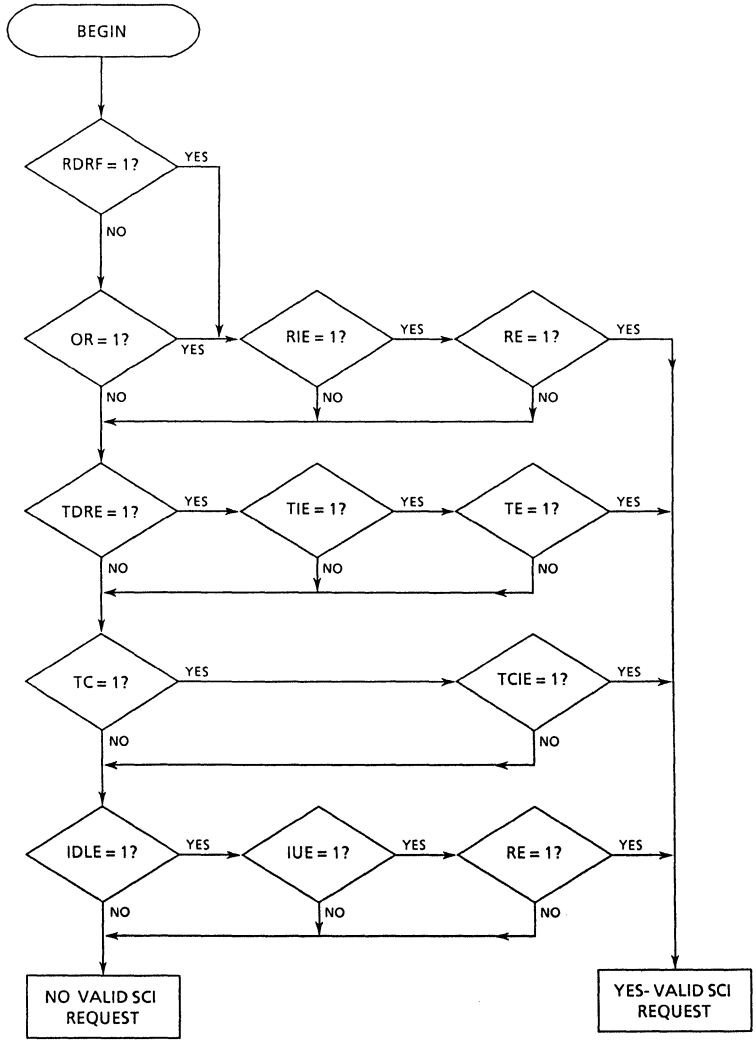


Figure 9.6 Interrupt Priority Resolution (Sheet 1 of 2)



Note : This is an expansion a block in Figure 9.5

Figure 9.6 Interrupt Priority Resolution (Sheet 2 of 2)



Note : This is an expansion a block in Figure 9.6

Figure 9.7 Interrupt Source Resolution Within SCI

9.3 LOW POWER MODES

The MCU contains two programmable low power consumption modes;WAIT and STOP. These two instructions are discussed below. Table 9.7 summarizes the activity on all pins of the MCU for all operating conditions.

9.3.1 WAIT Instruction

The WAI instruction puts the MCU in a low power consumption mode, keeping the oscillator running. Upon execution of a WAI instruction, the machine state is stacked and program execution stops. The wait state can be exited only by an unmasked interrupt or $\overline{\text{RESET}}$. If the I bit is set (interrupts masked) and the COP is disabled, the timer system will be turned off to additionally reduce power consumption. The amount of power savings is application dependent and depends upon circuitry connected to the MCU pins as well as which subsystems (i.e., timer, SPI, SCI) are active when the WAIT mode is entered. Turning off the A/D subsystem by clearing ADPU further reduces WAIT mode current.

9.3.2 STOP Instruction

The STOP instruction places the MCU in its lowest power consumption mode provided the S bit in the condition code register is clear. If the S bit is set, the STOP mode is disabled and STOP instructions are treated as NOPs (no operation). In the STOP mode, all clocks including the internal oscillator are stopped causing all internal processing to be halted. Recovery from the STOP mode may be accomplished by $\overline{\text{RESET}}$, $\overline{\text{XIRQ}}$, or an unmasked $\overline{\text{IRQ}}$. When the $\overline{\text{XIRQ}}$ is used, the MCU exits from the STOP mode regardless of the state of the X bit in the condition code register;however, the actual recovery sequence differs depending on the state of the X bit. If the X bit is clear, the MCU starts up with the stacking sequence leading to normal service of the $\overline{\text{XIRQ}}$ request. if the X bit is set, then processing will continue with the instruction immediately following the STOP instruction and no $\overline{\text{XIRQ}}$ interrupt service routine is requested. A $\overline{\text{RESET}}$ will always result in an exit from the STOP mode, and the start of MCU operation is determined by the reset vector.

Table 9.7 Pin State Summary for RESET, STOP, and WAIT

Pins	Single Chip Modes			Exanded Modes		
	RESET	WAIT	STOP	RESET	WAIT	STOP
Output Only						
E	Active E	Active E	0	Active E	Active E	0
XTAL!!!	Active	Active	1	Active	Active E	1
STRB/RW	0	SS	SS	1	1	1
PA3-PA6	0	SS	SS	0	SS	SS
PB0-PB7	0	SS	SS	HI ADD	HI ADD	HI ADD
Input/Output						
RESET	I (0)	I	I	I (0)	I	I
MODA/LIR	I (0)	OD (1)	OD (1)	I (1)	OD (1)	OD (1)
MODB/V_{STBY}	I (MODB)	I (V_{STBY})	I (V_{STBY})	I (MODES)	I (V_{STBY})	I (V_{STBY})
STRA/AS	I (STRA)	I (STRA)	I (STRA)	Active AS	Active AS	0
PA7	I	I/O	I/O	I	I/O	I/O
PC0-PC7	I	I/O	I/O	ADD/DATA	SP-8/DATA	I
PD0-PD5	I	I/O	I/O	I	I/O	I/O
Input Only						
EXTAL	Input Clock or Connect to Crystal with XTAL					
$\overline{IRQ}$	Terminate Unused Inputs to V_{DD}					
$\overline{XIRQ}$	Terminate Unused Inputs to V_{DD}					
PA0-PA2	Terminate Unused Inputs to V_{DD} or V_{SS}					
PE0-PE7	If Not Used, External Drive Not Required					
V_{RH}-V_{RL}	If Not Used, External Drive Not Required					

Symbols :

DATA	=	Current data present.
I	=	Input pin, if ( ) associated then this is required input state.
I/O	=	Input/output pin, state determined by data direction register.
HI ADD	=	High byte of the address.
ADD/DATA	=	Low byte of the address multiplexed with data.
OD	=	Open drain output, ( ) current output state.
SS	=	Steady state, output pin stays in current state.
SP-8	=	Address output during WAI period following WAI instruction, stack pointer value, at time of WAI, minus 8.
!!!	=	XTAL is output but not normally usable for any output function beyond crystal drive.

Since the oscillator is stopped in the STOP mode, a restart delay of 4064 clock cycle times may be required to allow oscillator stabilization. If the internal oscillator is being used, this delay is required;however, if a stable external oscillator is being used, a control bit in the OPTION register may be used (DLY = 0) to give a delay of four cycles.

10. CPU, ADDRESSING MODES, AND INSTRUCTION SET

This section provides a description of the CPU registers, addressing modes, and a summary of the M6811 instruction set. Special operations such as subroutine calls and interrupts are described and cycle-by-cycle operations for all instructions are presented.

10.1 CPU REGISTERS

In addition to being able to execute all M6800 and M6801 instructions, the TMP68HC11A8 uses a 4-page opcode map to allow execution of 91 new opcodes (see 10.2.7 Prebyte). Seven registers, discussed in the following paragraphs, are available to programmers as shown in Figure 10.1.

10.1.1 Accumulators A and B

Accumulator A and accumulator B are general-purpose 8-bit registers used to hold operands and results of arithmetic calculations or data manipulations. These two accumulators can be concatenated into a single 16-bit accumulator called the D accumulator.

10.1.2 Index Register X (IX)

The 16-bit IX register is used for indexed mode addressing. It provides a 16-bit indexing value which is added to an 8-bit offset provided in an instruction to create an effective address. The IX register can also be used as a counter or as a temporary storage register.

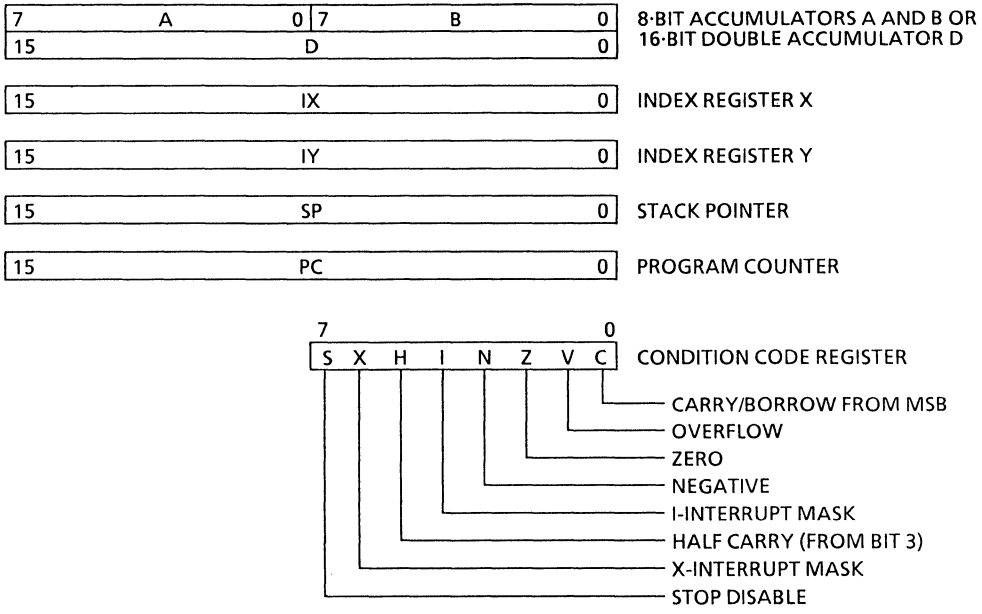


Figure 10.1 Programming Model

10.1.3 Index Register Y (IY)

The 16-bit IY register is also used for indexed mode addressing similar to the IX register;however, all instructions using the IY register require and extra byte of machine code and an extra cycle of execution time since they are two byte opcodes.

10.1.4 Stack Pointer (SP)

The stack pointer (SP) is a 16-bit register that contains the address of the next free location on the stack. The stack is configured as a sequence of last-in-first-out read/write registers which allow important data to be stored during interrupts and subroutine calls. Each time a new byte is added to the stack (a push), the SP is decremented;whereas, each time a byte is removed from the stack (a pull) the SP is incremented.

10.1.5 Program Counter (PC)

The program counter is a 16-bit register that contains the address of the next instruction to be executed.

10.1.6 Condition Code Register (CCR)

The condition code register is an 8-bit register in which each bit signifies the results of the instruction just executed. These bits can be individually tested by a program and a specific action can be taken as a result of the test. Each individual condition code register bit is explained below.

10.1.6.1 Carry/Borrow (C).

The C bit is set if there was a carry or borrow out of the arithmetic logic unit (ALU) during the last arithmetic operation. The C bit is also affected during shift and rotate instructions.

10.1.6.2 Overflow (V).

The overflow bit is set if there was an arithmetic overflow as a result of the operation; otherwise, the V bit is cleared.

10.1.6.3 Zero (Z).

The zero bit is set if the result of the last arithmetic, logic, or data manipulation operation was zero;otherwise, the Z bit is cleared.

10.1.6.4 Negative (N).

The negative bit is set if the result of the last arithmetic, logic, or data manipulation operation was negative;otherwise, the N bit is cleared. A result is said to be negative if its most significant bit is a one.

10.1.6.5 Interrupt Mask (I).

The I interrupt mask bit is set either by hardware or program instruction to disable (mask) all maskable interrupt sources (both external and internal).

10.1.6.6 Half Carry (H).

The half carry bit is set to a logic one when a carry occurs between bits 3 and 4 of the arithmetic logic unit during an ADD, ABA, or ADC instruction;otherwise, the H bit is cleared.

10.1.6.7 X Interrupt Mask (X).

The X interrupt mask bit is set only by hardware ($\overline{\text{RESET}}$ or $\overline{\text{XIRQ}}$ acknowledge);and it is cleared only by program instruction (TAP or RTI).

10.1.6.8 Stop Disable (S).

The stop disable bit is set to disable the STOP instruction, and cleared to enable the STOP instruction. The S bit is program controlled. The STOP instruction is treated as no operation (NOP) if the S bit is set.

10.2 Addressing Modes

Six addressing modes can be used to reference memory;they include:immediate, direct, extended, indexed (with either of two 16-bit index registers and an 8-bit offset), inherent and relative. Some instructions require and additional byte before the opcode to accommodate a multi-page opcode map;this byte is called a prebyte.

The following paragraphs provide a description of each addressing mode plus a discussion of the prebyte. In these descriptions the term effective address is used to indicate the address in memory from which the argument is fetched or stored, or from which execution is to proceed.

10.2.1 Immediate Addressing

In the immediate addressing mode, the actual argument is contained in the byte (s) immediately following the instruction, where the number of bytes matches the size of the register. These are two, three, or four (if prebyte is required) byte instructions.

10.2.2 Direct Addressing

In the direct addressing mode (sometimes called zero page addressing), the least significant byte of the operand address is contained in a single byte following the opcode and the most significant byte is assumed to be $00. Direct addressing allows the user to access $0000 through $00FF using two byte instructions and execution time is reduced by eliminating the additional memory access. In most applications, this 256-byte area is reserved for frequently referenced data. In the TMP68HC11A8, software can configure the memory map so that internal RAM, and/or internal registers, or external memory space can occupy these addresses.

10.2.3 Extended Addressing

In the extended addressing mode, the second and third bytes (following the opcode) contain the absolute address of the operand. These are three or four (if prebyte is required) byte instructions:one or two for the opcode, and two for the effective address.

10.2.4 Indexed Addressing

In the indexed addressing mode, one of the index registers (X or Y) is used in calculating the effective address. In this case, the effective address is variable and depends on two factors:1) the current contents of the index register (X or Y) being used, and 2) the 8-bit unsigned offset contained in the instruction. This addressing mode allows referencing any memory location in the 64K byte address space. These are usually two or three (if prebyte is required) byte instructions, the opcode plus the 8-bit offset.

10.2.5 Inherent Addressing

In the inherent addressing mode, all of the information is contained in the opcode. The operands (if any) are registers and no memory reference is required. These are usually one or two byte instructions.

10.2.6 Relative Addressing

The relative addressing mode is used for branch instructions. If the branch condition is true, the contents of the 8-bit signed byte following the opcode (the offset) is added to the contents of the program counter to form the effective branch address;otherwise, control proceeds to the next instruction. These are usually two byte instructions.

10.2.7 Prebyte

In order to expand the number of instructions used in the TMP68HC11A8, a prebyte instruction has been added to certain instructions. The instructions affected are usually associated with index register Y. The instruction opcodes which do not require a prebyte could be considered as page 1 of the overall opcode map. The remaining opcodes could be considered as pages 2, 3, and 4 of the opcode map and would require a prebyte;$18 for page 2, $1A for page 3, and $CD for page 4.

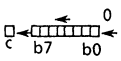
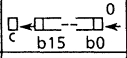
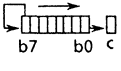
10.3 INSTRUCTION SET

The central processing unit (CPU) in the TMP68HC11A8 is basically a proper extension of the MC6801 CPU. In addition to its ability to execute all M6800 and M6801 instructions, the TMP68HC11A8 CPU has a paged operation code (opcode) map with a total of 91 new opcodes. Major functional additions include a second 16-bit index register (Y register), two types of 16-by-16 divide instruction, STOP and WAIT instructions, and bit manipulation instructions.

Table 10.1 shows all TMP68HC11A8 instructions in all possible addressing modes. For each instruction, the operand construction is shown as well as the total number of machine code bytes and execution time in CPU E-clock cycles. Notes are provided at the end of Table 10.1 which explain the letters in the Operand and Execution Time columns for some instructions. Definitions of "Special Ops" found in the Boolean Expression column are found in Figure 10.2.

Table 10.2 through 10.8 prdovide a detailed description of the information present on the address bus, data bus, and the read/write ($R/\overline{W}$) line during each cycle of each instruction. The information is useful in comparing actual with expected results during debug of both software and hardware as the program is executed. The information is categorized in groups according to addressing mode and number of cycles per instruction. In general, instructions with the same address mode and number of cycles execute in the same manner. Exceptions are indicated in the table.

Table 10.1 TMP68HC11A8 Instructions, Addressing Modes, and Execution Times (Sheet 1 of 7)

Source Form (s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes
				Opcode	Operand (s)				S X H I N Z V C
ABA	Add Accumulators	A + B→A	INH	1B		1	2	2-1	- - ↕ - ↕ ↕ ↕ ↕
ABX	Add B to X	IX + 00:B→IX	INH	3A		1	3	2-2	- - - - - - - -
ABY	Add B to Y	IY + 00:B→IY	INH	18 3A		2	4	2-4	- - - - - - - -
ADCA (opr)	Add with Carry to A	A + M + C→A	A IMM	89	ii	2	2	3-1	- - ↕ - ↕ ↕ ↕ ↕
			A DIR	99	dd	2	3	4-1	
			A EXT	B9	hh ll	3	4	5-2	
			A IND, X	A9	ff	2	4	6-2	
			A IND, Y	18 A9	ff	3	5	7-2	
ADCB (opr)	Add with Carry to B	B + M + C→B	B IMM	C9	ii	2	2	3-1	- - ↕ - ↕ ↕ ↕ ↕
			B DIR	D9	dd	2	3	4-1	
			B EXT	F9	hh ll	3	4	5-2	
			B IND, X	E9	ff	2	4	6-2	
			B IND, Y	18 E9	ff	3	5	7-2	
ADDA (opr)	Add Memory to A	A + M→A	A IMM	8B	ii	2	2	3-1	- - ↕ - ↕ ↕ ↕ ↕
			A DIR	9B	dd	2	3	4-1	
			A EXT	BB	hh ll	3	4	5-2	
			A IND, X	AB	ff	2	4	6-2	
			A IND, Y	18 AB	ff	3	5	7-2	
ADDB (opr)	Add Memory to B	B + M→B	B IMM	CB	ii	2	2	3-1	- - ↕ - ↕ ↕ ↕ ↕
			B DIR	DB	dd	2	3	4-1	
			B EXT	FB	hh ll	3	4	5-2	
			B IND, X	EB	ff	2	4	6-2	
			B IND, Y	18 EB	ff	3	5	7-2	
ADDD (opr)	Add 16-Bit to D	D + M:M + 1 →D	IMM	C3	jj kk	3	4	3-3	- - - - ↕ ↕ ↕ ↕
			DIR	D3	dd	2	5	4-7	
			EXT	F3	hh ll	3	6	5-10	
			IND, X	E3	ff	2	6	6-10	
			IND, Y	18 E3	ff	3	7	7-8	
ANDA (opr)	AND A with Memory	A • M→A	A IMM	84	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			A DIR	94	dd	2	3	4-1	
			A EXT	B4	hh ll	3	4	5-2	
			A IND, X	A4	ff	2	4	6-2	
			A IND, Y	18 A4	ff	3	5	7-2	
ANDB (opr)	AND B with Memory	B • M→B	B IMM	C4	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			B DIR	D4	dd	2	3	4-1	
			B EXT	F4	hh ll	3	4	5-2	
			B IND, X	E4	ff	2	4	6-2	
			B IND, Y	18 E4	ff	3	5	7-2	
ASL (opr)	Arithmetic Shift Left		EXT	78	hh ll	3	6	5-8	- - - - ↕ ↕ ↕ ↕
			IND, X	68	ff	2	6	6-3	
			IND, Y	18 68	ff	3	7	7-3	
ASLA			A INH	48		1	2	2-1	
ASLB			B INH	58		1	2	2-1	
ASLD	Arithmetic Shift Left Double		INH	05		1	3	2-2	- - - - ↕ ↕ ↕ ↕
ASR (opr)	Arithmetic Shift Right		EXT	77	hh ll	3	6	5-8	- - - - ↕ ↕ ↕ ↕
			IND, X	67	ff	2	6	6-3	
			IND, Y	18 67	ff	3	7	7-3	
ASRA			A INH	47		1	2	2-1	
ASRB			B INH	57		1	2	2-1	
BCC (rel)	Branch if Carry Clear	?C = 0	REL	24	rr	2	3	8-1	- - - - - - - -
BCLR (opr) (msk)	Clear Bit (s)	M • ($\overline{mm}$) →M	DIR	15	dd mm	3	6	4-10	- - - - ↕ ↕ 0 -
			IND, X	1D	ff mm	3	7	6-13	
			IND, Y	18 1D	ff mm	4	8	7-10	
BCS (rel)	Branch if Carry Set	?C = 1	REL	25	rr	2	3	8-1	- - - - - - - -
BEQ (rel)	Branch if = Zero	?Z = 1	REL	27	rr	2	3	8-1	- - - - - - - -

* Cycle-by-cycle number provides a reference to Table 10.2 through 10.8 which detail cycle-by-cycle operation.

Example: Table 10.1 cycle-by-cycle column reference number 2-4 equals table 10.2 line item 2-4.

Table 10.1 TMP68HC11A8 Instructions, Addressing Modes, and Execution Times (Sheet 2 of 7)

Source Form (s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes
				Opcode	Operand (s)				S X H I N Z V C
BGE (rel)	Branch if ≧ Zero	?N⊕V = 0	REL	2C	rr	2	3	8-1	- - - - - - - -
BGT (rel)	Branch if > Zero	?Z + (N⊕V) = 0	REL	2E	rr	2	3	8-1	- - - - - - - -
BHI (rel)	Branch if Higher	?C + Z = 0	REL	22	rr	2	3	8-1	- - - - - - - -
BHS (rel)	Branch if Higher or Same	?C = 0	REL	24	rr	2	3	8-1	- - - - - - - -
BITA (opr)	Bit (s) Test A with Memory	A • M	A IMM	85	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			A DIR	95	dd	2	3	4-1	
			A EXT	B5	hh ll	3	4	5-2	
			A IND, X	A5	ff	2	4	6-2	
			A IND, Y	18 A5	ff	3	5	7-2	
BITB (opr)	Bit (s) Test B with Memory	B • M	B IMM	C5	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			B DIR	D5	dd	2	3	4-1	
			B EXT	F5	hh ll	3	4	5-2	
			B IND, X	E5	ff	2	4	6-2	
			B IND, Y	18 E5	ff	3	5	7-2	
BLE (rel)	Branch if ≦ Zero	?Z + (N⊕V) = 1	REL	2F	rr	2	3	8-1	- - - - - - - -
BLO (rel)	Branch if Lower	?C = 1	REL	25	rr	2	3	8-1	- - - - - - - -
BLS (rel)	Branch if Lower or Same	?C + Z = 1	REL	23	rr	2	3	8-1	- - - - - - - -
BLT (rel)	Branch if < Zero	?N⊕V = 1	REL	2D	rr	2	3	8-1	- - - - - - - -
BMI (rel)	Branch if Minus	?N = 1	REL	2B	rr	2	3	8-1	- - - - - - - -
BNE (rel)	Branch if Not = Zero	?Z = 0	REL	26	rr	2	3	8-1	- - - - - - - -
BPL (rel)	Branch if Plus	?N = 0	REL	2A	rr	2	3	8-1	- - - - - - - -
BRA (rel)	Branch Always	?1 = 1	REL	20	rr	2	3	8-1	- - - - - - - -
BRCLR (opr) (msk) (rel)	Branch if Bit (s) Clear	?M • mm = 0	DIR	13	dd mm rr	4	6	4-11	- - - - - - - -
			IND, X	1F	ff mm rr	4	7	6-14	
			IND, Y	18 1F	ff mm rr	5	8	7-11	
BRN (rel)	Branch Never	?1 = 0	REL	21	rr	2	3	8-1	- - - - - - - -
BRSET (opr) (msk) (rel)	Branch if Bit (s) Set	? ($\overline{M}$) • mm = 0	DIR	12	dd mm rr	4	6	4-11	- - - - - - - -
			IND, X	1E	ff mm rr	4	7	6-14	
			IND, Y	18 1E	ff mm rr	5	8	7-11	
BSET (opr) (msk)	Set Bit (s)	M + mm→M	DIR	14	dd mm	3	6	4-10	- - - - ↕ ↕ 0 -
			IND, X	1C	ff mm	3	7	6-13	
			IND, Y	18 1C	ff mm	4	8	7-10	
BSR (rel)	Branch to Subroutine	See Special Ops	REL	8D	rr	2	6	8-2	- - - - - - - -
BVC (rel)	Branch if Overflow Clear	?V = 0	REL	28	rr	2	3	8-1	- - - - - - - -
BVS (rel)	Branch if Overflow Set	?V = 1	REL	29	rr	2	3	8-1	- - - - - - - -
CBA	Compare A to B	A-B	INH	11		1	2	2-1	- - - - ↕ ↕ ↕ ↕
CLC	Clear Carry Bit	0→C	INH	0C		1	2	2-1	- - - - - - - 0
CLI	Clear Interrupt Mask	0→I	INH	0E		1	2	2-1	- - - 0 - - - -
CLR (opr)	Clear Memory Byte	0→M	EXT	7F	hh ll	3	6	5-8	- - - - 0 1 0 0
			IND, X	6F	ff	2	6	6-3	
			IND, Y	18 6F	ff	3	7	7-3	
CLRA	Clear Accumulator A	0→A	A INH	4F		1	2	2-1	- - - - 0 1 0 0
CLRB	Clear Accumulator B	0→B	B INH	5F		1	2	2-1	- - - - 0 1 0 0
CLV	Clear Overflow Flag	0→V	INH	0A		1	2	2-1	- - - - - - 0 -
CMPA (opr)	Compare A to Memory	A-M	A IMM	81	ii	2	2	3-1	- - - - ↕ ↕ ↕ ↕
			A DIR	91	dd	2	3	4-1	
			A EXT	B1	hh ll	3	4	5-2	
			A IND, X	A1	ff	2	4	6-2	
			A IND, Y	18 A1	ff	3	5	7-2	

* Cycle-by-cycle number provides a reference to Table 10.2 through 10.8 which detail cycle-by-cycle operation.

Example: Table 10.1 cycle-by-cycle column reference number 2-4 equals table 10.2 line item 2-4.

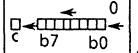
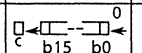
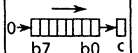
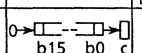
Table 10.1 TMP68HC11A8 Instructions, Addressing Modes, and Execution Times (Sheet 3 of 7)

Source Form (s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand (s)	Bytes	Cycle	Cycle by Cycle*	Condition Codes S X H I N Z V C
CMPB (opr)	Compare B to Memory	B-M	B IMM	C1	ii	2	2	3-1	- - - - ↕ ↕ ↕ ↕
			B DIR	D1	dd	2	3	4-1	
			B EXT	F1	hh ll	3	4	5-2	
			B IND, X	E1	ff	2	4	6-2	
			B IND, Y	18 E1	ff	3	5	7-2	
COM (opr)	1's Complement Memory Byte	$FF-M→M	EXT	73	hh ll	3	6	5-8	- - - - ↕ ↕ 0 1
			IND, X	63	ff	2	6	6-3	
			IND, Y	18 63	ff	3	7	7-3	
COMA	1's Complement A	$FF-A→A	A INH	43		1	2	2-1	- - - - ↕ ↕ 0 1
COMB	1's Complement B	$FF-B→B	B INH	53		1	2	2-1	- - - - ↕ ↕ 0 1
CPD (opr)	Compare D to Memory 16-Bit	D-M:M + 1	IMM	1A 83	jj kk	4	5	3-5	- - - - ↕ ↕ ↕ ↕
			DIR	1A 93	dd	3	6	4-9	
			EXT	1A B3	hh ll	4	7	5-11	
			IND, X	1A A3	ff	3	7	6-11	
			IND, Y	CD A3	ff	3	7	7-8	
CPX (opr)	Compare X to Memory 16-Bit	IX-M:M + 1	IMM	8C	jj kk	3	4	3-3	- - - - ↕ ↕ ↕ ↕
			DIR	9C	dd	2	5	4-7	
			EXT	BC	hh ll	3	6	5-10	
			IND, X	AC	ff	2	6	6-10	
			IND, Y	CD AC	ff	3	7	7-8	
CPY (opr)	Compare Y to Memory 16-Bit	IY-M:M + 1	IMM	18 8C	jj kk	4	5	3-5	- - - - ↕ ↕ ↕ ↕
			DIR	18 9C	dd	3	6	4-9	
			EXT	18 BC	hh ll	4	7	5-11	
			IND, X	1A AC	ff	3	7	6-11	
			IND, Y	18 AC	ff	3	7	7-8	
DAA	Decimal Adjust A	Adjust Sum to BCD	INH	19		1	2	2-1	- - - - ↕ ↕ ↕ ↕
DEC (opr)	Decrement Memory Byte	M-1→M	EXT	7A	hh ll	3	6	5-8	- - - - ↕ ↕ ↕ -
			IND, X	6A	ff	2	6	6-3	
			IND, Y	18 6A	ff	3	7	7-3	
DECA	Decrement Accumulator A	A-1→A	A INH	4A		1	2	2-1	- - - - ↕ ↕ ↕ -
DECB	Decrement Accumulator B	B-1→B	B INH	5A		1	2	2-1	- - - - ↕ ↕ ↕ -
DES	Decrement Stack Pointer	SP-1→SP	INH	34		1	3	2-3	- - - - - - - -
DEX	Decrement Index Register X	IX-1→IX	INH	09		1	3	2-2	- - - - - ↕ - -
DEY	Decrement Index Register Y	IY-1→IY	INH	18 09		2	4	2-4	- - - - - ↕ - -
EORA (opr)	Exclusive OR A with Memory	A ⊕ M→A	A IMM	88	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			A DIR	98	dd	2	3	4-1	
			A EXT	B8	hh ll	3	4	5-2	
			A IND, X	A8	ff	2	4	6-2	
			A IND, Y	18 A8	ff	3	5	7-2	
EORB (opr)	Exclusive OR B with Memory	B ⊕ M→B	B IMM	C8	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			B DIR	D8	dd	2	3	4-1	
			B EXT	F8	hh ll	3	4	5-2	
			B IND, X	E8	ff	2	4	6-2	
			B IND, Y	18 E8	ff	3	5	7-2	
FDIV	Fractional Divide 16 by 16	D/IX→IX;r→D	INH	03		1	41	2-17	- - - - - ↕ ↕ ↕
IDIV	Integer Divide 16 by 16	D/IX→IX;r→D	INH	02		1	41	2-17	- - - - - ↕ 0 ↕
INC (opr)	Increment Memory Byte	M + 1→M	EXT	7C	hh ll	3	6	5-8	- - - - ↕ ↕ ↕ -
			IND, X	6C	ff	2	6	6-3	
			IND, Y	18 6C	ff	3	7	7-3	
INCA	Increment Accumulator A	A + 1→A	A INH	4C		1	2	2-1	- - - - ↕ ↕ ↕ -
INCB	Increment Accumulator B	B + 1→B	B INH	5C		1	2	2-1	- - - - ↕ ↕ ↕ -
INS	Increment Stack Pointer	SP + 1→SP	INH	31		1	3	2-3	- - - - - - - -

* Cycle-by-cycle number provides a reference to Table 10.2 through 10.8 which detail cycle-by-cycle operation.

Example: Table 10.1 cycle-by-cycle column reference number 2-4 equals table 10.2 line item 2-4.

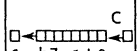
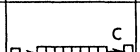
Table 10.1 TMP68HC11A8 Instructions, Addressing Modes, and Execution Times (Sheet 4 of 7)

Source Form (s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes
				Opcode	Operand (s)				S X H I N Z V C
INX	Increment Index Register X	IX + 1→IX	INH	08		1	3	2-2	- - - - - ↕ - -
INY	Increment Index Register Y	IY + 1→IY	INH	18 08		2	4	2-4	- - - - - ↕ - -
JMP (opr)	Jump	See Special Ops	EXT	7E	hh ll	3	3	5-1	- - - - - - - -
			IND, X	6E	ff	2	3	6-1	
			IND, Y	18 6E	ff	3	4	7-1	
JSR (opr)	Jump to Subroutine	See Special Ops	DIR	9D	dd	2	5	4-8	- - - - - - - -
			EXT	BD	hh ll	3	6	5-12	
			IND, X	AD	ff	2	6	6-12	
			IND, Y	18 AD	ff	3	7	7-9	
LDAA (opr)	Load Accumulator A	M→A	A IMM	86	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			A DIR	96	dd	2	3	4-1	
			A EXT	B6	hh ll	3	4	5-2	
			A IND, X	A6	ff	2	4	6-2	
			A IND, Y	18 A6	ff	3	5	7-2	
LDAB (opr)	Load Accumulator B	M→B	B IMM	C6	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			B DIR	D6	dd	2	3	4-1	
			B EXT	F6	hh ll	3	4	5-2	
			B IND, X	E6	ff	2	4	6-2	
			B IND, Y	18 E6	ff	3	5	7-2	
LDD (opr)	Load Double Accumulator D	M→A, M + 1→B	IMM	CC	jj kk	3	3	3-2	- - - - ↕ ↕ 0 -
			DIR	DC	dd	2	4	4-3	
			EXT	FC	hh ll	3	5	5-4	
			IND, X	EC	ff	2	5	6-6	
			IND, Y	18 EC	ff	3	6	7-6	
LDS (opr)	Load Stack Pointer	M:M + 1→SP	IMM	8E	jj kk	3	3	3-2	- - - - ↕ ↕ 0 -
			DIR	9E	dd	2	4	4-3	
			EXT	BE	hh ll	3	5	5-4	
			IND, X	AE	ff	2	5	6-6	
			IND, Y	18 AE	ff	3	6	7-6	
LDX (opr)	Load Index Register X	M:M + 1→IX	IMM	CE	jj kk	3	3	3-2	- - - - ↕ ↕ 0 -
			DIR	DE	dd	2	4	4-3	
			EXT	FE	hh ll	3	5	5-4	
			IND, X	EE	ff	2	5	6-6	
			IND, Y	CD EE	ff	3	6	7-6	
LDY (opr)	Load Index Register Y	M:M + 1→IY	IMM	18 CE	jj kk	4	4	3-4	- - - - ↕ ↕ 0 -
			DIR	18 DE	dd	3	5	4-5	
			EXT	18 FE	hh ll	4	6	5-6	
			IND, X	1A EE	ff	3	6	6-7	
			IND, Y	18 EE	ff	3	6	7-6	
LSL (opr)	Logical Shift Left		EXT	78	hh ll	3	6	5-8	- - - - ↕ ↕ ↕ ↕
			IND, X	68	ff	2	6	6-3	
			IND, Y	18 68	ff	3	7	7-3	
LSLA			A INH	48		1	2	2-1	
LSLB			B INH	58		1	2	2-1	
LSLD	Logical Shift Left Double		INH	05		1	3	2-2	- - - - ↕ ↕ ↕ ↕
LSR (opr)	Logical Shift Right		EXT	74	hh ll	3	6	5-8	- - - - 0 ↕ ↕ ↕
			IND, X	64	ff	2	6	6-3	
			IND, Y	18 64	ff	3	7	7-3	
LSRA			A INH	44		1	2	2-1	
LSRB			B INH	54		1	2	2-1	
LSRD	Logical Shift Right Double		INH	04		1	3	2-2	- - - - 0 ↕ ↕ ↕
MUL	Multiply 8 by 8	A × B→D	INH	3D		1	10	2-13	- - - - - - - ↕

* Cycle-by-cycle number provides a reference to Table 10.2 through 10.8 which detail cycle-by-cycle operation.

Example: Table 10.1 cycle-by-cycle column reference number 2-4 equals table 10.2 line item 2-4.

Table 10.1 TMP68HC11A8 Instructions, Addressing Modes, and Execution Times (Sheet 5 of 7)

Source Form (s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand (s)	Bytes	Cycle	Cycle by Cycle*	Condition Codes S X H I N Z V C
NEG (opr)	2's Complement Memory Byte	0 – M→M	EXT	70	hh ll	3	6	5-8	- - - - ↕ ↕ ↕ ↕
			IND, X	60	ff	2	6	6-3	
			IND, Y	18 60	ff	3	7	7-3	
NEGA	2's Complement A	0 – A→A	A INH	40		1	2	2-1	- - - - ↕ ↕ ↕ ↕
NEGB	2's Complement B	0 – B→B	B INH	50		1	2	2-1	- - - - ↕ ↕ ↕ ↕
NOP	No Operation	No Operation	INH	01		1	2	2-1	- - - - - - - -
ORAA (opr)	OR Accumulator A (Inclusive)	A + M→A	A IMM	8A	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			A DIR	9A	dd	2	3	4-1	
			A EXT	BA	hh ll	3	4	5-2	
			A IND, X	AA	ff	2	4	6-2	
			A IND, Y	18 AA	ff	3	5	7-2	
ORAB (opr)	OR Accumulator B (Inclusive)	B + M→B	B IMM	CA	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			B DIR	DA	dd	2	3	4-1	
			B EXT	FA	hh ll	3	4	5-2	
			B IND, X	EA	ff	2	4	6-2	
			B IND, Y	18 EA	ff	3	5	7-2	
PSHA	Push A onto Stack	A→Stk, SP = SP – 1	A INH	36		1	3	2-6	- - - - - - - -
PSHB	Push B onto Stack	B→Stk, SP = SP – 1	B INH	37		1	3	2-6	- - - - - - - -
PSHX	Push X onto Stack (Lo First)	IX→Stk, SP = SP – 2	INH	3C		1	4	2-7	- - - - - - - -
PSHY	Push Y onto Stack (Lo First)	IY→Stk, SP = SP – 2	INH	18 3C		2	5	2-8	- - - - - - - -
PULA	Pull A from Stack	SP = SP + 1, A←Stk	A INH	32		1	4	2-9	- - - - - - - -
PULB	Pull B from Stack	SP = SP + 1, B←Stk	B INH	33		1	4	2-9	- - - - - - - -
PULX	Pull X from Stack (Hi First)	SP = SP + 2, IX←Stk	INH	38		1	5	2-10	- - - - - - - -
PULY	Pull Y from Stack (Hi First)	SP = SP + 2, IY←Stk	INH	18 38		2	6	2-11	- - - - - - - -
ROL (opr)	Rotate Left	 c b7 ← b0	EXT	79	hh ll	3	6	5-8	- - - - ↕ ↕ ↕ ↕
			IND, X	69	ff	2	6	6-3	
			IND, Y	18 69	ff	3	7	7-3	
ROLA			A INH	49		1	2	2-1	
ROLB			B INH	59		1	2	2-1	
ROR (opr)	Rotate Right	 c b7 → b0	EXT	76	hh ll	3	6	5-8	- - - - ↕ ↕ ↕ ↕
			IND, X	66	ff	2	6	6-3	
			IND, Y	18 66	ff	3	7	7-3	
RORA			A INH	46		1	2	2-1	
RORB			B INH	56		1	2	2-1	
RTI	Return from Interrupt	See Special Ops	INH	3B		1	12	2-14	↕ ↕ ↕ ↕ ↕ ↕ ↕ ↕
RTS	Return from Subroutine	See Special Ops	INH	39		1	5	2-12	- - - - - - - -
SBA	Subtract B from A	A – B→A	INH	10		1	2	2-1	- - - - ↕ ↕ ↕ ↕
SBCA (opr)	Subtract with Carry from A	A – M – C→A	A IMM	82	ii	2	2	3-1	- - - - ↕ ↕ ↕ ↕
			A DIR	92	dd	2	3	4-1	
			A EXT	B2	hh ll	3	4	5-2	
			A IND, X	A2	ff	2	4	6-2	
			A IND, Y	18 A2	ff	3	5	7-2	
SBCB (opr)	Subtract from Carry from B	B – M – C→B	B IMM	C2	ii	2	2	3-1	- - - - ↕ ↕ ↕ ↕
			B DIR	D2	dd	2	3	4-1	
			B EXT	F2	hh ll	3	4	5-2	
			B IND, X	E2	ff	2	4	6-2	
			B IND, Y	18 E2	ff	3	5	7-2	
SEC	Set Carry	1→C	INH	0D		1	2	2-1	- - - - - - - - 1
SEI	Set Interrupt Mask	1→I	INH	0F		1	2	2-1	- - - 1 - - - -
SEV	Set Overflow Flag	1→V	INH	0B		1	2	2-1	- - - - - - 1 -

* Cycle-by-cycle number provides a reference to Table 10.2 through 10.8 which detail cycle-by-cycle operation.

Example: Table 10.1 cycle-by-cycle column reference number 2-4 equals table 10.2 line item 2-4.

Table 10.1 TMP68HC11A8 Instructions, Addressing Modes, and Execution Times (Sheet 6 of 7)

Source Form (s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes
				Opcode	Operand (s)				S X H I N Z V C
STAA (opr)	Store Accumulator A	A→M	A DIR	97	dd	2	3	4-2	- - - - ↕ ↕ 0 -
			A EXT	B7	hh ll	3	4	5-3	
			A IND, X	A7	ff	2	4	6-5	
			A IND, Y	18 A7	ff	3	5	7-5	
STAB (opr)	Store Accumulator B	B→M	B DIR	D7	dd	2	3	4-2	- - - - ↕ ↕ 0 -
			B EXT	F7	hh ll	3	4	5-3	
			B IND, X	E7	ff	2	4	6-5	
			B IND, Y	18 E7	ff	3	5	7-5	
STD (opr)	Store Accumulator D	A→M, B→M + 1	DIR	DD	dd	2	4	4-4	- - - - ↕ ↕ 0 -
			EXT	FD	hh ll	3	5	5-5	
			IND, X	ED	ff	2	5	6-8	
			IND, Y	18 ED	ff	3	6	7-7	
STOP	Stop Internal Clocks		INH	CF		1	2	2-1	- - - - - - - -
STS (opr)	Store Stack Pointer	SP→M:M + 1	DIR	9F	dd	2	4	4-4	- - - - ↕ ↕ 0 -
			EXT	BF	hh ll	3	5	5-5	
			IND, X	AF	ff	2	5	6-8	
			IND, Y	18 AF	ff	3	6	7-7	
STX (opr)	Store Index Register X	IX→M:M + 1	DIR	DF	dd	2	4	4-4	- - - - ↕ ↕ 0 -
			EXT	FF	hh ll	3	5	5-5	
			IND, X	EF	ff	2	5	6-8	
			IND, Y	CD EF	ff	3	6	7-7	
STY (opr)	Store Index Register Y	IY→M:M + 1	DIR	18 DF	dd	3	5	4-6	- - - - ↕ ↕ 0 -
			EXT	18 FF	hh ll	4	6	5-7	
			IND, X	1A EF	ff	3	6	6-9	
			IND, Y	18 EF	ff	3	6	7-7	
SUBA (opr)	Subtract Memory from A	A – M→A	A IMM	80	ii	2	2	3-1	- - - - ↕ ↕ ↕ ↕
			A DIR	90	dd	2	3	4-1	
			A EXT	B0	hh ll	3	4	5-2	
			A IND, X	A0	ff	2	4	6-2	
			A IND, Y	18 A0	ff	3	5	7-2	
SUBB (opr)	Subtract Memory from B	B – M→B	B IMM	C0	ii	2	2	3-1	- - - - ↕ ↕ ↕ ↕
			B DIR	D0	dd	2	3	4-1	
			B EXT	F0	hh ll	3	4	5-2	
			B IND, X	E0	ff	2	4	6-2	
			B IND, Y	18 E0	ff	3	5	7-2	
SUBD (opr)	Subtract Memory from D	D – M:M + 1→D	IMM	83	jj kk	3	4	3-3	- - - - ↕ ↕ ↕ ↕
			DIR	93	dd	2	5	4-7	
			EXT	B3	hh ll	3	6	5-10	
			IND, X	A3	ff	2	6	6-10	
			IND, Y	18 A3	ff	3	7	7-8	
SWI	Software Interrupt	See Special Ops	INH	3F		1	14	2-15	- - - 1 - - - -
TAB	Transfer A to B	A→B	INH	16		1	2	2-1	- - - - ↕ ↕ 0 -
TAP	Transfer A to CC Register	A→CCR	INH	06		1	2	2-1	↕ ↕ ↕ ↕ ↕ ↕ ↕ ↕
TBA	Transfer B to A	B→A	INH	17		1	2	2-1	- - - - ↕ ↕ 0 -
TEST	TEST (Only in Test Modes)	Address Bus Counts	INH	00		1	**	2-20	- - - - - - - -
TPA	Transfer CC Register to A	CCR→A	INH	07		1	2	2-1	- - - - - - - -
TST (opr)	Test for Zero or Minus	M – 0	EXT	7D	hh ll	3	6	5-9	- - - - ↕ ↕ 0 0
			IND, X	6D	ff	2	6	6-4	
			IND, Y	18 6D	ff	3	7	7-4	
TSTA		A – 0	A INH	4D		1	2	2-1	- - - - ↕ ↕ 0 0
TSTB		B – 0	B INH	5D		1	2	2-1	- - - - ↕ ↕ 0 0
TSX	Transfer Stack Pointer to X	SP + 1→IX	INH	30		1	3	2-3	- - - - - - - -
TSY	Transfer Stack Pointer to Y	SP + 1→IY	INH	18 30		2	4	2-5	- - - - - - - -

* Cycle-by-cycle number provides a reference to Table 10.2 through 10.8 which detail cycle-by-cycle operation.

Example: Table 10.1 cycle-by-cycle column reference number 2-4 equals table 10.2 line item 2-4.

Table 10.1 TMP68HC11A8 Instructions, Addressing Modes, and Execution Times (Sheet 7 of 7)

Source Form (s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes
				Opcode	Operand (s)				S X H I N Z V C
TXS	Transfer X to Stack Pointer	IX – 1→SP	INH	35		1	3	2-2	- - - - - - - -
TYS	Transfer Y to Stack Pointer	IY – 1→SP	INH	18 35		2	4	2-4	- - - - - - - -
WAI	Wait for Interrupt	Stack Regs & WAIT	INH	3E		1	***	2-16	- - - - - - - -
XGDX	Exchange D with X	IX→D, D→IX	INH	8F		1	3	2-2	- - - - - - - -
XGDY	Exchange D with Y	IY→D, D→IY	INH	18 8F		2	4	2-4	- - - - - - - -

* Cycle-by-cycle number provides a reference to Table 10.2 through 10.8 which detail cycle-by-cycle operation.
Example: Table 10.1 cycle-by-cycle column reference number 2-4 equals table 10.2 line item 2-4.

** Infinity or Until Reset Occurs

*** 12 Cycles are used beginning with the opcode fetch. A wait state is entered which remains in effect for an integer number of MPU E-clock cycles (n) until an interupt is recognized. Finally, two additional cyces are used to fetch the appropriate interrupt vector (14 + n total).

dd = 8-Bit Direct Address ($0000-$00FF) (High Byte Assumed to be $00)
ff = 8-Bit Positive Offset $00 (0) to $FF (255) (Is Added to Index)
hh = High Order Byte of 16-Bit Extended Address
ii = One Byte of Immediate Data
jj = High Order Byte of 16-Bit Immediate Data
kk = Low Order Byte of 16-Bit Immediate Data
ll = Low Order Byte of 16-Bit Extended Address
mm = 8-Bit Bit Mask (Set Bits to be Affected)
rr = Signed Relative Offset $80 (– 128) to $7F (+ 127)
(Offset Relative to the Address Following the Machine Code Offset Byte)

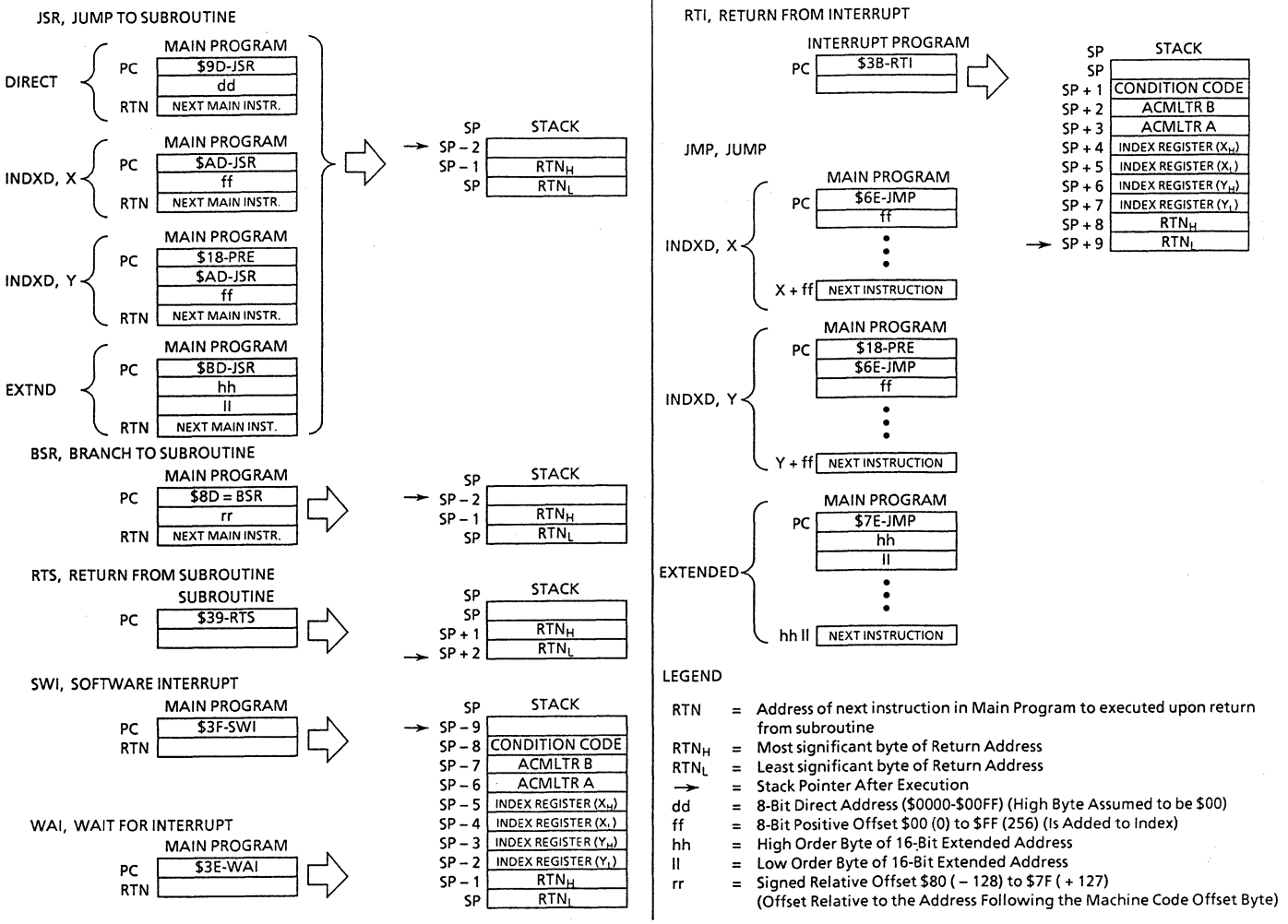


Figure 10.2 Special Operations

Table 10.2 Cycle-by-Cycle Operation — Inherent Mode (Sheet 1 of 4)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
2-1	ABA, ASLA, ASLB, ASRA, ASRB, CBA, CLC, CLI, CLRA, CLRB, CLV, COMA, COMB, DAA, DECA, DECB, INCA, INCB, LSLA, LSLB, LSRA, LSRB, NEGA, NEGB, NOP, ROLA, ROLB, RORA, RORB, SBA, SEC, SEI, SEV, STOP, TAB, TAP, TBA, TPA, TSTA, TSTB	2	1 2	Opcode Address Opcode Address + 1	1 1	Opcode Irrelevant Data
2-2	ABX, ASLD, DEX, INX, LSLD, LSRD, TXS, XGDX	3	1 2 3	Opcode Address Opcode Address + 1 $FFFF	1 1 1	Opcode Irrelevant Data Irrelevant Data
2-3	DES, INS, TSX	3	1 2 3	Opcode Address Opcode Address + 1 Previous SP Value	1 1 1	Opcode Irrelevant Data Irrelevant Data
2-4	ABY, DEY, INY, TYS, XGDY	4	1 2 3 4	Opcode Address Opcode Address + 1 Opcode Address + 2 $FFFF	1 1 1 1	Opcode (Page Select Byte) ($18) Opcode (Second Byte) Irrelevant Data Irrelevant Data
2-5	TSY	4	1 2 3 4	Opcode Address Opcode Address + 1 Opcode Address + 2 Stack Pointer	1 1 1 1	Opcode (Page Select Byte) ($18) Opcode (Second Byte) ($30) Irrelevant Data Irrelevant Data
2-6	PSHA, PSHB	3	1 2 3	Opcode Address Opcode Address + 1 Stack Pointer	1 1 0	Opcode Irrelevant Data Accumulator Data
2-7	PSHX	4	1 2 3 4	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer – 1	1 1 0 0	Opcode ($3C) Irrelevant Data IXL (Low Byte) to Stack IXH (High Byte) to Stack
2-8	PSHY	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Opcode Address + 2 Stack Pointer Stack Pointer – 1	1 1 1 0 0	Opcode (Page Select Byte) ($18) Opcode (Second Byte) ($3C) Irrelevant Data IXL (Low Byte) to Stack IXH (High Byte) to Stack

* The reference number is given to provide a cross-reference to Table 10.1

Table 10.2 Cycle-by-Cycle Operation — Inherent Mode (Sheet 2 of 4)

Reference Number*	Address Mode and Instructions	Cycle	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
2-9	PULA, PULB	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	1	Irrelevant Data
			4	Stack Pointer + 1	1	Operand Data from Stack
2-10	PULX	5	1	Opcode Address	1	Opcode ($38)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	1	Irrelevant Data
			4	Stack Pointer + 1	1	IXH (High Byte) from Stack
			5	Stack Pointer + 2	1	IXL (Low Byte) from Stack
2-11	PULY	6	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($38)
			3	Opcode Address + 2	1	Irrelevant Data
			4	Stack Pointer	1	Irrelevant Data
			5	Stack Pointer + 1	1	IYH (High Byte) from Stack
			6	Stack Pointer + 2	1	IYL (Low Byte) from Stack
2-12	RTS	5	1	Opcode Address	1	Opcode ($39)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	1	Irrelevant Data
			4	Stack Pointer + 1	1	Address of Next Instruction (High Byte)
			5	Stack Pointer + 2	1	Address of Next Instruction (Low Byte)
2-13	MUL	10	1	Opcode Address	1	Opcode ($3D)
			2	Opcode Address + 1	1	Irrelevant Data
			3	$FFFF	1	Irrelevant Data
			4	$FFFF	1	Irrelevant Data
			5	$FFFF	1	Irrelevant Data
			6	$FFFF	1	Irrelevant Data
			7	$FFFF	1	Irrelevant Data
			8	$FFFF	1	Irrelevant Data
			9	$FFFF	1	Irrelevant Data
			10	$FFFF	1	Irrelevant Data
2-14	RTI	12	1	Opcode Adderess	1	Opcode ($3B)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	1	Irrelevant Data
			4	Stack Pointer + 1	1	Condition Code Register from Stack
			5	Stack Pointer + 2	1	B Accumulator from Stack
			6	Stack Pointer + 3	1	A Accumulator from Stack
			7	Stack Pointer + 4	1	IXH (High Byte) from Stack
			8	Stack Pointer + 5	1	IXL (Low Byte) from Stack
			9	Stack Pointer + 6	1	IYH (High Byte) from Stack

* The reference number is given to provide a cross-reference to Table 10.1

Table 10.2 Cycle-by-Cycle Operation — Inherent Mode (Sheet 3 of 4)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
2-14 (Continued)	RTI	12	10	Stack Pointer + 7	1	IYL (Low Byte) from Stack
			11	Stack Pointer + 8	1	Address of Next Instruction (High Byte)
			12	Stack Pointer + 9	1	Address of Next Instruction (Low Byte)
2-15	SWI	14	1	Opcode Adderess	1	Opcode ($3F)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	0	Return Address (Low Byte)
			4	Stack Pointer – 1	0	Return Address (High Byte)
			5	Stack Pointer – 2	0	IYL (Low Byte) to Stack
			6	Stack Pointer – 3	0	IYH (High Byte) to Stack
			7	Stack Pointer – 4	0	IXL (Low Byte) to Stack
			8	Stack Pointer – 5	0	IXH (High Byte) to Stack
			9	Stack Pointer – 6	0	A Accumulator to Stack
			10	Stack Pointer – 7	0	B Accumulator to Stack
			11	Stack Pointer – 8	0	Condition Code Register to Stack
			12	Stack Pointer – 8	1	Irrelevant Data
			13	Address of SWI Vector (First Location)	1	SWI Service Routine Address (High Byte)
			14	Address of Vector + 1 (Second Location)	1	SWI Service Routine Address (Low Byte)
2-16	WAI	14 + n	1	Opcode Adderess	1	Opcode ($3E)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	0	Return Address (Low Byte)
			4	Stack Pointer – 1	0	Return Address (High Byte)
			5	Stack Pointer – 2	0	IYL (Low Byte) to Stack
			6	Stack Pointer – 3	0	IYH (High Byte) to Stack
			7	Stack Pointer – 4	0	IXL (Low Byte) to Stack
			8	Stack Pointer – 5	0	IXH (High Byte) to Stack
			9	Stack Pointer – 6	0	A Accumulator to Stack
			10	Stack Pointer – 7	0	B Accumulator to Stack
			11	Stack Pointer – 8	0	Condition Code Register to Stack
			12 to n + 12	Stack Pointer – 8	1	Irrelevant Data
			n + 13	Address of Vector (First Location)	1	Service Routine Address (High Byte)
			n + 14	Address of Vector + 1 (Second Location)	1	Service Routine Address (Low Byte)
2-17	FDIV, IDIV	41	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Irrelevant Data
			3 – 41	$FFFF	1	Irrelevant Data

* The reference number is given to provide a cross-reference to Table 10.1

Table 10.2 Cycle-by-Cycle Operation — Inherent Mode (Sheet 4 of 4)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
2-18	Page 1 Illegal Opcodes	15	1	Opcode Address	1	Opcode (Illegal)
			2	Opcode Address + 1	1	Irrelevant Data
			3	$FFFF	1	Irrelevant Data
			4	Stack Pointer	0	Return Address (Low Byte)
			5	Stack Pointer – 1	0	Return Address (High Byte)
			6	Stack Pointer – 2	0	IYL (Low Byte) to Stack
			7	Stack Pointer – 3	0	IYH (High Byte) to Stack
			8	Stack Pointer – 4	0	IXL (Low Byte) to Stack
			9	Stack Pointer – 5	0	IXH (High Byte) to Stack
			10	Stack Pointer – 6	0	A Accumulator
			11	Stack Pointer – 7	0	B Accumulator
			12	Stack Pointer – 8	0	Condition Code Register to Stack
			13	Stack Pointer – 8	1	Irrelevant Data
			14	Address of SWI Vector (First Location)	1	Service Routine Address (High Byte)
			15	Address of Vector + 1 (Second Location)	1	Service Routine Address (Low Byte)
2-19	Page 2, 3, or4 Illegal Opcodes	16	1	Opcode Adderess	1	Opcode (Legal Page Select)
			2	Opcode Address + 1	1	Opcode (Illegal Second Byte)
			3	Opcode Address + 2	1	Irrelevant Data
			4	$FFFF	1	Irrelevant Data
			5	Stack Pointer	0	Return Address (Low Byte)
			6	Stack Pointer – 1	0	Return Address (High Byte)
			7	Stack Pointer – 2	0	IYL (Low Byte) to Stack
			8	Stack Pointer – 3	0	IYH (High Byte) to Stack
			9	Stack Pointer – 4	0	IXL (Low Byte) to Stack
			10	Stack Pointer – 5	0	IXH (High Byte) to Stack
			11	Stack Pointer – 6	0	A Accumulator
			12	Stack Pointer – 7	0	B Accumulator
			13	Stack Pointer – 8	0	Condition Code Register to Stack
			14	Stack Pointer – 8	1	Irrelevant Data
			15	Address of SWI Vector (First Location)	1	SWI Service Routine Address (High Byte)
			16	Address of Vector + 1 (Second Location)	1	SWI Service Routine Address (Low Byte)
2-20	TEST	Infinite	1	Opcode Adderess	1	Opcode ($00)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Opcode Address + 1	1	Irrelevant Data
			4	Opcode Address + 2	1	Irrelevant Data
			5-n	Previous Address + 1	1	Irrelevant Data

* The reference number is given to provide a cross-reference to Table 10.1

Table 10.3 Cycle-by-Cycle Operation — Immediate Mode

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
3-1	ADCA, ADCB, ADDA, ADDB, ANDA, ANDB, BITA, BITB, CMPA, CMPB, EORA, EORB, LDAA, LDAB, ORAA, ORAB, SBCA, SBCB, SUBA, SUBB	2	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Data
3-2	LDD, LDS, LDX	3	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Data (High Byte)
			3	Opcode Address + 2	1	Operand Data (Low Byte)
3-3	ADDD, CPX, SUBD	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Data (High Byte)
			3	Opcode Address + 2	1	Operand Data (Low Byte)
			4	$FFFF	1	Irrelevant Data
3-4	LDY	4	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($EC)
			3	Opcode Address + 2	1	Operand Data (High Byte)
			4	Opcode Address + 3	1	Operand Data (Low Byte)
3-5	CPD, CPY	5	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Operand Data (High Byte)
			4	Opcode Address + 3	1	Operand Data (Low Byte)
			5	$FFFF	1	Irrelevant Data

* The reference number is given to provide a cross-reference to Table 10.1

Table 10.4 Cycle-by-Cycle Operation — Direct Mode (Sheet 1 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
4-1	ADCA, ADCB, ADDA, ADDB, ANDA, ANDB, BITA, BITB, CMPA, CMPB, EORA, EORB, LDAA, LDAB, ORAA, ORAB, SBCA, SBCB, SUBA, SUBB	3	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	1	Operand Data
4-2	STAA, STAB	3	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	0	Data from Accumulator
4-3	LDD, LDS, LDX	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	1	Operand Data (High Byte)
			4	Operand Address + 1	1	Operand Data (Low Byte)
4-4	STD, STS, STX	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	0	Register Data (High Byte)
			4	Operand Address + 1	0	Register Data (Low Byte)
4-5	LDY	5	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($DE)
			3	Opcode Address + 2	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			4	Operand Address	1	Operand Data (High Byte)
			5	Operand Address + 1	1	Operand Data (Low Byte)
4-6	STY	5	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($DE)
			3	Opcode Address + 2	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			4	Operand Address	0	Register Data (High Byte)
			5	Operand Address + 1	0	Register Data (Low Byte)
4-7	ADDD, CPX, SUBD	5	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	1	Operand Data (High Byte)
			4	Operand Address + 1	1	Operand Data (Low Byte)
			5	$FFFF	1	Irrelevant Data

* The reference number is given to provide a cross-reference to Table 10.1

Table 10.4 Cycle-by-Cycle Operation — Direct Mode (Sheet 2 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
4-8	JSR	5	1	Opcode Address	1	Opcode ($9D)
			2	Opcode Address + 1	1	Subroutine Address (Low Byte) (High Byte Assumed to be $00)
			3	Subroutine Address	1	First Subroutine Opcode
			4	Stack Pointer	0	Return Address (Low Byte)
			5	Stack Pointer – 1	0	Return Address (High Byte)
4-9	CPD, CPY	6	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			4	Operand Address	1	Operand Data (High Byte)
			5	Operand Address + 1	1	Operand Data (Low Byte)
			6	$FFFF	1	Irrelevant Data
4-10	BCLR, BSET	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	1	Original Operand Data
			4	Opcode Address + 2	1	Mask Byte
			5	$FFFF	1	Irrelevant Data
			6	Operand Address	0	Result Operand Data
4-11	BRCLR, BRSET	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	1	Original Operand Data
			4	Opcode Address + 2	1	Mask Byte
			5	Opcode Address + 3	1	Branch Offset
			6	$FFFF	1	Irrelevant Data

* The reference number is given to provide a cross-reference to Table 10.1

Table 10.5 Cycle-by-Cycle Operation — Extended Mode (Sheet 1 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
5-1	JMP	3	1	Opcode Address	1	Opcode ($7E)
			2	Opcode Address + 1	1	Jump Address (High Byte)
			3	Opcode Address + 2	1	Jump Address (Low Byte)
5-2	ADCA, ADCB, ADDA, ADDB, ANDA, ANDB, BITA, BITB, CMPA, CMPB, EORA, EORB, LDAA, LDAB, ORAA, ORAB, SBCA, SBCB, SUBA, SUBB	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	1	Operand Data
5-3	STAA, STAB	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	0	Accumulator Data
5-4	LDD, LDS, LDX	5	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	1	Operand Data (High Byte)
			5	Operand Address + 1	1	Operand Data (Low Byte)
5-5	STD, STS, STX	5	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	0	Register Data (High Byte)
			5	Operand Address + 1	0	Register Data (Low Byte)
5-6	LDY	6	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($FF)
			3	Opcode Address + 2	1	Operand Address (High Byte)
			4	Opcode Address + 3	1	Operand Address (Low Byte)
			5	Operand Address	1	Operand Data (High Byte)
			6	Operand Address + 1	1	Operand Data (Low Byte)
5-7	STY	6	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($FF)
			3	Opcode Address + 2	1	Operand Address (High Byte)
			4	Opcode Address + 3	1	Operand Address (Low Byte)
			5	Operand Address	0	Register Data (High Byte)
			6	Operand Address + 1	0	Register Data (Low Byte)
5-8	ASL, ASR, CLR, COM, DEC, INC, LSL, LSR, NEG, ROL, ROR	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	1	Original Operand Data

* The reference number is given to provide a cross-reference to Table 10.1

Table 10.5 Cycle-by-Cycle Operation — Extended Mode (Sheet 2 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
5-8 (Continued)	ASL, ASR, CLR, COM, DEC, INC, LSL, LSR, NEG, ROL, ROR	6	5	$FFFF	1	Irrelevant Data
			6	Operand Address	0	Result Operand Data
5-9	TST	6	1	Opcode Address	1	Opcode ($7D)
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	1	Original Operand Data
			5	$FFFF	1	Irrelevant Data
			6	$FFFF	1	Irrelevant Data
5-10	ADDD, CPX, SUBD	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	1	Operand Data (High Byte)
			5	Operand Address + 1	1	Operand Data (Low Byte)
			6	$FFFF	1	Irrelevant Data
5-11	CPD, CPY	7	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Operand Address (High Byte)
			4	Opcode Address + 3	1	Operand Address (Low Byte)
			5	Operand Address	1	Operand Data (High Byte)
			6	Operand Address + 1	1	Operand Data (Low Byte)
			7	$FFFF	1	Irrelevant Data
5-12	JSR	6	1	Opcode Address	1	Opcode ($BD)
			2	Opcode Address + 1	1	Subroutine Address (High Byte)
			3	Opcode Address + 2	1	Subroutine Address (Low Byte)
			4	Subroutine Address	1	First Opcode in Subroutine
			5	Stack Pointer	0	Return Address (Low Byte)
			6	Stack Pointer – 1	0	Return Address (High Byte)

* The reference number is given to provide a cross-reference to Table 10.1

Table 10.6 Cycle-by-Cycle Operation — Indexed X Mode (Sheet 1 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
6-1	JMP	3	1	Opcode Address	1	Opcode ($6E)
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
6-2	ADCA, ADCB, ADDA, ADDB, ANDA, ANDB, BITA, BITB, CMPA, CMPB, EORA, EORB, LDAA, LDAB, ORAA, ORAB, SBCA, SBCB, SUBA, SUBB	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Operand Data
6-3	ASL, ASR, CLR, COM, DEC, INC, LSL, LSR, NEG, ROL, ROR	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Original Operand Data
			5	$FFFF	1	Irrelevant Data
			6	(IX) + Offset	0	Result Operand Data
6-4	TST	6	1	Opcode Address	1	Opcode ($6D)
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Original Operand Data
			5	$FFFF	1	Irrelevant Data
			6	$FFFF	1	Irrelevant Data
6-5	STAA, STAB	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	0	Accumulator Data
6-6	LDD, LDS, LDX	5	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Operand Data (High Byte)
			5	(IX) + Offset + 1	1	Operand Data (Low Byte)
6-7	LDY	6	1	Opcode Address	1	Opcode (Page Select Byte) ($1A)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($EE)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IX) + Offset	1	Operand Data (High Byte)
			6	(IX) + Offset + 1	1	Operand Data (Low Byte)
6-8	STD, STS, STX	5	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data

* The reference number is given to provide a cross-reference to Table 10.1

Table 10.6 Cycle-by-Cycle Operation — Indexed X Mode (Sheet 2 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
6-8 (Continued)	STD, STS, STX	5	4	(IX) + Offset	0	Register Data (High Byte)
			5	(IX) + Offset + 1	0	Register Data (Low Byte)
6-9	STY	6	1	Opcode Address	1	Opcode (Page Select Byte) ($1A)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($EF)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IX) + Offset	0	Register Data (High Byte)
			6	(IX) + Offset + 1	0	Register Data (Low Byte)
6-10	ADDD, CPX, SUBD	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Register Data (High Byte)
			5	(IX) + Offset + 1	1	Register Data (Low Byte)
			6	$FFFF	1	Irrelevant Data
6-11	CPD, CPY	7	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IX) + Offset	1	Operand Data (High Byte)
			6	(IX) + Offset + 1	1	Operand Data (Low Byte)
			7	$FFFF	1	Irrelevant Data
6-12	JSR	6	1	Opcode Address	1	Opcode ($AD)
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	First Opcode in Subroutine
			5	Stack Pointer	0	Return Address (Low Byte)
			6	Stack Pointer – 1	0	Return Address (High Byte)
6-13	BCLR, BSET	7	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Original Operand Data
			5	Opcode Address + 2	1	Mask Byte
			6	$FFFF	1	Irrelevant Data
			7	(IX) + Offset	0	Result Operand Data
6-14	BRCLR, BRSET	7	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Original Operand Data
			5	Opcode Address + 2	1	Mask Byte
			6	Opcode Address + 3	1	Branch Offset
			7	$FFFF	1	Irrelevant Data

* The reference number is given to provide a cross-reference to Table 10.1

Table 10.7 Cycle-by-Cycle Operation — Indexed Y Mode (Sheet 1 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
7-1	JMP	4	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($6E)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
7-2	ADCA, ADCB, ADDA, ADDB, ANDA, ANDB, BITA, BITB, CMPA, CMPB, EORA, EORB, LDAA, LDAB, ORAA, ORAB, SBCA, SBCB, SUBA, SUBB	5	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Operand Data
7-3	ASL, ASR, CLR, COM, DEC, INC, LSL, LSR, NEG, ROL, ROR	7	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Original Operand Data
			6	$FFFF	1	Irrelevant Data
			7	(IY) + Offset	0	Result Operand Data
7-4	TST	7	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($6D)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Original Operand Data
			6	$FFFF	1	Irrelevant Data
			7	$FFFF	1	Irrelevant Data
7-5	STAA, STAB	5	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	0	Accumulator Data
7-6	LDD, LDS, LDX, LDY	6	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Operand Data (High Byte)
			6	(IY) + Offset + 1	1	Operand Data (Low Byte)
7-7	STD, STS, STX, STY	6	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data

* The reference number is given to provide a cross-reference to Table 10.1

Table 10.7 Cycle-by-Cycle Operation — Indexed Y Mode (Sheet 2 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
7-7 (Continued)	STD, STS, STX, STY	6	5	(IY) + Offset	0	Register Data (High Byte)
			6	(IY) + Offset + 1	0	Register Data (Low Byte)
7-8	ADDD, CPD, CPX, CPY, SUBD	7	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Operand Data (High Byte)
			6	(IY) + Offset + 1	1	Operand Data (Low Byte)
			7	$FFFF	1	Irrelevant Data
7-9	JSR	7	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($AD)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	First Opcode in Subroutine
			6	Stack Pointer	0	Return Address (Low Byte)
			7	Stack Pointer – 1	0	Return Address (High Byte)
7-10	BCLR, BSET	8	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Original Operand Data
			6	Opcode Address + 3	1	Mask Byte
			7	$FFFF	1	Irrevant Data
			8	(IY) + Offset	0	Result Operand Data
7-11	BRCLR, BRSET	8	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Original Operand Data
			6	Opcode Address + 3	1	Mask Byte
			7	Opcode Address + 4	1	Branch Offset
			8	$FFFF	1	Irrelevant Data

* The reference number is given to provide a cross-reference to Table 10.1

Table 10.8 Cycle-by-Cycle Operation – Relative Mode

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
8-1	BCC, BCS, BEQ, BGE, BGT, BHI, BHS, BLE, BLO, BLS, BLT, BMI, BNE, BPL, BRA, BRN, BVC, BVS	3	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Branch Offset
			3	$FFFF	1	Irrelevant Data
8-2	BSR	6	1	Opcode Address	1	Opcode ($8D)
			2	Opcode Address + 1	1	Branch Offset
			3	$FFFF	1	Irrelevant Data
			4	Subroutine Address	1	Opcode of Next Instruction
			5	Stack Pointer	0	Return Address (Low Byte)
			6	Stack Pointer – 1	0	Return Address (High Byte)

* The reference number is given to provide a cross-reference to Table 10.1

11. ELECTRICAL SPECIFICATIONS

11.1 MAXIMUM RATINGS

Raing	Symbol	Value	Unit
Supply Voltage	V_{DD}	− 0.3 to + 7.0	V
Input Voltage	V_{in}	− 0.3 to + 7.0	V
Operating Temperature Range TMP68HC11A8	T_A	T_L to T_H − 40 to 85	℃
Storage Temperature Range	T_{stg}	− 55 to 150	℃
Current Drain per Pin* Excluding V_{DD}, V_{SS}, V_{RH}, and V_{RL}	I_D	25	mA

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. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (e.g., either GND or V_{DD}).

* One pin at a time, observing maximum power dissipation limits.

11.2 THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance	θ_{JA}		℃/W
Plastic 52-Pin Quad Pack (PLCC)		TBD	
Plastic 48-Pin Dual-In-Line		TBD	

11.3 POWER CONSIDERATIONS

The avarage chip-junction temperature, T_J, in ℃ can be obtained from:

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

where:

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

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

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

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

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

$$K = PD \cdot (TA + 273℃) + \theta_{JA} \cdot P_D^2 \quad (3)$$

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.

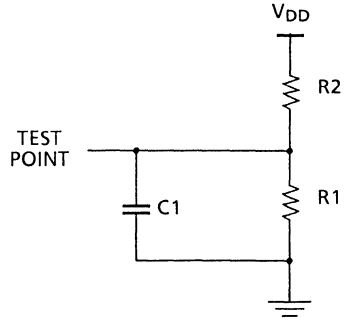
11.4 DC ELECTRICAL CHARACTERISTICS (V_{DD}=5.0 Vdc±10%,V_{SS}=0 Vdc,T_A=T_L to T_H, unless otherwise noted)

Characteristic		Symbol	Min	Max	Unit
Output Voltage I_{Load} = ± 10.0μA (see Note 1)	All Outputs All Outputs Except $\overline{RESET}$ and MODA	VOL VOH	– V_{DD} – 0.1	0.1 –	V
Output High Voltage ILoad = -0.8mA, VDD = 4.5V (see Note 1)	All Outputs Except RESET, XTAL, and MODA	V_{OH}	V_{DD} – 0.8	–	V
Output Low Voltage I_{Load} = 1.6mA	All Outputs Except XTAL	V_{OL}	–	0.4	V
Input High Voltage	All Inputs Except $\overline{RESET}$ $\overline{RESET}$	V_{IH}	0.7 × V_{DD} 0.8 × V_{DD}	V_{DD} V_{DD}	V
Input Low Voltage	All Inputs	V_{IL}	V_{SS}	0.2 × V_{DD}	V
I/O Ports, Three-State Leakage V_{in} = V_{IH} or V_{IL}	PA7, PC0-PC7, PD0-PD5, AS/STRA, MODA LIR, RESET	I_{OZ}	–	± 10	μA
Input Current (see Note 2) V_{in} = V_{DD} or V_{SS} V_{in} = V_{DD} or V_{SS}	 PA0-PA2,$\overline{IRQ}$ $\overline{XIRQ}$ MODB/V_{STBY}	I_{in}	 – –	 ± 1 ± 10	μA
RAM Standby Voltage	Powerdown	V_{SB}	4.0	V_{DD}	V
RAM Standby Current	Powerdown	I_{SB}	–	20	μA
Total Supply Current(see Note 3) RUN: Single Chip Expanded Multiplexed WAIT: All Peripheral Functions Shut Down Single-Chip Mode Expanded Multiplexed Mode STOP: No Clocks, Single-Chip Mode		I_{DD} W_{IDD} S_{IDD} 	 – – – – –	 15 27 6 10 100	 mA mA mA mA μA
Input Capacitance PA7, PC0-PC7, PD0-PD5, AS/STRA, MODA/$\overline{LIR}$, $\overline{RESET}$	PA0-PA2, PE0-PE7, $\overline{IRQ}$, $\overline{XIRQ}$, EXTAL	C_{in}	– –	8 12	pF
Power Dissipation	Single Chip Mode Expanded Multiplexed Mode	P_D	– –	85 150	mW

Notes :

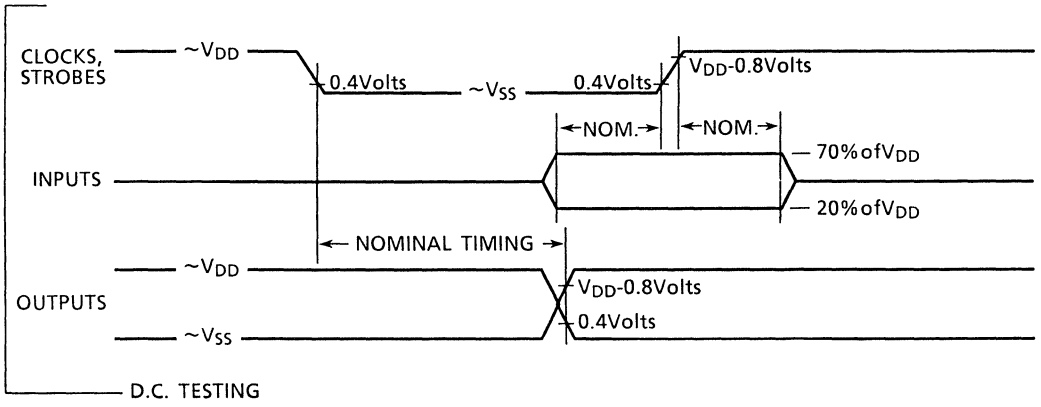
1. V_{OH} specification for $\overline{RESET}$ and MODA is not applicable because they are open-drain pins.
   V_{OH} specification not applicable to ports C and D in wire-OR mode.
2. See A/D specification for leakage current for port E.
3. All ports configured as inputs,
   $V_{IL} \leqq 0.2$ V,
   $V_{IH} \geqq V_{DD} - 0.2$ V,
   No dc loads,

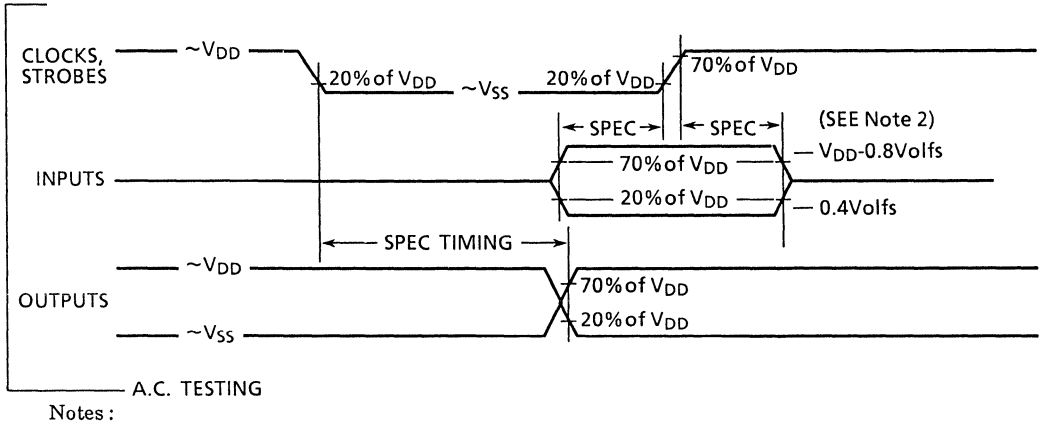
EXTAL is driven with a square wave, and
t_{cyc}=476.5 ns.



Equivalent Test Load[1]

Pins	R1	R2	C
PA3-PA7 PB0-PB7 PC0-PC7 PD0, PD5 E, AS, R/W	3.26kΩ	2.38kΩ	90pF
PD1-PD4	3.26kΩ	2.38kΩ	200pF





Notes:
1. Full test loads are applied during all dc electrical and ac timing measurements.
2. During ac timing measurements, inputs are driven to 0.4 volts and V_{DD}-0.8 volts while timing measurements are taken at the 20% and 70% of V_{DD} points.

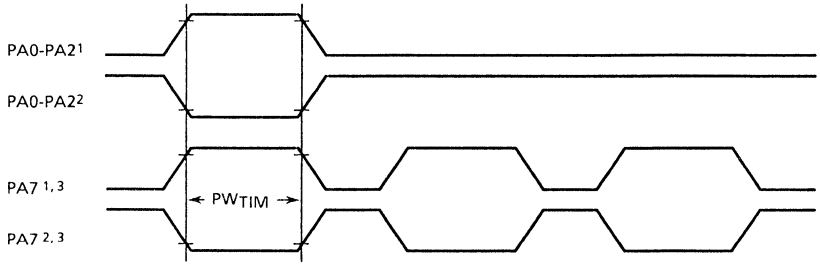
Figure 11.1 Test Methods

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

Characteristic	Symbol	1.0MHz		2.0MHz		2.1MHz		Unit
		Min	Max	Min	Max	Min	Max	
Frequency of Operation	f_O	dc	1.0	dc	2.0	dc	2.1	MHz
E Clock Period	t_{cyc}	1000	–	500	–	476	–	ns
Crystal Frequency	f_{XTAL}	–	4.0	–	8.0	–	8.4	MHz
External Oscillator Frequency	$4f_O$	dc	4.0	dc	8.0	dc	8.4	MHz
Processor Control Setup Time(see Figures 11.3, 11.5, and 11.6) t_{PCS} = 1/4 t_{cyc}-50 ns	t_{PCS}	200	–	75	–	69	–	ns
Reset Input Pulse Width (see Note 1 and Figure 11.3) (To Guarantee External Reset Vector)	PW_{RSTL}	8	–	8	–	8	–	t_{cyc}
(Minimum Input Time; May be Preempted by Internal Reset)		1	–	1	–	1	–	
Mode Programming Setup Time (see Figure 11.3)	t_{MPS}	2	–	2	–	2	–	tcyc
Mode Programming Hold Time (see Figure 11.3)	t_{MPH}	0	–	0	–	0	–	ns
Interrupt Pulse Width, IRQ Edge Sensitive Mode (see Figures 11.4 and 11.6) PW_{IRQ} = t_{cyc} + 20 ns	PW_{IRQ}	1020	–	520	–	496	–	ns
Wait Recovery Startup Time (See Figure 11.5)	t_{WRS}	–	4	–	4	–	4	tcyc
Timer Pulse Width Input Capture, Pulse Accumulator Input (see Figure 11.2) PW_{TIM} = t_{cyc} + 20 ns	PW_{TIM}	1020	–	520	–	496	–	ns

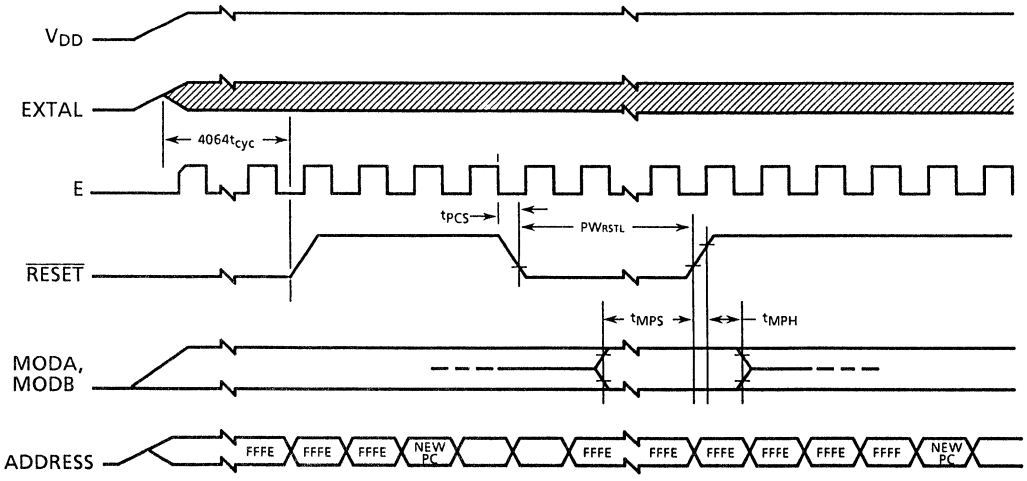
Note : 1. $\overline{\text{RESET}}$ will be recognized during the first clock cycle it is held low. Internal circuity then drives the pin low for four clock cycles, releases the pin, and samples the pin level two cycles later to detemine the source of the interrupt. See SECTION9 RESETS, INTERRUPT, AND LOW POWER MODES for details.

2. All timing is shown with respect to 20% V_{DD} and 70% V_{DD} unless otherwise noted.



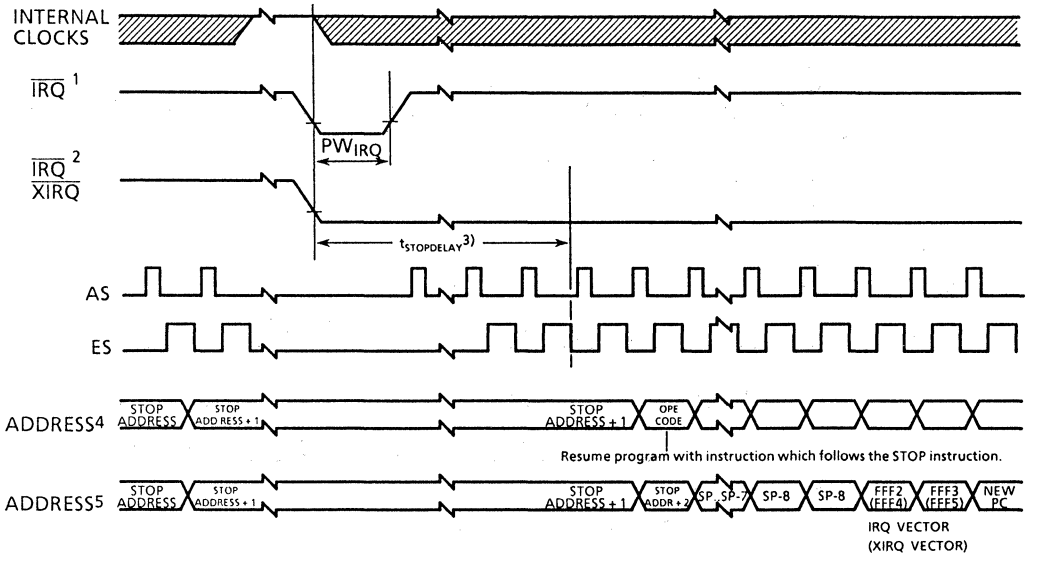
Notes :
1. Rising edge sensitive input
2. Falling edge sensitive input
3. Maximum pulse accumulator clocking rate is E frequency divided by 2.

Figure 11.2 Timer Inputs Timing Diagram



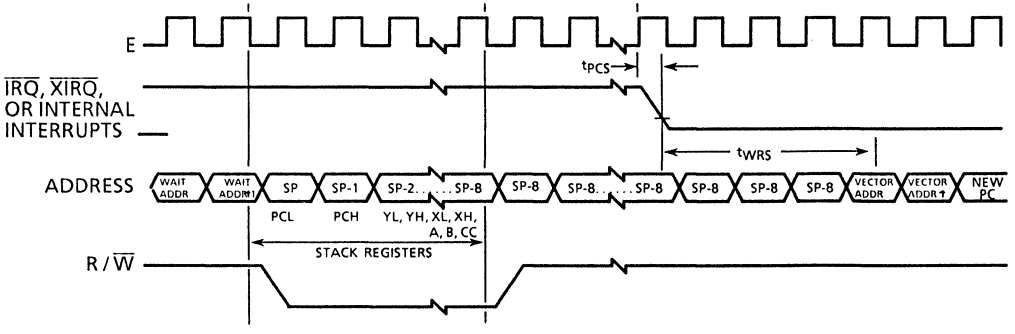
Note : Refer to Table 9.7 for pin states during $\overline{\text{RESET}}$

Figure 11.3 POR and External Reset Timing Diagram



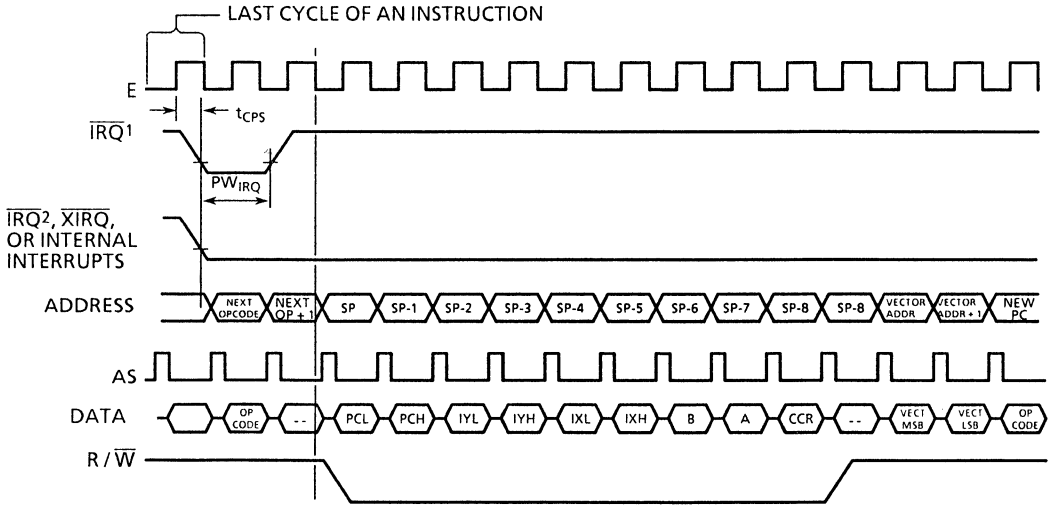
Notes :
1. Edge sensitive $\overline{IRQ}$ pin (IRQE bit = 1)
2. Level sensitive $\overline{IRQ}$ pin (IRQE bit = 0)
3. $t_{STOPDELAY} = 4064t_{cyc}$ if DLY bit = 1 or $4t_{cyc}$ if DLY = 0.
4. $\overline{XIRQ}$ WITH X bit CCR = 1 .
5. IRQ, or ($\overline{XIRQ}$ with X bit in CCR = 0) .

Figure 11.4 STOP Recovery Diagram



Notes :
1 Refer to Table 9.7 for pin states during WAIT.
2 $\overline{RESET}$ will also cause recovery from WAIT.

Figure 11.5 WAIT Recovery from Interrupt Timing Diagram



Notes :
1. Edge sensitive $\overline{IRQ}$ pin (IRQE bit = 1).
2. Level sensitive $\overline{IRQ}$ pin (IRQE bit = 0).

Figure 11.6 Interrupt Timing Diagram

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

Characteristic	Symbol	1.0MHz		2.0MHz		2.1MHz		Unit
		Min	Max	Min	Max	Min	Max	
Frequency of Operation(E Clock Frequency)	f_o	1.0	1.0	2.0	2.0	2.1	2.1	MHz
E Clock Period	t_{cyc}	1000	–	500	–	476	–	ns
Peripheral Data Setup Time (MCU Read of Ports A,C,D, and E) (see Figure 11.8)	t_{PDSU}	100	–	100	–	100	–	ns
Peripheral Data Hole Time (MCU Read of Ports A,C,D, and E) (see Figure 11.8)	t_{PDH}	50	–	50	–	50	–	ns
Delay Time, Peripheral Data Write (see Figures 11.7,11.9,11.12, and 11.13)	t_{PWD}							ns
MCU Write to Port A		–	150	–	150	–	150	
MCU Writes to Ports B,C, and D $t_{PWD} = 14t_{cyc} + 90$ ns		–	340	–	215	–	209	
Input Data Setup Time(Port C) (see Figure 11.10 and 11.11)	t_{IS}	60	–	60	–	60	–	ns
Input Data Hold Time(Port C) (see Figure 11.10 and 11.11)	t_{IH}	100	–	100	–	100	–	ns
Delay Time, E Fall to STRB $t_{DEB} = 1.4t_{cyc} + 100$ ns (see Figures 11.9,11.11,11.12,and 11.13)	t_{DEB}	–	350	–	225	–	219	ns
Setup Time, STRA Asserted to E Fall(see Note 1) (see Figures 11.11,11.12,and 11.13)	t_{AES}	0	–	0	–	0	–	ns
Delay Time, STRA Asserted to Port C Data Output Valid (see Figure 11-13)	t_{PCD}	–	100	–	100	–	100	ns
Hold Time, STRA Negated to Port C Data (see Figure 11.13)	t_{PCH}	10	–	10	–	10	–	ns
Three-State Hold Time (see Figure 11.13)	t_{PCZ}	–	150	–	150	–	150	ns

Notes :

1. If this setup time is met, STRB will acknowledge in the next cycle. If it is not met, the response may be delayed one more cycle.
2. Port and D timing is valid for active drive (CWOM and DWOM bits not set in PIOC and SPCR registers respectively).
3. All timing is shown with respect to 20% V_{DD} and 70% V_{DD} unless otherwise noted.

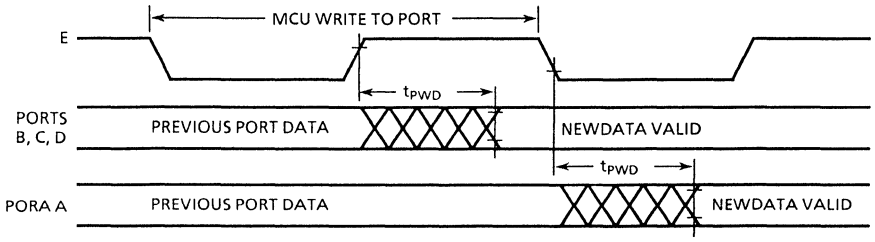
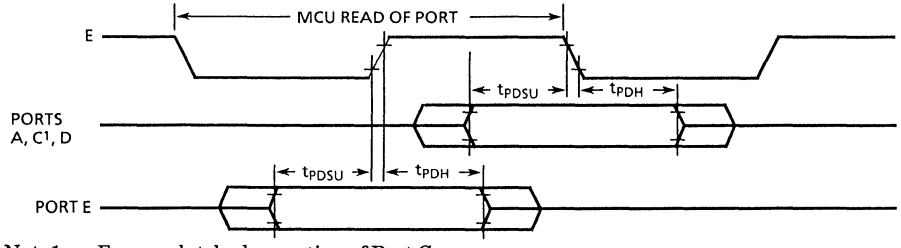


Figure 11.7 Port Write Timing Diagram



Note1 : For non-latched operation of Port C.

Figure 11.8 Port Read Timing Diagram

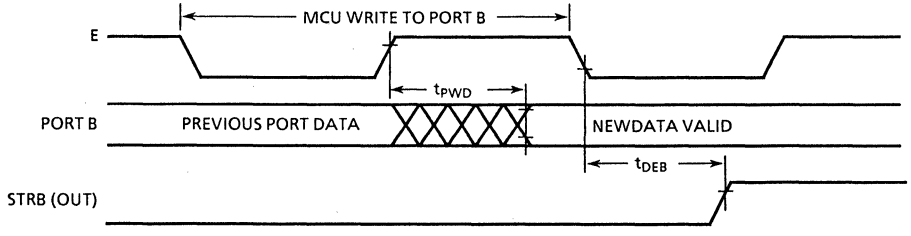


Figure 11.9 Simple Output Strobe Timing Diagram

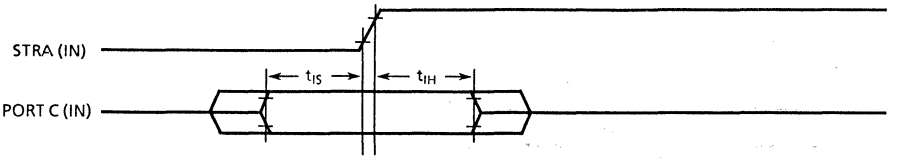
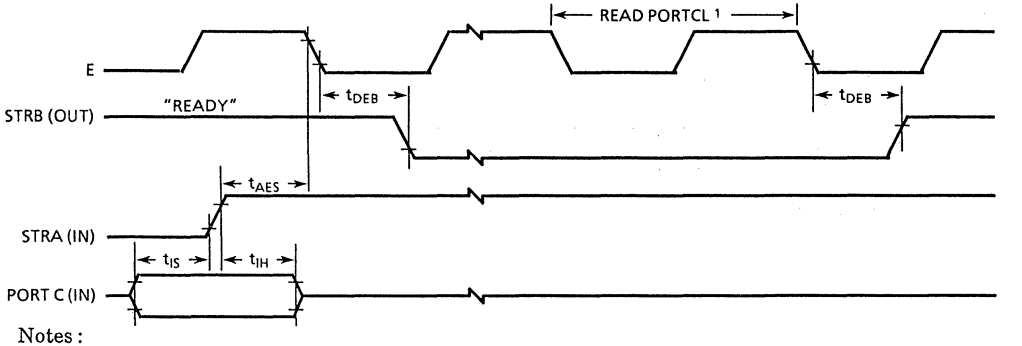
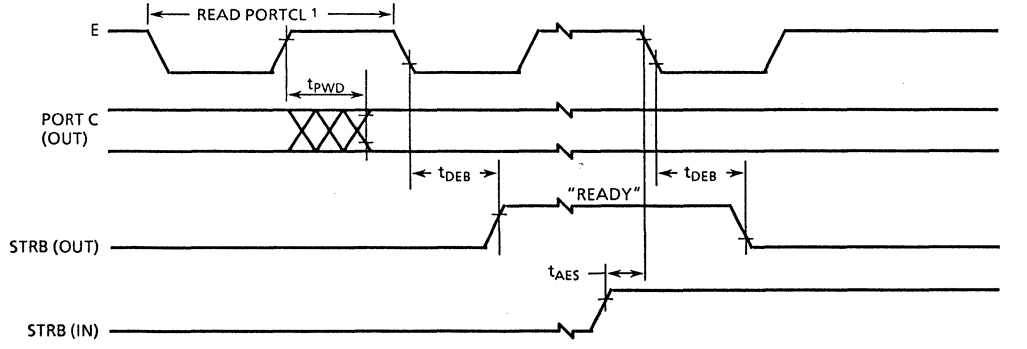


Figure 11.10 Simple Input Strobe Timing Diagram



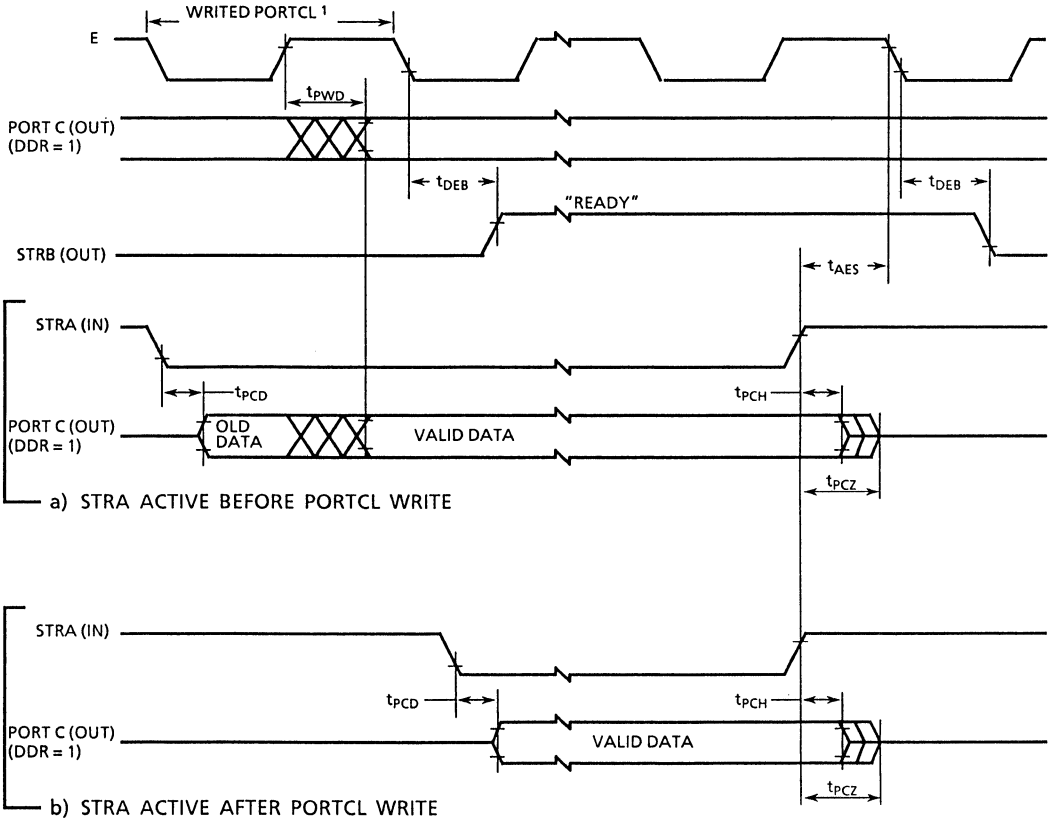
Notes :
1. After reading PIOC with STAF set.
2. Figure shows rising edge STRA (EGA = 1) and high true STRB (INVB = 1)

Figure 11.11 Port C Input Handshake Timing Diagram



Notes :
1. After reading PIOC with STAF set.
2. Figure shows rising edge STRA (EGA = 1) and high true STRB (INVB = 1)

Figure 11.12 Port C Output Handshake Timing Diagram



Notes :
1. After PIOC with STAF set.
2. Figure shows edge STRA (EGA = 1) and high true STRB (INVB = 1)

Figure 11.13 Three-State Variation of Output Handshake Timing Diagram (STRA Enables Output Buffer)

11.7 A/D CONVERTER CHARACTERISTICS(VDD = 5.0 Vdc ± 10%, VSS = 0Vdc, TA = TL to TH, 750kHz ≦ E ≦ 2.1MHz, unless otherwise noted)

Characteristic	Parameter	Min	Absolute	Max	Unit
Resolution	Number of Bits Resolved by the A.D	8	–	–	Bits
Non-Linearity	Maximum Deviation from the Ideal and an Actual A.D Transfer Characteristics	–	–	± 1/2	LSB
Zero Error	Difference Between the Output of an Ideal and an Actual A.D for Zero Input Voltage	–	–	± 1/2	LSB
Full Scale Error	Difference Between the Output of an Ideal A.D for Full-Scale Input Voltage	–	–	± 1/2	LSB
Total Unadjusted Error	Maximum Sum of Non-Linearity, Zero Error, and Full-Scale Error	–	–	± 1/2	LSB
Quantization Error	Uncertainty Due to Converter Resolution	–	–	± 1/2	LSB
Absolute Accuracy	Difference Between the Actual Input Voltage and the Full-Scale Weighted Equivalent of the Binary Output Code, All Error Sources Included	–	–	± 1	LSB
Conversion Range	Analog Input Voltage Range	V_{RL}	–	V_{RH}	V
V_{RH}	Maximum Analog Reference Voltage (see Note 2)	V_{RL}	–	V_{DD} + 0.1	V
V_{RL}	Maximum Analog Reference Voltage (see Note 2)	V_{SS} – 0.1	–	V_{RH}	V
ΔV_R	Maximum Difference between V_{RH} and V_{RL} (see Note 2)	3	–	–	V
Conversion Time	Total Time to Perform a Single Analog-to-Digital Conversion: a. E Clock b. Internal RC Oscillator	– –	32 –	– tcyc + 32	tcyc µs
Monotonicity	Conversion Result Never Decreases with an Increase in Input Voltage and has no Missing Codes		Guaranteed		
Zero Input Reading	Conversion Result when $V_{in} = V_{RL}$	00	–	–	Hex
Full Scale Reading	Conversion Result when $V_{in} = V_{RH}$	–	–	FF	Hex
Sample Acquisition Time	Analog Input Acquisition Sampling Time: a. E Clock b. Internal RC Oscillator	– –	12 –	– 12	tcyc µs
Sample Hold Capacitance	Input Capacitance during Sample PE0-PE7	–	20 (Typ)	–	pF
Input Leakage	Input Leakage on A/D Pins PE0-PE7 V_{RL}, V_{RH}	– –	– –	400 1.0	nA µA

Notes:

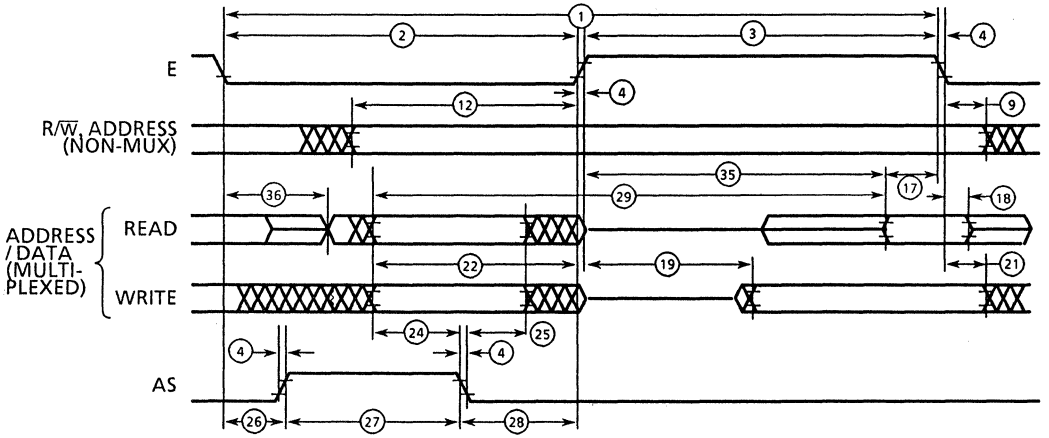
1. Source impedances greater than 10KΩ will adversely affect accuracy, due mainly to input leakage.
2. Performance verified down to 2.5V ΔV_R, but accuracy is tested and guaranteed at ΔV_R = 5V ± 10%

11.8 EXPANSION BUS TIMING (V_{DD}=5.0 Vdc±10%,V_{SS}=0 Vdc,T_A=T_L to T_H,see Figure 11.14)

Num	Characteristic	Symbol	1.05MHz		1.05MHz		1.05MHz		Unit
			Min	Max	Min	Max	Min	Max	
	Frequency of Operation(E Clock Frequency)	f_o	1.0	1.0	2.0	2.0	2.1	2.1	MHz
1	Cycle Time	t_{cyc}	1000	–	500	–	476	–	ns
2	PUlse Width, E Low PW_{EL} = 12t_{cyc}-23 ns	PW_{EL}	477	–	227	–	215	–	ns
3	PUlse Width, E High PW_{EH} = 1/2t_{cyc}-28 ns	PW_{EL}	472	–	222	–	210	–	ns
4	E and AS Rise and fall Time	t_r, t_f	–	20	–	20	–	20	ns
9	Address Hold Time t_{AV} = 1/8t_{cyc}-29.5ns see Note 1(a)	t_{AH}	95.5	–	33	–	30	–	ns
12	Non-Muxed Address Valid Time to E rise t_{AV} = PW_{EL}-(t_{ASD} + 80 ns) see Note 1(b)	t_{AV}	281.5	–	94	–	85	–	ns
17	Read Data Setup Time	t_{DSR}	30	–	30	–	30	–	ns
18	Read Data Hold Time(Max = t_{MAD})	t_{DHR}	10	145.5	10	83	10	80	ns
19	Write Data Delay Time t_{DDW} = 1/8t_{cyc} + 65.5 ns see Note 1(a)	t_{DDW}	–	190.5	–	128	–	125	ns
21	Write Data Hold Time t_{DHW} = 1/8t_{cyc}-29.5 ns see Note 1(a)	t_{DHW}	95.5	–	33	–	30	–	ns
22	Muxed Address Valid Time to E Rise t_{AVM} = PW_{EL}-(t_{ASD} + 90 ns) see Note 1(b)	t_{AVM}	271.5	–	84	–	75	–	ns
24	Muxed Address Valid Time to As Fall t_{AVM} = PW_{ASH}-70 ns	t_{ASL}	151	–	26	–	20	–	ns
25	Muxed Address Hold Time t_{AHL} = 1/8 t_{cyc}-29.5 ns see Note 1(b)	t_{AHL}	95.5	–	33	–	30	–	ns
26	Delay Time,E to AS Rise t_{ASD} = 1/8 t_{cyc}-9.5 ns see Note 1(a)	t_{ASD}	115.5	–	53	–	50	–	ns
27	Pulse Width, AS High PW_{ASH} = 1/4t_{cyc}-29 ns	P_{WASH}	221	–	96	–	90	–	ns
28	Delay Time, AS to E Rise t_{ASED} = 1/8t_{cyc}-9.5 ns see Note 1(b)	t_{ASED}	115.5	–	53	–	50	–	ns
29	MPU Address Access Time t_{ACCA} = t_{AVM} + t_r + PW_{EH}-t_{DSR} see note 1(b)	t_{ACCA}	733.5	–	296	–	275	–	ns
35	MPU Access Time t_{ACCE} = PW_{EH}-t_{DSR}	t_{ACCE}	–	442	–	192	–	180	ns
36	Muxed Address Delay (Previous Cycle MPU Read) t_{MAD} = t_{ASD} + 30 ns see Note 1(a)	t_{MAD}	145.5	–	83	–	80	–	ns

Notes :

1. Input clocks with duty cycles other than 50% will affect bus performance. Timing parameters affected by input clock duty cycle are identified by (a) and (b). To recalculate the approximate bus timing values, substitute the follwing expressions in place of 1/8 t_{cyc} in the formulas where applicable:
   (a) (1-DC)×1/4tcyc
   (b) DC×1/4tcyc
   Where :
   DC is the decimal value of duty cycle percentage (high time)
2. All timing is shown with respect to 20% V_{DD} and 70% V_{DD} unless othewise noted.



Note : Measurement point shown are 20% and 70% V_{DD}.

Figure 11.14 Expansion Bus Timing Diagram

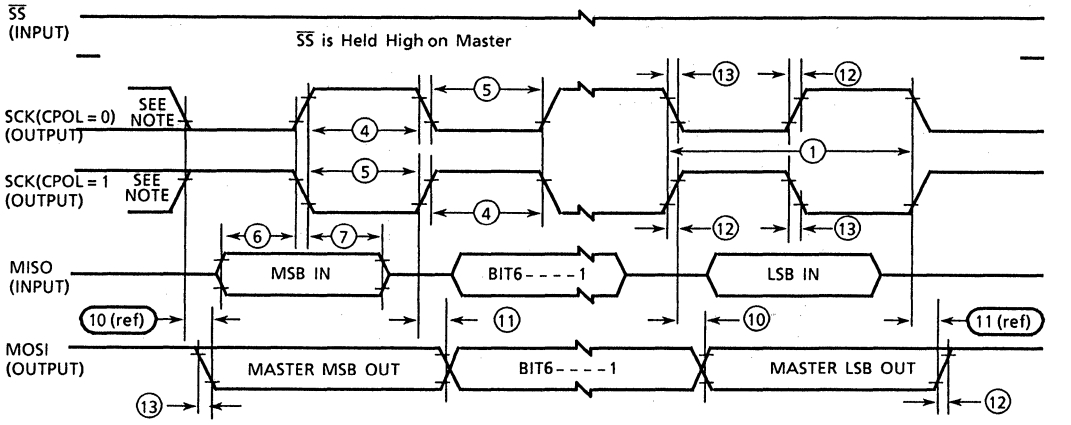
11.9 SERIAL PERIPHERAL INTERFACE (SPI) TIMING(V_{DD}=5.0 Vdc±10%, V_{SS}=0 Vdc, T_A=T_L to T_H, see Figure 11.15)

Num.	Characteristic	Symbol	Min	Max	Unit
	Operating Frequency Master Slave	 f_{OP} (m) f_{OP} (S)	 dc dc	 0.5 2.1	 fop MHz
1	Cycle Time Master Slave	 f_{cyc} (m) f_{cyc} (S)	 2.0 480	 – –	 t_{cyc} ns
2	Enable Lead Time Master Slave	 t_{lead} (m) t_{lead} (S)	 * 240	 – –	 ns ns
3	Enable Lag Time Master Slave	 t_{lag} (m) t_{lag} (S)	 * 240	 – –	 ns ns
4	Clock (SCK) High Time Master Slave	 t_w (SCKH) $_m$ t_w (SCKH) $_s$	 340 190	 – –	 ns ns
5	Clock (SCK) Low Time Master Slave	 t_w (SCKL) $_m$ t_w (SCKL) $_s$	 340 190	 – –	 ns ns
6	Data Setup Time (Inputs) Master Slave	 t_{su} (m) t_{su} (s)	 100 100	 – –	 ns ns
7	Data Hold Time (Inputs) Master Slave	 t_h (m) t_h (s)	 100 100	 – –	 ns ns
8	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 (After Enable Edge)**	t_v (S)	–	240	ns
11	Data Hold Time (Outputs) (After Enable Edge)	t_{ho}	0	–	ns
12	Rise Time (20% V_{DD} to 70% V_{DD}, C_L = 200pF) SPI Outputs(SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and $\overline{SS}$)	 t_{rm} t_{rs}	 – –	 100 2.0	 ns µs
13	Fall Time (70% V_{DD} to 20% V_{DD}, C_L = 200pF) SPI outputs(SCK, MOSI, and MISO) SPI Inputs(SCK, MOSI, MISO, and $\overline{SS}$)	 t_{rm} t_{fs}	 – –	 100 2.0	 ns µs

*Signal production depends on software.

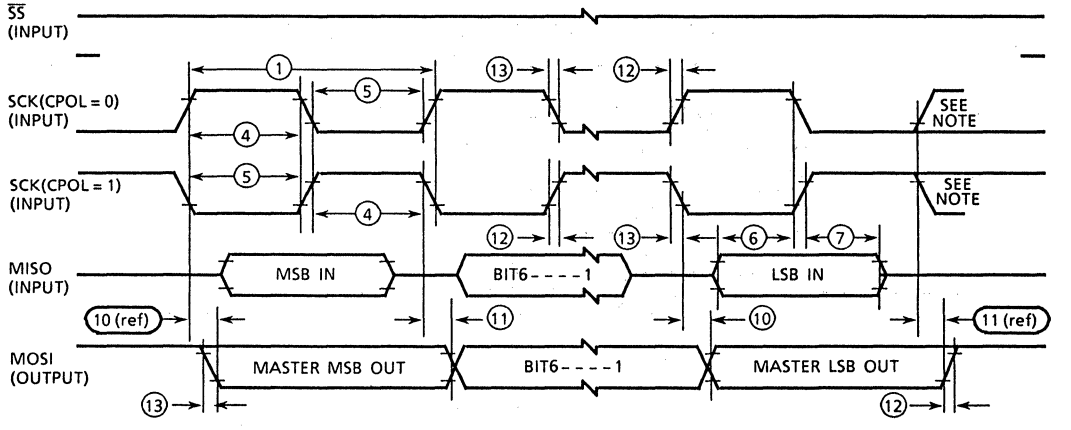
**Assumes 200 pF load on all SPI pins.

Note : All timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted.



Note : This first clock edge is generated intemally but is seen at the SCK pin.

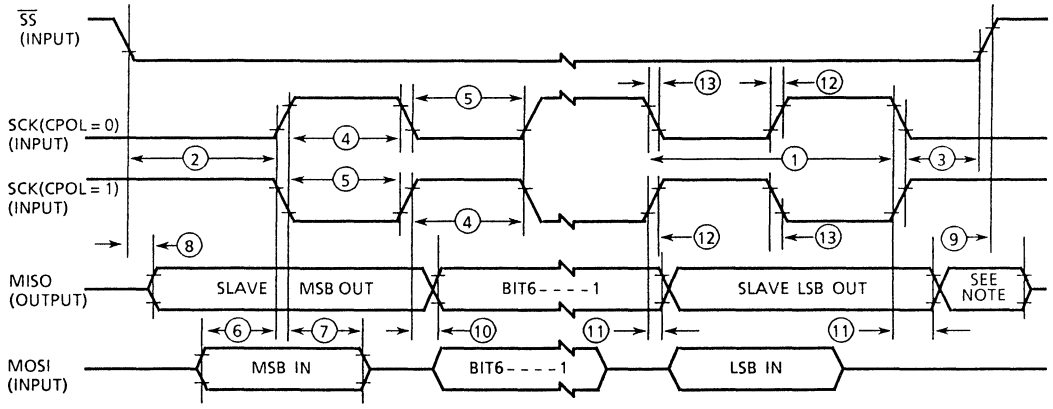
a) SPI MASTER TIMING (CPHA = 0)



Note : This iast clock edge is generated intemally but is seen at the SCK pin.

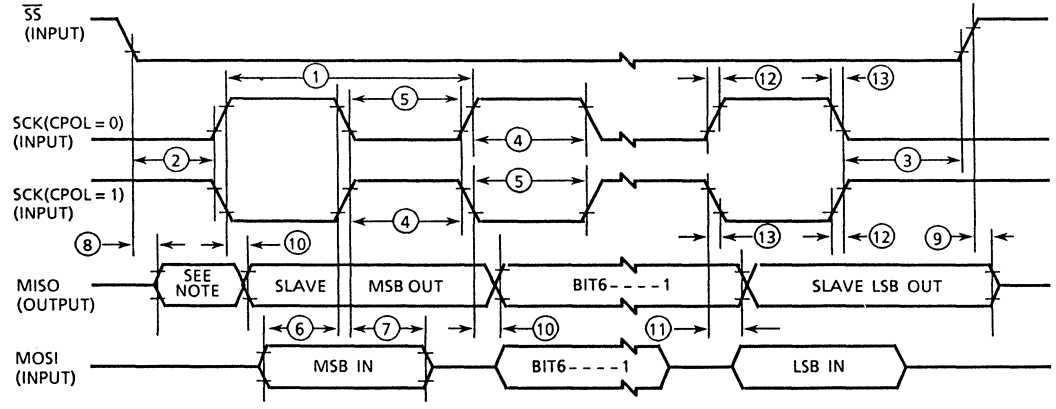
b) SPI MASTER TIMING (CPHA = 1)

Figure 11.15 SPI Timing Diagrams (Sheet 1 of 2)



Note : Not defined but normally MSB of character just received.

c) SPI SLAVE TIMING (CPHA = 0)



Note : Not defined but normally LSB of character previously transmitted.

d) SPI SLAVE TIMING (CPHA = 1)

Figure 11.15 SPI Timing Diagrams (Sheet 2 of 2)

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

Characteristic	Temperature Range			Unit
	– 40 to 85°C	– 40 to 105°C	– 40 to 125°C	
Programming time (See Note 1) Under 1.0 MHz with RC Oscillator Enable 1.0 to 2.0 MHz with RC Oscillator Disabled 2.0 MHz (or Anytime RC Oscillator Enabled)	 10 20 10	 15 Must Use RC 15	 20 Must Use RC 20	ms
Erase Time (see Note 1) Byte, Row, and Bulk	10	10	10	ms
Write Erase Endurance (See Note 2)	10, 000	10, 000	10, 000	Cycles
Data Retention (See Note 2)	10	10	10	Years

Notes:

1. The RC oscillator must be enabled (by setting the CSEL bit in the OPTION register) for EEPROM programming and erasure when the E-clock frequency is below 1.0MHz.
2. See current quarterly Reliability Monitor Report for current failure rate information.

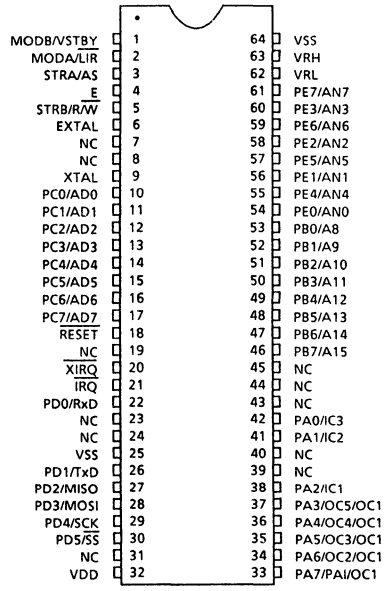
12. MECHANICAL DATA

This section contains the pin assignment and package dimension diagrams for the TMP68HC11A8 as well as information to be used as a guide when ordering the MCU.

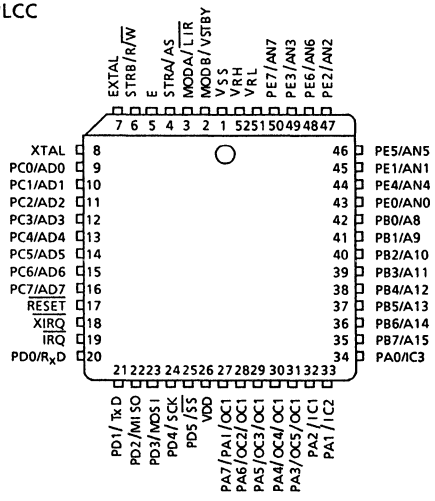
12.1 PIN ASSIGNMENTS

The TMP68HC11A8 is available in both a 48-pin plastic dual-in-line package and a 52-pin plastic lead chip carrier (PLCC) package. The following paragraphs provide pin assignments for both package versions.

N SUFFIX
64 PIN S-DIP



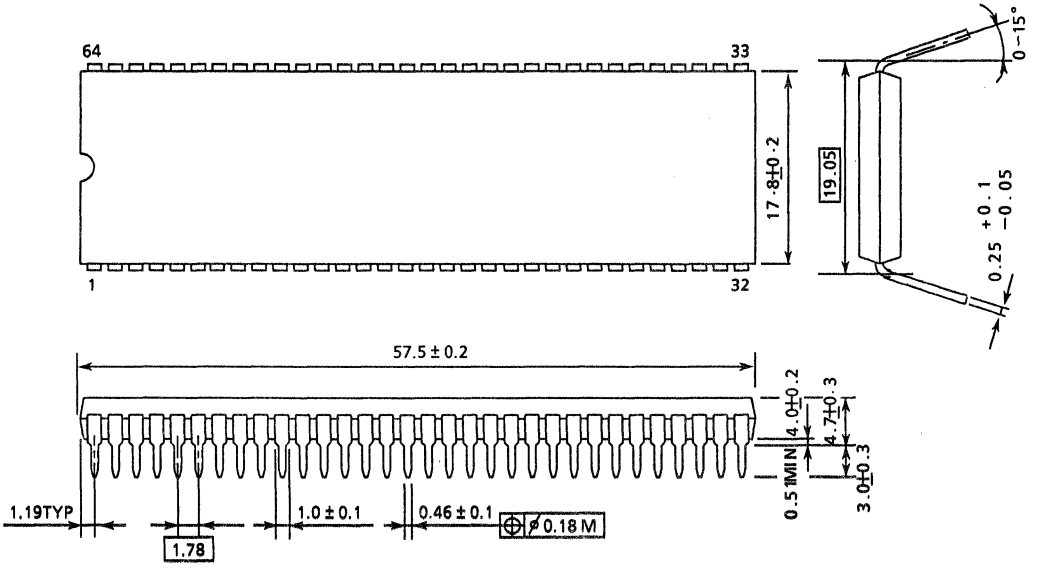
T SUFFIX
52 PIN PLCC



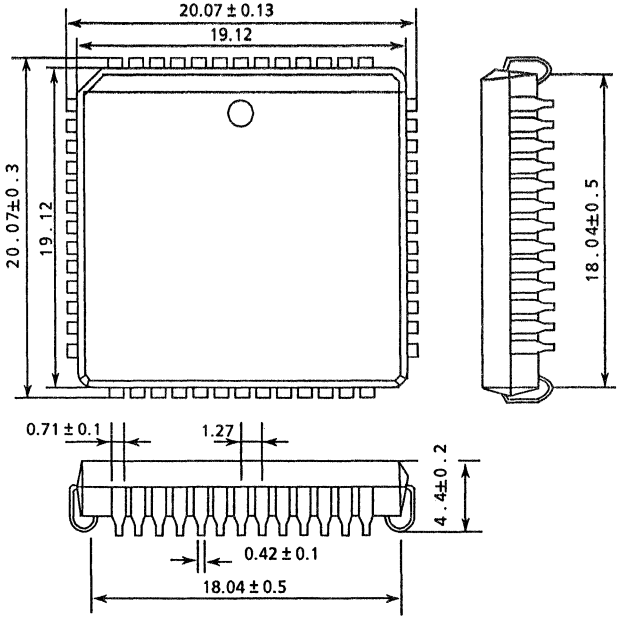
12.2 PACKAGE DIMENSIONS

N SUFFIX 64PIN S-DIP

(UNIT : mm)



T SUFFIX 52PIN PLCC



SECTION 3
TMP68HC11E9

1. This technical data may be controlled under U.S. Export Administration Regulations and may be subject to the approval of the U.S. Department of Commerce prior to export. Any export or re-export directly or indirectly, in contravention of the U.S. Export Administration Regulations is strictly prohibited.

2. LIFE SUPPORT POLICY

   Toshiba products described in this databook are not authorized for use as critical components in life support systems without the written consent of the appropriate officer of Toshiba America, Inc. Life support systems are either systems intended for surgical implant in the body or systems which sustain life.

   A critical component is any component of a life support system whose failure to perform may cause a malfunction or failure of the life support system, or may affect its safety or effectiveness.

3. The information in this databook has been carefully checked and is believed to be reliable, however, no responsibility can be assumed for inaccuracies that may not have been caught. All information in this databook is subject to change without prior notice. Furthermore, Toshiba cannot assume responsibility for the use of any license under the patent rights of Toshiba or any third parties.

4. No part of this manual may be transferred or reproduced without prior permission of Toshiba Corporation.

CONTENTS

1 INTRODUCTION

The TMP68HC11E9 is an advanced 8-bit microcontroller (MCU) with highly sophisticated on-chip peripheral capabilities. New design techniques were used to achieve a nominal bus speed of 2MHz. In addition, the fully static design allows operation at frequencies down to dc, further reducing power consumption.

1.1 FEATURES

The following are some of the hardware and software highlights.

HARDWARE FEATURES

- 12K Bytes of ROM
- 512 Bytes of EEPROM (with Block Protect for Enhanced Security)
- 512 Bytes of RAM (All Saved During Standby) Relocatable to Any 4K Boundary
- Enhanced 16-Bit Timer System:

  Four Stage Programmable Prescaler

  Three Input Capture/Five Output Compare Functions or

  Four Input Capture/Four Output Compare Functions (S/W selectable)
- 8-Bit Pulse Accumulator Circuit
- Enhanced NRZ Serial Communications Interface (SCI)
- Serial Peripheral Interface (SPI)
- Eight Channel, 8-Bit Analog-to-Digital Converter
- Real Time Interrupt Circuit
- Computer Operating Properly (COP) Watchdog System
- Available in Plastic Shrink Dual-In-Line Packages and Plastic Leaded Chip Carrier Packages

SOFTWARE FEATURES

- Enhanced M6800/M6801 Instruction Set
- 16×16 Integer and Fractional Divide Features
- Bit Manipulation
- WAIT Mode
- STOP Mode

1.2 GENERAL DESCRIPTION

The high-density CMOS technology used on the TMP68HC11E9 combines smaller size and higher speeds with the low power and high noise immunity of CMOS. On-chip memory systems include 12K bytes of ROM, 512 bytes of electrically eraseable programmable ROM (EEPROM), and 512 bytes of static RAM.

Major peripheral functions are provided on-chip. An eight channel analog-to-digital (A/D) converter is included with eight bits of resolution. An asynchronous serial communications interface (SCI) and a separate synchronous serial peripheral interface (SPI) are included. The main 16-bit free-running timer system has three input capture lines, five output compare lines, and a real-time interrupt function. An 8-bit pulse accumulator subsystem can count external events or measure external periods.

Self monitoring circuitry is included on-chip to protect against system errors. A computer operating properly (COP) watchdog system protects against software failures. A clock monitor system generates a system reset in case the clock is lost or runs too slow. An illegal opcode detection circuit provides a non-maskable interrupt if an illegal opcode is detected.

Two software controlled operating modes, WAIT and STOP, are available to conserve additional power.

A block diagram of the TMP68HC11E9 is shown in Figure 1.1.

1.3 PROGRAMMER'S MODEL

In addition to being able to execute all M6800 and M6801 instructions, the TMP68HC11E9 allows execution of 91 new opcodes. Figure 1.2 shows the seven CPU registers which are available to the programmer.

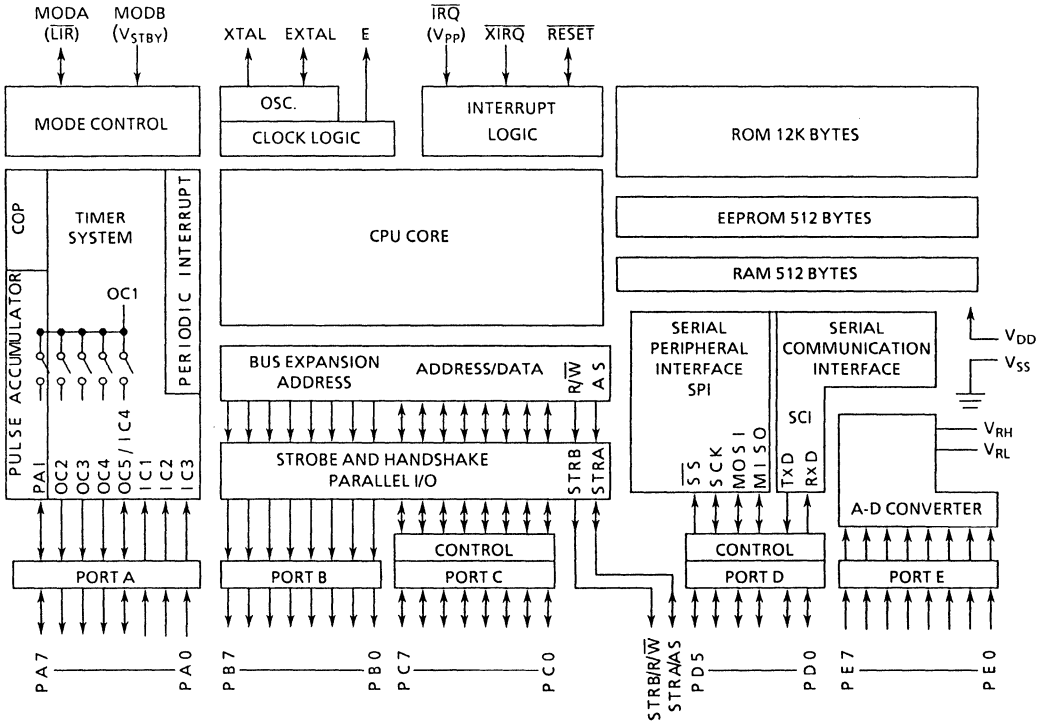


Figure 1.1 Block Diagram

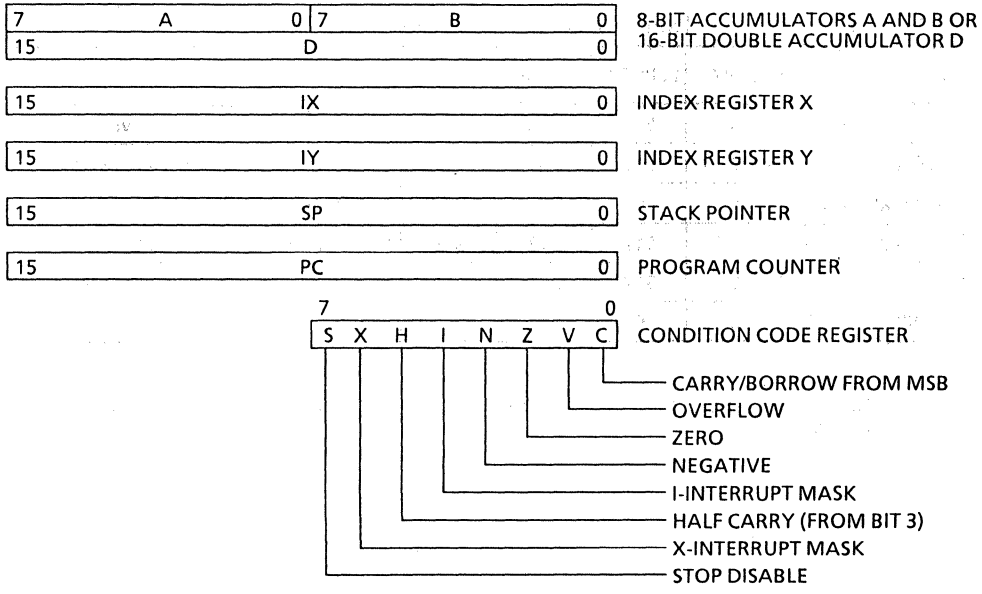


Figure 1.2 Programming Model

1.4 SUMMARY OF TMP68HC11 FAMILY

Table 1.1 and the following paragraphs summarize the current members of the TMP68HC11 Family. This data sheet describes the TMP68HC11E9, E1, and E0 version and is to be used as a secondary reference for other versions. Family members differ mainly in the types and amounts of memory. The A series parts ('A8, 'A1, and 'A0) are the foundation of the TMP68HC11 Family.

Notice that each major derivative has an x8 or x9, x1, and x0 variation. These variations all use identical die. A configuration (CONFIG) register is implemented with EEPROM cells and is used to semi-permanently disable the ROM of x1 variations. The ROM and EEPROM are disabled on x0 variations.

The E series was developed for applications requiring four input capture functions for the timer, more ROM, or more RAM. These parts are modified to allow the former output compare five function to be configured as either an output compare or as a fourth input capture function. The amount of RAM was also increased to 512 bytes and the amount of ROM was increased to 12K bytes.

All E series parts are available in 64-pin plastic shrink-dual-in-line (S-DIP) packages and52-pin plastic lead chip currier (PLCC) packages.

Table 1.1 TMP68HC11 Family Members

Device Number	ROM	EEPROM	RAM	CONFIG[1]	Comments
TMP68HC11A8	8K	512	256	$0F	Family Built Around this Device
TMP68HC11A1	0	512	256	$0D	Same Die as ' A8 but ROM Disabled
TMP68HC11A0	0	0	256	$0C	Same Die as ' A8 but ROM and EEPROM Disabled
TMP68HC11E9	12K	512	512	$0F	Four Input Captures and Bigger RAM and 12K ROM
TMP68HC11E1	0	512	512	$0D	'E9 with ROM Disabled
TMP68HC11E0	0	0	512	$0C	'E9 with ROM and EEPROM Disabled

Notes :

1. CONFIG register values in this table reflect the value programmed prior to shipment from TOSHIBA.

2. SIGNAL DESCRIPTIONS AND OPERATING MODES

The signal descriptions and operating modes are presented in this section. When the microcontroller is in an expanded multiplexed operating mode, 18 pins change function to support a multiplexed address/data bus.

2.1 SIGNAL PIN DESCRIPTIONS

The following paragraphs provide a description of the input/output signals. Reference is made, where applicable, to other sections that contain more detail about the function being performed.

2.1.1 Input Power (V_{DD}) and Ground (V_{SS})

Power is supplied to the microcontroller using these pins. V_{DD} is the positive power input and V_{SS} is ground. Although the TMP68HC11E9 is a CMOS device, very fast signal transitions are present on many of its pins. Short rise and fall times are present even when the microcontroller is operating at slow clock rates. Special care must be taken to provide good power supply bypassing at the MCU. Recommended bypassing would include a 0.1 µF ceramic capacitor between the V_{DD} and V_{SS} pins and physically adjacent to one of the two pins. A bulk capacitance, whose size depends on the other circuity in the system, should also be present on the circuit board.

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

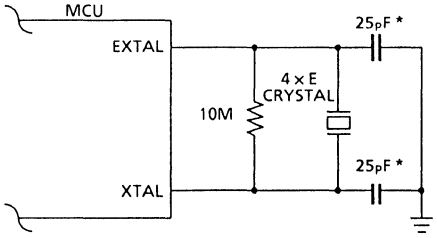
This active low bidirectional control signal is used as an input to initialize the TMP68HC11E9 to a known startup state, and as an open-drain output to indicate that an internal failure has been detected in either the clock monitor or computer operating properly (COP) watchdog circuit. This reset signal is significantly different from the reset signal used on other Motorola MCUs. Please refer to SECTION 9 RESETS, INTERRUPTS, AND LOW POWER MODES before designing circuitry to generate or monitor this signal.

2.1.3 Crystal Driver and External Clock Input (XTAL, EXTAL)

These two pins provide the interface for either a crystal or a CMOS compatible clock to control the internal clock generator circuitry. The frequency applied to these pins shall be four times higher than the desired E clock rate. The XTAL pin is normally left unterminated when using an external CMOS compatible clock input to the EXTAL pin. However, a 10K to 100K load resistor to ground may be used to reduce RFI noise emission. The XTAL output is normally intended to drive only a crystal.

The XTAL output may be buffered with a high input impedance buffer such as the 74HC04, or it may be used to drive the EXTAL input of another TMP68HC11.

In all cases take extra care in the circuit board layout around the oscillator pins. Load capacitances shown in the oscillator circuits include all stray layout capacitances. Refer to Figures 2.1, 2.2, and 2.3 for diagrams of oscillator circuits.



* This value includes all stray capacitances.

Figure 2.1 Common Crystal Connections

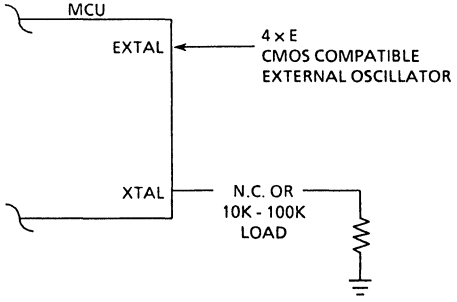
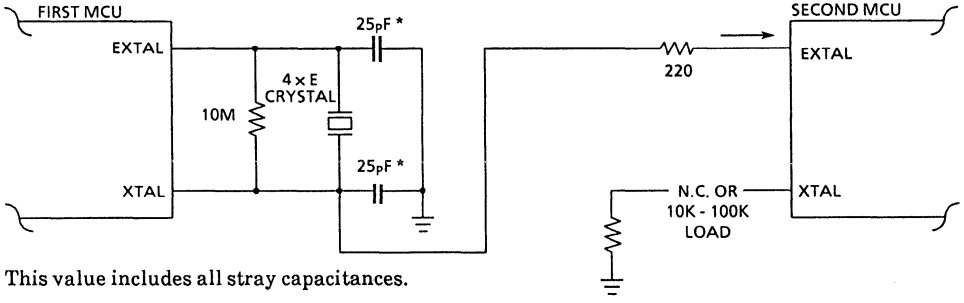


Figure 2.2 External Oscillator Connections



* This value includes all stray capacitances.

Figure 2.3 One Crystal Driving 2 MCUs

2.1.4 E Clock Output (E)

This is the output connection for the internally generated E clock which can be used as a timing reference. The frequency of the E clock output is actually one fourth that of the input frequency at the XTAL and EXTAL pins. When the E clock output is low an internal process is taking place and, when high, data is being accessed. The E clock signal is halted when the MCU is in STOP mode.

2.1.5 Interrupt Request ($\overline{IRQ}$)

The $\overline{IRQ}$ input provides a means for requesting asynchronous interrupts to the TMP68HC11E9. It is program selectable (OPTION register) with a choice of either negative edge-sensitive or level-sensitive triggering, and is always configured to level-sensitive triggering by reset. The $\overline{IRQ}$ pin requires an external pullup resistor to V_{DD} (typically 4.7K Ω).

During factory testing, this pin is also used as a bulk V_{PP} power supply-input. This allows for parallel programming of as many as half of the bytes in the EEPROM in a single programming operation.

2.1.6 Non-Maskable Interrupt ($\overline{XIRQ}$)

This input provides a means for requesting a non-maskable interrupt, after reset initialization. During reset, the X bit in the condition code register is set and any interrupt is masked until MCU software enables it. The $\overline{XIRQ}$ input is level sensitive and requires an external pullup resistor to V_{DD}.

2.1.7 Mode A/Load Instruction Register and Mode B/Standby Voltage (MODA/$\overline{LIR}$, MODB/V_{STBY})

During reset, MODA and MODB are used to select one of the four operating modes. Refer to Table 2.1 Paragraph 2.2 OPERATING MODES provides additional information.

Table 2.1 Operating Modes Versus MODA and MODB

MODB	MODA	Mode Selected
1	0	Single Chip
1	1	Expanded Multiplexed
0	0	Special Bootstrap
0	1	Special Test

After the operating mode has been selected, the $\overline{LIR}$ pin provides an open-drain output to indicate that an instruction is starting. All instructions are make up of a series of E clock cycles. The $\overline{LIR}$ signal goes low during the first E clock cycle of each instruction (opcode fetch). This output is provided as an aid in program debugging.

The V_{STBY} signal is used as the input for RAM standby power. When the voltage on this pin is more than one MOS threshold (about 0.7 volts) above the V_{DD} voltage, the internal 512-byte RAM and part of the reset logic are powered from this signal rather than the V_{DD} input. This allows RAM contents to be retained without V_{DD} power applied to the MCU. Reset must be driven low before V_{DD} is removed and must remain low until V_{DD} has been restored to a valid level.

2.1.8 A/D Converter Reference Voltages (V_{RL}, V_{RH})

These two inputs provide the reference voltages for the analog-to-digital converter circuitry.

2.1.9 Strobe B and Read/Write (STRB/R/$\overline{W}$)

This signal acts as a strobe B output or as a data bus direction indicator depending on the operating mode.

In single-chip operating mode, the STRB output acts as a programmable strobe for handshake with other parallel I/O devices. Refer to SECTION 4 PARALLEL I/O for additional information.

In expanded multiplexed operating mode, R/$\overline{W}$ is used to control the direction of transfers on the external data bus. A low on the R/$\overline{W}$ signal indicates data is being written to the external data bus. A high on this signal indicates that a read cycle is in progress. R/$\overline{W}$ will stay low during consecutive data bus write cycles, such as in a double-byte store. The NAND of inverted R/$\overline{W}$ with the E clock should be used as the write enable signal for an external static RAM.

2.1.10 Strobe A and Address Strobe (STRA/AS)

This signal acts as an edge detecting strobe A input or as an address strobe bus control output depending on the operating mode.

In single-chip operating mode, the STRA input acts as a programmable strobe for handshake with other parallel I/O devices. Refer to SECTION 4 PARALLEL I/O for additional information.

In expanded multiplexed operating mode, the AS output is used to demultiplex the address and data signals at port C. Refer to 2.2.2 Expanded Multiplexed Operating Mode for additional information.

2.1.11 Port Signals

Ports A, D, and E signals are independent of the operating mode. Port B provides eight general purpose output signals in single-chip operating modes and provides eight high-order address signals when the microcontroller is in expanded multiplexed operating modes. Port C provides eight general purpose input/output signals when the microcontroller is in single-chip operating modes. When the microcontroller is in

expanded multiplexed operating modes, port C is used for a multiplexed address/data bus. Table 2.2 shows an summary of the 40 port signals as they relate to the operating modes. Unused inputs and I/O pins configured as inputs should be terminated high or low.

2.1.11.1 Port A.

Port A may be configured for: four input capture functions (IC1, IC2, IC3, IC4), and three output compare functions (OC2, OC3, OC4,) , and either a pulse accumulator input (PAI) or a fifth output compare function (OC1) . Refer to 8.1 PROGRAMMABLE TIMER for additional information.

Any port A pin that is not used for its alternate timer function may be used as a general-purpose input or output line.

Table 2.2 Port Signal Summary

Port-Bit	Single-Chip and Bootstrap Mode	Expanded Multiplexed and Special Test Mode
A-0	PA0/IC3	PA0/IC3
A-1	PA1/IC2	PA1/IC2
A-2	PA2/IC1	PA2/IC1
A-3	PA3/OC5/IC4/and-or OC1	PA3/OC5/IC4/and-or OC1
A-4	PA4/OC4/and-or OC1	PA4/OC4/and-or OC1
A-5	PA5/OC3/and-or OC1	PA5/OC3/and-or OC1
A-6	PA6/OC2/and-or OC1	PA6/OC2/and-or OC1
A-7	PA7/PAI/and-or OC1	PA7/PAI/and-or OC1
B-0	PB0	A8
B-1	PB1	A9
B-2	PB2	A10
B-3	PB3	A11
B-4	PB4	A12
B-5	PB5	A13
B-6	PB6	A14
B-7	PB7	A15
C-0	PC0	A0/D0
C-1	PC1	A1/D1
C-2	PC2	A2/D2
C-3	PC3	A3/D3
C-4	PC4	A4/D4
C-5	PC5	A5/D5
C-6	PC6	A6/D6
C-7	PC7	A7/D7
D-0	PD0/RxD	PD0/R_XD
D-1	PD1/TxD	PD1/T_XD
D-2	PD2/MISO	PD2/MISO
D-3	PD3/MOSI	PD3/MOSI
D-4	PD4/SCK	PD4/SCK
D-5	PD5/$\overline{SS}$	PD5/$\overline{SS}$
	STRA	AS
	STRB	R/$\overline{W}$
E-0	PE0/AN0	PE0/AN0
E-1	PE1/AN1	PE1/AN1
E-2	PE2/AN2	PE2/AN2
E-3	PE3/AN3	PE3/AN3
E-4	PE4/AN4	PE4/AN4
E-5	PE5/AN5	PE5/AN5
E-6	PE6/AN6	PE6/AN6
E-7	PE7/AN7	PE7/AN7

2.1.11.2 Port B.

While in single-chip operating modes, all of the port B pins are general-purpose output pins. During MCU reads of this port, the level sensed at the input side of the port

B output drivers is read. Port B may also be use in a simple strobed output mode where an output pulse appears at the STRB signal each time data is written to port B.

When in expanded multiplexed operating modes, all of the port B pins act as high order address output signals. During each MCU cycle, bits 8 through 15 of the address are output on the PB0-PB7 lines respectively.

2.1.11.3 Port C.

While in single-chip operating modes, all port C pins are general-purpose input/output pins. Port C inputs can be latched by providing an input transition to the STRA signal. Port C may also be used in full handshake modes of parallel I/O where the STRA input and STRB output act as handshake control lines.

When in expanded multiplexed operating modes, all port C pins are configured as multiplexed address/data signals. During the address portion of each MCU cycle, bits 0 through 7 of the address are output on the PC0-PC7 lines. During the data portion of each MCU cycle (E high), pins 0 through 7 are bidirectional data signals (D0-D7). The direction of data at the port C pins is indicated by the R/$\overline{W}$ signal.

2.1.11.4 Port D.

Port D pins 0-5 may be used for general purpose I/O signals. Port D pins alternately serve as the serial communications interface (SCI) and serial peripheral interface (SPI) signals when those subsystems are enabled.

Pin PD0 is the receive data input (RxD) signal for the serial communication interface (SCI).

Pin PD1 is the transmit data output (TxD) signal for the SCI.

Pins PD2 through PD5 are dedicated to the SPI. PD2 is the master-in-slave-out (MISO) signal. PD3 is the master-out-slave in (MOSI) signal. PD4 is the serial clock (SCK) signal and PD5 is the slave select ($\overline{SS}$) input.

2.1.11.5 Port E.

Port E is used for general-purpose inputs and/or analog-to-digital (A/D) input channels. Reading port E during the sampling portion of an A/D conversion could cause very small disturbances and affect the accuracy of that result. If very high accuracy is required, avoid reading port E during conversions.

2.2 OPERATING MODES

There are four operation modes for the TMP68HC11E9:single-chip operating mode, expanded multiplexed operating mode, special bootstrap operating mode, and special test operating mode. Table 2.1 shows how the operating mode is selected. The following paragraphs describe these operating modes.

2.2.1 Single-Chip Operating Mode

In single-chip operating mode, the TMP68HC11E9 functions as a monolithic microcontroller without external address or data buses. Port B, port C, strobe A, and strobe B function as general purpose I/O and handshake signals. Refer to SECTION 4 PARALLEL I/O for additional information.

2.2.2 Expanded Multiplexed Operating Mode

In expanded multiplexed operating mode, the TMP68HC11E9 has the capability of accessing a 64K byte address space. This total address space includes the same on-chip memory addresses used for single-chip operating mode plus external peripheral and memory devices. The expansion bus is made up of port B and port C, and control signals AS and R/$\overline{W}$. Figure 2.4 shows a recommended way of demultiplexing low order addresses from data at port C. The address, R/$\overline{W}$, and AS signals are active and valid for all bus cycles including accesses to internal memory locations.

2.2.3 Special Bootstrap Operating Mode

The bootstrap mode is considered a special operating mode as distinguished from the normal single-chip operating mode. This is a very versatile operating mode since there are essentially no limitations on the special purpose program that can be loaded into the internal RAM. The boot loader program is contained in the 192 byte bootstrap ROM. This ROM is enabled only if the MCU is reset in special bootstrap operating mode, and appears as internal memory space at locations $BF40-$BFFF. The boot loader program will use the SCI to read a variable length program, up to 512 bytes, into on-chip RAM at locations $0000-$01FF. After the finale byte is received, control is automatically passed to that program at location $0000.

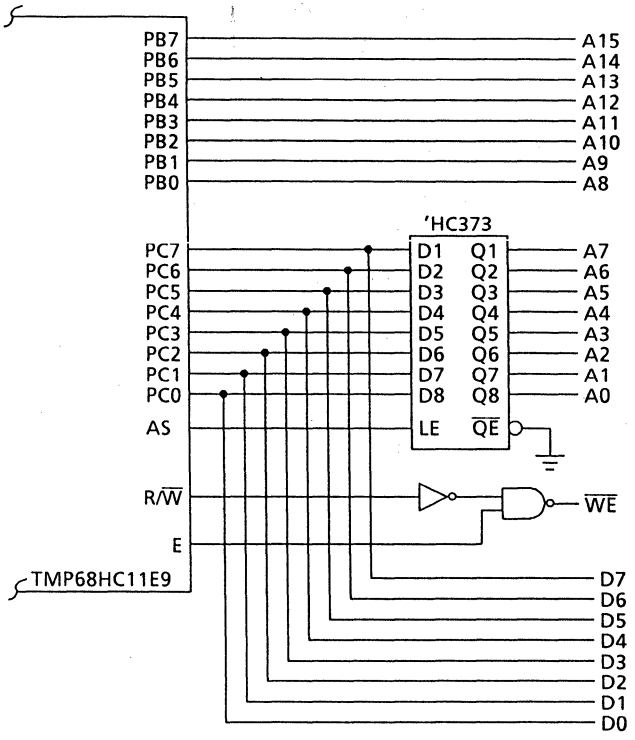


Figure 2.4 Address/Data Demultiplexing

The TMP68HC11E9 communicates through the SCI port. After reset in special bootstrap operating mode, the SCI is running at E clock/16 (7812 baud for E clock equal 2 MHz). If the security feature was specified and the security bit is set, $FF is output by the SCI transmitter. The EEPROM is then erased. If erasure is unsuccessful, $FF is output again and erasure is attempted again. Upon successful erasure of the EEPROM, all internal RAM is written over with $FF. The CONFIG register is then erased. The boot loader program now proceeds as though the part had not been in security mode.

Note : If the security mode is not included by specific request (mask option), the code which checks for security and erases EEPROM is not included in the boot loader ROM.

If the part is not in security mode (or has completed the above erase sequence), a break character is output by the SCI transmitter. For normal use of the boot loader program, the user sends $FF to the SCI receiver at either E clock/16 (7218 baud for E clock = 2 MHz) or E clock/104 (1200 baud for E clock = 2 MHz).

Note : This $FF is not echoed through the SCI transmitter.

Next the user must downlord up to 512 bytes of program data to be put into RAM starting at location $0000. These characters are echoed through the transmitter. When loading is complete, the program jumps to location $0000 and begins executing that code. The boot loader program ends the down load after 512 bytes or when the receive data line is idle for at least four character times.

If the SCI transmitter pin is to be used, an external pullup resistor is required because port D pins are configured for wire-OR operation.

In special bootstrap operating mode the interrupt vectors are directed to RAM as shown in Table 2.3 This allows the user to use interrupts by way of a jump table. For example:to use the SWI interrupt, a jump instruction would be placed in RAM at locations $00F4, $00F5, and $00F6. When an SWI is encountered, the vector (which is in the boot loader ROM program) will direct program control to location $00F4 in RAM which in turn contains a JUMP instruction to the interrupt service routine. The program to administer the security option is longer than the basic boot loader, so parts with the security option may not have a complete pseudo vector table.

Table 2.3 Bootstrap Mode Interrupt Vectors

Address	Vector
00C4	SCI
00C7	SPI
00CA	Pulse Accumulator Input Edge
00CD	Pulse Accumulator Overflow
00D0	Timer Overflow
00D3	Timer Output Compare 5
00D6	Timer Output Compare 4
00D9	Timer Output Compare 3
00DC	Timer Output Compare 2
00DF	Timer Output Compare 1
00E2	Timer Input Capture 3
00E5	Timer Input Capture 2
00E8	Timer Input Capture 1
00EB	Real Time Interrupt
00EE	IRQ
00F1	XIRQ
00F4	SWI
00F7	Illegal Opcode
00FA	COP Fail
00FD	Clock Monitor
BF40 (Boot)	Reset

2.2.4 Additional Boot Loader Program Options

The user may transmit a $55 (only at E clock/16) as the first character rather than the normal $FF. This will cause the program to jump directly to location $0000, skipping the download.

The user may tie receiver to the transmitter (with an external pull-up resistor). This will cause the program to jump directly to the beginning of EEPROM ($B600). Another way to cause the program to jump directly to EEPROM is to transmit either a break or $00 as the first character rather than the normal $FF.

Note that none of these options bypass the security check and so do not compromise those customers using security.

Keep in mind that upon entry to the downloaded program at location $0000, some registers have been changed from their reset states. The SCI transmitter and receiver are enabled which cause port D pins 0 and 1 to be dedicated to SCI use. Also port D is configured for wired-OR operation. It may be necessary for the user to write to the SCCR2 and SPCR registers to disable the SCI and/or port D wire-OR operation.

2.2.5 Special Test Operating Mode

The test mode is a special operating mode intended primarily for factory testing. This mode is very similar to the expanded multiplexed operating mode. In special test operating mode, the reset and interrupt vectors are fetched from external memory locations $BFC0-$BFFF rather than $FFC0-$FFFF. There are no time limits for protection of the TMSK2, OPTION, and INIT registers, so these registers may be written repeatedly. Also a special TEST1 register is enabled which allows several factory test functions to be invoked.

The special test operating mode is not recommended for use by an end user because of the reduced system security; however, an end user may wish to come out of reset in special test operating mode. Then, after some initialization, the SMOD and MDA bits could be rewritten to select a normal operating mode to re-enable the protection features.

3. ON-CHIP MEMORIES

This section describes the on-chip ROM, RAM and EEPROM memories. The memory maps for each mode of operation are shown and the RAM and I/O mapping register (INIT) is described. The INIT register allows the on-chip RAM and the 64 control registers to be moved to suit the needs of a particular application.

3.1 MEMORY MAPS

Composite memory maps for each mode of operation are shown in Figure 3.1 Memory locations are shown in the shaded areas and the contents of these shaded areas are shown to the right. These modes include single-chip, expanded multiplexed, special bootstrap, and special test.

Single-chip operating modes do not generate external addresses. Refer to Table 3.1 for a full list of the registers.

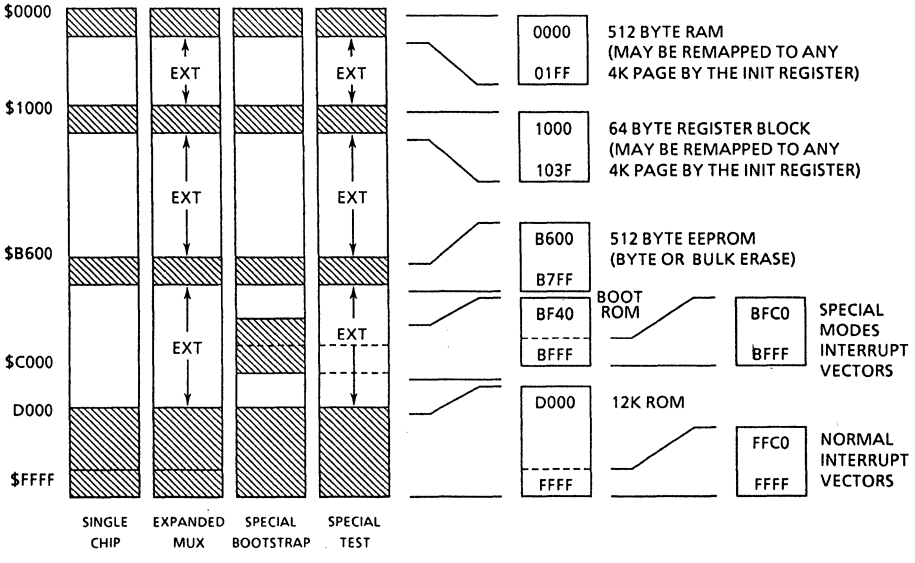


Figure 3.1 Memory Maps

Table 3.1 Register and Control Bit Assignments (Sheet 1 of 2)

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0		
$1000	Bit7	–	–	–	–	–	–	Bit0	PORTA	I/O Port A
$1001									Reserved	
$1002	STAF	STAI	CWOM	HNDS	OIN	PLS	EGA	INVB	PIOC	Parallel I/O Control Register
$1003	Bit7	–	–	–	–	–	–	Bit0	PORTC	I/O Port C
$1004	Bit7	–	–	–	–	–	–	Bit0	PORTB	Output Port B
$1005	Bit7	–	–	–	–	–	–	Bit0	PORTCL	Alternate Latched Port C
$1006									Reserved	
$1007	Bit7	–	–	–	–	–	–	Bit0	DDRC	Data Direction for Port C
$1008			Bit5	–	–	–	–	Bit0	PORTD	I/O Port D
$1009			Bit5	–	–	–	–	Bit0	DDRD	Data Direction for Port D
$100A	Bit7	–	–	–	–	–	–	Bit0	PORTE	Input Port E
$100B	FOC1	FOC2	FOC3	FOC4	FOC5				CFORC	Compare Force Register
$100C	OC1M7	OC1M6	OC1M5	OC1M4	OC1M3				OC1M	OC1 Action Mask Register
$100D	OC1D7	OC1D6	OC1D5	OC1D4	OC1D3				OC1D	OC1 Action Data Register
$100E	Bit15	–	–	–	–	–	–	Bit8	TCNT	Timer Counter Register
$100F	Bit7	–	–	–	–	–	–	Bit0		
$1010	Bit15	–	–	–	–	–	–	Bit8	TIC1	Input Capture 1 Register
$1011	Bit7	–	–	–	–	–	–	Bit0		
$1012	Bit15	–	–	–	–	–	–	Bit8	TIC2	Input Capture 2 Register
$1013	Bit7	–	–	–	–	–	–	Bit0		
$1014	Bit15	–	–	–	–	–	–	Bit8	TIC3	Input Capture 3 Register
$1015	Bit7	–	–	–	–	–	–	Bit0		
$1016	Bit15	–	–	–	–	–	–	Bit8	TOC1	Output Compare 1 Register
$1017	Bit7	–	–	–	–	–	–	Bit0		
$1018	Bit15	–	–	–	–	–	–	Bit8	TOC2	Output Compare 2 Register
$1019	Bit7	–	–	–	–	–	–	Bit0		
$101A	Bit15	–	–	–	–	–	–	Bit8	TOC3	Output Compare 3 Register
$101B	Bit7	–	–	–	–	–	–	Bit0		
$101C	Bit15	–	–	–	–	–	–	Bit8	TOC4	Output Compare 4 Register
$101D	Bit7	–	–	–	–	–	–	Bit0		
$101E	Bit15	–	–	–	–	–	–	Bit8	TI4O5	Output Compare 5 Register/
$101F	Bit7	–	–	–	–	–	–	Bit0		Input Capture 4 Register

Table 3.1 Register and Control Bit Assignments (Sheet 2 of 2)

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0		
$1020	OM2	OL2	OM3	OL3	OM4	OL4	OM5	OL5	TCTL1	Timer Control Register 1
$1021	EDG4B	EDG4A	EDG1B	EDG1A	EDG2B	EDG2A	EDG3B	EDG3A	TCTL2	Timer Control Register 2
$1022	OC1I	OC2I	OC3I	OC4I	I4O5I	IC1I	IC2I	IC3I	TMSK1	Timer Interrupt Mask Reg.1
$1023	OC1F	OC2F	OC3F	OC4F	I4O5F	IC1F	IC2F	IC3F	TFLG1	Timer Interrupt Flag Reg.1
$1024	TOI	RTII	PAOVI	PAII			PR1	PR0	TMSK2	Timer Interrupt Mask Reg.2
$1025	TOF	RTIF	PAOVF	PAIF					TFLG2	Timer Interrupt Flag Reg. 2
$1026	DDRA7	PAEN	PAMOD	PEDGE	DDRA3	I4/O5	RTR1	RTR0	PACTL	Pulse Accum.Control Reg.
$1027	Bit7	–	–	–	–	–	–	Bit0	PACNT	Pulse Accum.Count Reg.
$1028	SPIE	SPE	DWOM	MSTR	CPOL	CPHA	SPR1	SPR0	SPCR	SPI Control Register
$1029	SPIF	WCOL		MODF					SPSR	SPI Status Register
$102A	Bit7	–	–	–	–	–	–	Bit0	SPDR	SPI Data Register
$102B	TCLR		SCP1	SCP0	RCKB	SCR2	SCR1	SCR0	BAUD	SCI Baud Rate Control
$102C	R8	T8		M	WAKE				SCCR1	SCI Control Register 1
$102D	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK	SCCR2	SCI Control Register 2
$102E	TDRE	TC	RDRF	IDLE	OR	NF	FE		SCSR	SCI Status Register
$102F	Bit7	–	–	–	–	–	–	Bit0	SCDR	SDI Data (Read RDR, Write TDR)
$1030	CCF		SCAN	MULT	CD	CC	CB	CA	ADCTL	A/D Control Register
$1031	Bit7	–	–	–	–	–	–	Bit0	ADR1	A/D Result Register 1
$1032	Bit7	–	–	–	–	–	–	Bit0	ADR2	A/D Result Register 2
$1033	Bit7	–	–	–	–	–	–	Bit0	ADR3	A/D Result Register 3
$1034	Bit7	–	–	–	–	–	–	Bit0	ADR4	A/D Result Register 4
$1035				PTCON	BPRT3	BPRT2	BPRT1	BPRT0	BPROT	EEPROM Block Protect Reg.
$1036 Thru $1038									Reserved	
$1039	ADPU	CSEL	IRQE	DLY	CME		CR1	CR0	OPTION	System Configuration Options
$103A	Bit7	–	–	–	–	–	–	Bit0	COPRST	Arm/Reset COP Timer Circuitry.

Table 3.1 Register and Control Bit Assignments (Sheet 3 of 3)

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0		
$103B	ODD	EVEN		BYTE	ROW	ERASE	EELAT	EEPGM	PPROG	EEPROM Prog. Control Reg.
$103C	RBOOT	SMOD	MDA	IRV	PSEL3	PSEL2	PSEL1	PSEL0	HPRIO	Highest Priority 1-Bit Int and Misc
$103D	RAM3	RAM2	RAM1	RAM0	REG3	REG2	REG1	REG0	INIT	RAM and I/O Mapping Reg.
$103E	TILOP		OCCR	CBYP	DISR	FCM	FCOP	TCON	TEST1	Factory TEST Control Register
$103F					NOSEC	NOCOP	ROMON	EEON	CONFIG	COP, ROM, and EEPROM Enables

In expanded multiplexed operating modes, memory locations are basically the same as the single-chip operating modes;however, the locations between the shaded areas (designated EXT) are for externally addressed memory and I/O. If an external memory or I/O device is located to overlap an enabled internal resource, the internal resource will take priority. For reads of such an address the data (if any) driving the port C data inputs is ignored and will not result in any harmful conflict with the internal read. For writes to such an address data is driven out of the port C data pins as well as to the internal location. No external devices should drive port C during write accesses to internal locations;however, there is normally no conflict since the external address decode and/or data direction control should incorporate the R/$\overline{W}$ signal in their development. The R/$\overline{W}$, AS, address, and write data signals are valid for all accesses including accesses to internal memory and registers.

The special bootstrap operating mode memory locations are similar to the single-chip operating mode memory locations except that a bootstrap program at memory locations $BF40 through $BFFF is enabled. The reset and interrupt vectors are addressed at $BFC0-$BFFF while in the special bootstrap operating mode. These vector addresses are within the 192 byte memory used for the bootstrap program.

The special test operating mode memory map is the same as the expanded multiplexed operating mode memory map except that the reset and interrupt vectors are located at external memory locations $BFC0-$BFFF.

3.2 RAM AND I/O MAPPING REGISTER (INIT)

There are 64 internal registers which are used to control the operation of the MCU. These registers can be relocated on 4K boundaries within the memory space, using the INIT register. Refer to Table 3.1 for a complete list of the registers. The registers and control bits are explained throughout this document.

The INIT register is a special-purpose 8-bit register which may be used during initialization to change the default locations of RAM and control registers within the

MCU memory map. It may be written to only once within the initial 64E clock cycles after a reset and thereafter becomes a read-only register.

	7	6	5	4	3	2	1	0	
$103D	RAM3	RAM2	RAM1	RAM0	REG3	REG2	REG1	REG0	INIT
RESET	0	0	0	0	0	0	0	1	

The default starting address for internal RAM is $0000 and the default starting address for the 64 control registers is $1000 (the INIT register is set to $01 by reset). The upper four bits of the INIT register specify the starting address for the 512 byte RAM and the lower four bits of INIT specify the starting address for the 64 control registers. These four bits are matched to the upper four bits of the 16-bit address.

Throughout this document, the control register addresses will be displayed with the high-order digit shown as a bold "1" to indicate that the register block may be relocated to some 4K memory page other than its default position of $1000-$103F.

Note that if the RAM is relocated to either $D000, $E000, or $F000, which is in conflict with the internal ROM, (no conflict if the ROMON bit in the configuration register is zero), RAM will take priority and the conflicting ROM will become inaccessible. Also, if the 64 control registers are relocated so that they conflict with the RAM and/or ROM, then the 64 control registers take priority and the RAM and/or ROM at those locations become inaccessible. No harmful conflicts result, the lower priority resources simply become inaccessible. Similarly, if an internal resource conflicts with an external device no harmful conflict results. Data from the external device will not be applied to the internal data bus and cannot interfere with the internal read.

Note that there are unused register locations in the 64 byte control register block. Reads of these unused registers will return data from the undriven internal data bus and not from another resource that happens to be located at the same address.

3.3 ROM

The internal 12K ROM occupies the highest 12K of the memory map ($D000-$FFFF). This ROM is disabled when the ROMON bit in the CONFIG register is clear. The ROMON bit is implemented with an EEPROM cell and is programmed using the same procedures for programming the on-chip EEROM. For further information refer to 3.5.3 System Configuration Register (CONFIG).

In the single-chip operating mode, internal ROM is enabled regardless of the state of the ROMON bit.

There is also a 192 byte mask programmed boot ROM in the TMP68HC11E9. This bootstrap program ROM controls the operation of the special bootstrap operating mode

and is only enabled following reset in the special bootstrap operating mode. For more information refer to 2.2.3 Special Bootstrap Operating Mode.

3.4 RAM

The 512 byte internal RAM may be relocated during initialization by writing to the INIT register. The reset default position is $0000 through $01FF. This RAM is implemented with static cells and retains its contents during the WAIT and STOP modes.

The contents of the 512-byte RAM can also be retained by supplying a low current backup power source to the MODB/V_{STBY} pin. When using a standby power source, V_{DD} may be removed;however, reset must go low before V_{DD} is removed and remain low unit V_{DD} has been restored.

3.5 EEPROM

The 512 bytes of EEPROM are located at $B600 through $B7FF and have the same read cycle time as the internal ROM. The write (or programming) mechanism for the EEPROM is controlled by the PPROG register. The EEPROM is disabled when the EEON bit in the CONFIG register is zero. The EEON bit is implemented with an EEPROM cell.

The erased state of an EEPROM byte is $FF. Programming changes ones to zeros. If any bit in a location needs to be changed from a zero to a one, the byte must be erased in a separate operation before it is reprogrammed. If a new data byte has no ones in bit positions which were already programmed to zero, it is acceptable to program the new data without erasing the EEPROM byte first. For example, programming $50 to a location which was already $55 would change the location to $50.

Programming and erasure of the EEPROM relies on an internal high-voltage charge pump. At E clock frequencies below 2 MHz the efficiency of this charge pump decreases which increases the time required to program or erase a location. The recommended program and erase time is 10 milliseconds when the E clock is 2 MHz and should be increased to as much as 20 milliseconds when E is between 1 MHz and 2 MHz. When the E clock is below 1 MHz, the clock source for the charge pump should be switched from the system clock to an on-chip R-C oscillator clock. This is done by setting the CSEL bit in the OPTION register. A 10 millisecond period should be allowed after setting the CSEL bit to allow the charge pump to stabilize. Note that the CSEL bit also controls a clock to the analog to digital converter subsystem.

3.5.1 EEPROM Programming Control Register (PPROG)

This 8-bit register is used to control programming and erasure of the 512-byte EEPROM. Reset clears this register so the EEPROM is configured for normal reads.

	7	6	5	4	3	2	1	0	
$103B	ODD	EVEN	0	BYTE	ROW	ERASE	EELAT	EEPGM	PPROG
RESET	0	0	0	0	0	0	0	0	

ODD - Program Odd Rows (TEST)

EVEN - Program Even Rows (TEST)

Bit 5 - Not implemented.

This bit always reads zero.

BYTE - Byte Erase Select

This bit overrides the ROW bit.

0 = Row or Bulk Erase

1 = Erase Only One Byte

ROW - Row Erase Select

If the BYTE bit is 1, ROW has no meaning.

0 = Bulk Erase

1 = Row Erase

ERASE - Erase Mode Select

0 = Normal Read or Program

1 = Erase Mode

EELAT - EEPROM Latch Control

0 = EEPROM Address and Data Configured for Read Mode

1 = EEPROM Address and Data Configured for Programming/Erasing

EEPGM - EEPROM Programming Voltage Enable

0 = Programming Voltage Switched Off

1 = Programming Voltage Turned On

If an attempt is made to set both the EELAT and EEPGM bits in the same write cycle, neither will be set. If a write to an EEPROM address is performed while the EEPGM bit is set, the write is ignored and the programming operation currently in progress is not disturbed. These two safeguards were included to prevent accidental EEPROM changes in cases of program runaway.

3.5.2 EEPROM Block Protect Register (BPROT)

This register prevents inadvertent writes to the CONFIG register and to the 512 bytes of EEPROM. The bits in this registers may only be written to zero during the first 64 E clock cycles after reset in the normal modes. Once the bits are set to zero, the associated EEPROM section and/or the CONFIG register may be programmed or erased in the normal manner. The EEPROM is only visible if the EEON bit in the CONFIG register is set to "one". The bits in the BPROT register may be written back to one (in any mode) to protect the EEPROM and/or the CONFIG register, but can only be cleared again if operating in the test or bootstrap modes.

	7	6	5	4	3	2	1	0	
$1035	0	0	0	PTCON	BPRT3	BPRT2	BPRT1	BPRT0	BPROT
RESET	0	0	0	1	1	1	1	1	

Bits 7-5 - Not Implemented

These bits always read as zero

PTCON - Protect CONFIG Register

1 = Programming/erasure of the CONFIG register disabled

0 = Programming/erasure of the CONFIG register allowed

BPRT3-BPRT0 - Block Protect

When set, these bits protect a block of EEPROM from programming and erasure, and when cleared allow programming and erase of the associated block.

Bit	Block Protected	Size
BPRT0	$B600-$B61F	32 Bytes
BPRT1	$B620-$B65F	64 Bytes
BPRT2	$B660-$B6DF	128 Bytes
BPRT3	$B6E0-$B7FF	288 Bytes

3.5.3 Programming/Erasing Internal EEPROM

The EEPROM programming and erasure process is controlled by the PPROG register. The appropriate bits in the BPROT register must be cleared before the EEPROM can be altered.

The following paragraphs describe the various operations performed on the EEPROM and include example program segments to demonstrate programming and erase operations.

These program segments are intended to be simple straightforward examples of the sequences needed for basic program and erase operations. There are no special restrictions on the address modes used and bit manipulation instructions may be used. Other MCU operations can continue to be performed during EEPROM programming and erasure provided these operations do not include reads of data from EEPROM (the EEPROM is disconnected from the read data bus during EEPROM program and erase operations). The subroutine DLY10 used in these program segments is not shown but can be any set of instructions which takes ten milliseconds.

3.5.3.1 Read.

For the read operation the EELAT bit in the PPROG register must be clear. When this bit is cleared, the remaining bits in the PPROG register have no meaning or effect, and the EEPROM may be read as if it were a normal ROM. The block protect register has no effect during reads.

3.5.3.2 Programming.

During EEPROM programming, the ROW and BYTE bits are not used. If the E clock frequency is 1 MHz or less, the CSEL bit in the OPTION register must be set. Recall that in this EEPROM, zeros must be erased by a separate erase operation before programming. The following program segment demonstrates how to program an EEPROM byte.

*On entry, A = data to be programmed and X = an EEPROM address

```
        •
        •
        •
PROG    LDAB    #$02
        STAB    $103B     Set EELAT Bit (EEPGM=0)
        STAA    0,X       Store Data to EEPROM Address
        LDAB    #$03
        STAB    $103B     Set EEPGM Bit (EELAT=1)
        JSR     DLY10     Delay 10ms
        CLR     $103B     Turn Off High Voltage and Set to READ Mode
        •
        •
        •
```

3.5.3.3 Bulk Erase.

The following program segment demonstrates how to bulk erase the 512-byte EEPROM. The CONFIG register is not affected in this example.

```
        •
        •
        •
BULKE   LDAB    #$06
        STAB    #103B   Set to Bulk Erase Mode
        STAB    $B600   Write any Data to any EEPROM Address
        LDAB    #$07
        STAB    $103B   Turn On Programming Voltage
        JSR     DLY10   Delay 10 ms
        CLR     $103B   Turn Off High Voltage and Set to READ Mode
        •
        •
        •
```

3.5.3.4 Row Erase.

The following program segment demonstrates the row erase function. A 'row' is sixteen bytes ($B600-$B60F, $B610-$B61F...$B7F0-$B7FF). This type of erase operation saves time compared to byte erase when large sections of EEPROM are to be erased.

*On entry X = any address in the row to be erased

```
        •
        •
        •
ROWE    LDAB    #$0E
        STAB    $103B   Set to Row Erase Mode
        STAB    0, X    Write any Data to any Address in Row
        LDAB    #$0F
        STAB    $103B   Turn on High Voltage
        JSR     DLY10   Delay 10 ms
        CLR     $103B   Turn Off High Voltage and Set to Read Mode
        •
        •
        •
```

3.5.3.5 Byte Erase.

The following program segment shows the byte erase function.

*On entry, X = address of byte to be erased

```
        •
        •
        •
BYTEE   LDAB    #$16
        STAB    $103B   Set to Row Erase Mode
        STAB    0,X     Write any Data to any Address to Erase
        LDAB    #$17
        STAB    $103B   Turn on High Voltage
        JSR     DLY10   Delay 10 ms
        CLR     $103B   Turn Off High Voltage and Set to Read Mode
        •
        •
        •
```

3.5.4 System Configuration Register (CONFIG)

The TMP68HC11E9 can be configured to specific system requirements through the use of hardwired options such as the mode select pins, semi-permanent EEPROM control bit specifications (CONFIG register), or by use of control registers. The configuration control register (CONFIG) is implemented in EEPROM cells and controls the presence of ROM and EEPROM in the memory map, as well as enabling the COP watchdog system. A security feature to protect data in the EEPROM and RAM is also available on mask programmed TMP68HC11E9s.

	7	6	5	4	3	2	1	0	
$103F	0	0	0	0	NOSEC	NOCOP	ROMON	EEON	CONFIG
RESET	(see 3.5.3.2 OPERATION OF THE CONFIG MECHANISM)								

Bits 7, 6, 5, and 4 - Not Implemented

These bits are always read as zero.

NOSEC - Security Mode Disable Bit

This bit is only implemented if it is specifically requested at the time mask ROM information is requested. When this bit is not implemented it always reads one.
When RAM and EEPROM security are required, the NOSEC bit can be programmed to zero to enable a software anti-theft mechanism. When clear, the NOSEC bit prevents the selection of expanded multiplexed operating modes. If the MCU is reset in the special bootstrap operating mode while NOSEC is zero, EEPROM, RAM, and CONFIG are erased before the loading process continues.

0 = Enable Security Mode

1 = Disable Security Mode

NOCOP - COP System Disable

0 = COP Watchdog System Enabled

1 = COP Watchdog System Disabled

ROMON - Enable On-Chip ROM

When this bit is clear, the 12K ROM is disabled, and that memory space becomes externally accessed space. In the single-chip operating mode, the internal 12K ROM is enabled regardless of the state of the ROMON bit.

EEON - Enable On-Chip EEPROM

When this bit is clear, the 512-byte EEPROM is disabled, and that memory space becomes externally accessed space.

3.5.4.1 Programming and Erasure of the CONFIG Register.

Since the CONFIG register is implemented with EEPROM cells, special provisions must be made to erase and program this register. The normal EEPROM control bits in the PPROG register are used for this purpose. Programming follows the same procedure as programming a byte in the 512-byte EEPROM except the CONFIG register address is used. Erase also follows the same procedure as that used for the EEPROM.

The CONFIG register may be programmed or erased (including byte erase) while the MCU is operating in any mode.

The following program segment demonstrates how to program the CONFIG register. This program assumes that the CONFIG register was previously erased and the PTCON bit in the BPROT register is clear.

*On entry, A = data to be programmed into CONFIG

```
        •
        •
        •
PROGC   LDAB    #$02
        STAB    $103B    Set EELAT Bit (EEPGM = 0)
        STAA    $103F    Store Data to CONFIG Address
        LDAB    #$03
        STAB    $103B    Turn on Programming Voltage
        JSR     DLY10    Delay 10 ms
        CLR     $103B    Turn Off High Voltage and Set to READ Mode
```

•
•
•

The following program segment demonstrates the byte erase procedure for the CONFIG register.

•
•
•

```
BYTEC   LDAB    #$16
        STAB    $103B     Set Byte Erase Mode
        STAB    $103F     Write any Data to CONFIG
        LDAB    #$17
        STAB    $103B     Turn on Programming Voltage
        JSR     DLY10     Delay 10 ms
        CLR     $103B     Turn Off High Voltage and Set to READ Mode
```

•
•
•

3.5.4.2 Operation of the Configuration Mechanism.

The CONFIG register consists of an EEPROM byte and static working latches. This register controls the startup configuration of the MCU. The contents of the EEPROM CONFIG byte are transferred into static working latches during any reset sequence. The operation of the MCU is controlled directly by these latches and not the actual EEPROM byte. Changes to the EEPROM byte do not affect operation of the MCU until after the next reset sequence. When programming the CONFIG register, the EEPROM byte is being accessed. When the CONFIG register is being read, the static latches are being accessed.

To change the value in the CONFIG register proceed as follows:

1) Erase the CONFIG register.

Note : Do not issue a reset at this time.

2) Program the new value to the CONFIG register.

3) Issue a reset so the new configuration will take effect.

4. PARALLEL I/O

The TMP68HC11E9 has 40 I/O pins arranged as five 8-bit ports. All of these pins serve multiple functions depending on the operating mode and data in the control register. This section explains the operation of these pins only when they are used for parallel I/O.

Ports C and D are used as general purpose input and/or output pins under direct control of their respective data direction registers. Ports A, B, and E, with the exception of port A pins 3 and 7, are fixed direction inputs or outputs and therefore do not have data direction registers. Port B, port C, the STRA pin, and the STRB pin are used for strobed and/or handshake modes of parallel I/O, as well as general purpose I/O.

4.1 GENERAL PURPOSE I/O (PORTS C AND D)

Each port I/O line has an associated bit in a specific port data register and port data direction register. The data direction register bits are used to specify the primary direction of data for each I/O line. When an output line is read, the value at the input to the pin driver is returned. When a line is configured as an input, that pin becomes a high-impedance input. If a write is executed to an input line, the value does not affect the I/O pin, but is stored in an internal latch. When the line becomes an output, this value appears at the I/O pin. Data direction register bits are cleared by reset to configure I/O pins as inputs.

The AS and R/$\overline{W}$ pins are dedicated to bus control while in the expanded multiplexed operating modes, or parallel I/O strobes (STRA and STRB) while in the single chip operating modes.

4.2 FIXED DIRECTION I/O (PORTS A, B, AND E)

The lines for ports A, B, and E (except for port A bits 3 and 7) have fixed data directions. When port A is being used for general purpose I/O, bits 0, 1, and 2 are configured as input only and writes to these lines have no effect. Bits 4, 5, and 6 of port A are configured as output only and reads of these lines return the levels sensed at the input to the line drivers. Port A bits 3 and 7 can be configured as either a general-purpose input or output using the DDRA3 and DDRA7 bit in the pulse accumulator control register. When port B is being used for general purpose output, it is configured as output only and reads of these lines will return the levels sensed at the input of the pin drivers. Port E contains the eight A/D channel inputs, but these lines may also be used as general purpose digital inputs. Writes to the port E address have no effect.

4.3 SIMPLE STROBED I/O

The simple strobed mode of parallel I/O is invoked and controlled by the parallel I/O control register (PIOC). This mode is selected when the handshake bit (HNDS) in the PIOC register is clear. Port C becomes a strobed input port with the STRA line as the

edge-detecting latch command input. Also, port B becomes a strobed output port with the STRB line as the output strobe. The logic sense of the STRB output is selected by the invert strobe B bit (INVB) in the PIOC register.

4.3.1 Strobed Input Port C

In this mode, there are two addresses where port C may be read, the PORTC data register and the alternate latched port C register (PORTCL). The data direction register still controls the data direction of all port C lines. Even when the strobed input mode is selected, any or all of the port C lines may still be used for general purpose I/O.

The STRA line is used as an edge-detecting input, and the edge-select for strobe A (EGA) bit in the PIOC register defines either falling or rising edge as the significant edge. Whenever the selected edge is detected at the STRA pin, the current logic levels at port C lines are latched into the PORTCL register and the strobe A flag (STAF) in the PIOC register is set. If the strobe A interrupt enable (STAI) bit in PIOC is also set, an internal interrupt sequence is requested. The strobe A flag (STAF) is automatically cleared by reading the PIOC register (with STAF set) followed by a read of the PORTCL register. Data is latched in the PORTCL register whether or not the STAF flag was previously clear.

4.3.2 Strobed Output Port B

In this mode, the STRB pin is a strobe output which is pulsed for two E clock periods each time there is a write to port B. The INVB bit in the PIOC register controls the polarity of the pulse on the STRB line.

4.4 FULL HANDSHAKE I/O

The full handshake modes of parallel I/O involve port C and the STRA and STRB lines. There are two basic modes (input and output) and an additional variation on the output handshake mode that allows three-stated operation of port C. In all handshake modes, STRA is an edge-detecting input, and STRB is a handshake output line.

When full input handshake protocol is specified, both general purpose input and/or general purpose output can coexist at port C. When full output handshake protocol is specified, general purpose output can coexist with the handshake outputs at port C, but the three-state feature of the output handshake mode interferes with general purpose input in two ways. First, in full output handshake, the port C lines are outputs whenever STRA is at its active level regardless of the data direction register bits. This potentially conflicts with any external device trying to drive port C unless that external device has an open-drain type output driver. Second, the value returned on reads of port C is the state of the outputs of an internal port C output latch regardless of the states of the data direction register bits, so that the data written for output handshake can be read even if the pins are in a three-state condition.

4.4.1 Input Handshake Protocol

In the input handshake protocol, port C is a latching input port, STRA is an edge-sensitive latch command from the external system that is driving port C, and STRB is a "ready" output line controlled by logic in the MCU.

When a "ready" condition is recognized, the external device places data on the port C lines, then pulses the STRA line. The active edge on the STRA line latches the port C data into the PORTCL register, sets the STAF flag (optionally causing an interrupt), and deasserts the STRB line. Deassertion of the STRB line automatically inhibits the external device from strobing new data into port C. Reading the PORTCL latch register (independent of clearing the STAF flag) asserts the STRB line, indicating that new data may now be applied to port C.

The STRB line can be configured (with the PLS control bit) to be a pulse output (pulse mode) or a static output (interlocked mode).

The port C data direction register bits should be cleared for each line that is to be used as a latched input line. However, some port C lines can be used as latched inputs with the input handshake protocol while, at the same time, using some port C lines as static inputs, and some port C lines as static outputs. The input handshake protocol has no effect on the use of port C lines as static inputs or as static outputs. Reads of the PORTC data register always return the static logic level at the port C lines (for lines configured as inputs). Writes to either the PORTC data register or the alternate latched port C register (PORTCL) send information to the same port C output register without affecting the input handshake strobes.

4.4.2 Output Handshake Protocol

In the output handshake protocol, port C is an output port, STRB is a "read" output, and STRA is an edge-sensitive acknowledge input signal, used to indicate to the MCU that the output data has been accepted by the external device. In a variation of this output handshake protocol, STRA is also used as an output-enable input, as well as an edge-sensitive acknowledge input.

The MCU places data on the port C output lines and then indicates stable data is available by asserting the STRB line. The external device then processes the available data and pulses the STRA line to indicate that new data may be placed on the port C output lines. The active edge on the STRA line causes the STRB line to be deasserted and the STAF status flag to be set. In response to the STAF bit being set, the program transfers new data out of port C as required. Writing data to the PORTCL register causes the data to appear on port C lines and asserts the STRB line.

There is a variation to the output handshake protocol that allows three-state operation on port C. It is possible to directly connect this 8-bit parallel port to other three-state devices with no additional parts.

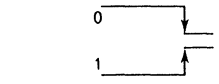
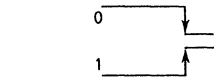
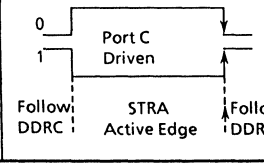
While the STRA input line is inactive, all port C lines obey the data direction specified by the data direction register so that lines which are configured as inputs are high impedance. When the STRA line is activated, all port C lines are forced to outputs regardless of the data in the data direction register. Note that in output handshake protocol, reads of port C always return the value sensed at the input to the output buffer regardless of the state of the data direction register bits because the lines would not necessarily have meaningful data on them in the three-state variation of this protocol. This operation makes it inpracticle to use some port C lines as static inputs, while using others as handshake output, but does not interfere with the use of some port C lines as static outputs. Port C lines intended as static outputs or normal handshake outputs should have their corresponding data direction register bits set, and lines intended as three-state handshake outputs should have their corresponding data direction register bits clear.

4.5 PARALLEL I/O CONTROL REGISTER (PIOC)

The parallel handshake I/O functions are available only in the single-chip operating mode. The PIOC is a read/write register except for bit 7 which is read only. Table 4.1 shows a summary of handshake I/O operations.

Table 4.1 Handshake I/O Opertions Summary

	STAI	CWOM	INVB
0	STAF Interrupts Inhibted	Port C Outputs Normal	STRB Active Low
1	STAF Interrupts Enabled	Port C Outputs Open-Drain	STRB Active High

	STAF Clearing Sequence[1]	HNDS	OIN	PLS	EGA	Port C	Port B
Simple Strobe Mode	Read PIOC with STAF = 1 then Read PORTCL	0	×	×		Inputs latched into PORTCL on any active edge on STRA.	STRB pulses on writes to port B.
Full Input Handshake	Read PIOC with STAF = 1 then Read PORTCL	1	0	0 = STRB Active Level 1 = STRB Active Pulse		Inputs latched into PORTCL on any active edge on STRA.	Normal output port. Unaffected in handshake modes.
Full Output Handshake	Read PIOC with STAF = 1 then Write to PORTCL	1	1	0 = STRB Active Level 1 = STRB Active Pulse		Driven as outputs is STRA at active level. Follows DDRC if STRA not at active level.	Normal output port. Unaffected in handshake modes.

Note :

1. Set by active edge on STRA.

	7	6	5	4	3	2	1	0	
$1002	STAF	STAI	CWOM	HNDS	OIN	PLS	EGA	INVB	PIOC
RESET	0	0	0	0	0	U	1	1	

STAF - Strobe A Interrupt Status Flag

This bit is set when a selected edge occurs on strobe A. Clearing it depends on the state of HNDS and OIN bits. In simple strobed mode or in full input handshake mode, STAF is cleared by reading the PIOC register with STAF set followed by reading the PORTCL register. In output handshake, STAF is cleared by reading the PIOC register with STAF set followed by writing to the PORTCL register.

STAI - Strobe A Interrupt Enable Mask

When the 1 bit in the condition code register is clear and STAI is set, STAF (when set) will request an interrupt.

CWOM - Port C Write-OR Mode

CWOM affects all eight port C pins to together

0 = Port C outputs are normal CMOS outputs

1 = Port C outputs act as open-drain outputs

HNDS - Handshake Mode

When clear, strobe A acts as a simple input strobe to latch data into PORTCL, and strobe B acts as a simple output strobe which pulses after a write to port B. When set, a handshake protocol involving port C, STRA, and STRB is selected (see the definition for the ON bit).

0 = Simple strobe mode

1 = Full input or output handshake mode

OIN - Output or Input Handshaking

This bit has no meaning when HNDS = 0.

0 = Input handshake

1 = Output handshake

PLS - Pulse/Interlocked Handshake Operation

This bit has no meaning if HNDS = 0. When interlocked handshake operation is selected, strobe B, once activated, stays active until the selected edge of strobe A is detected. When pulsed handshake operation is selected, strobe B is pulsed for two E cycles.

0 = Interlocked handshake selected

1 = Pulsed handshake selected

EGA - Active Edge for Strobe A

0 = Falling edge of STRA is selected. When output handshake is selected, port C lines obey the data direction register while STRA is low, but port C is forced to output when STRA is high.

1 = Rising edge of STRA is selected. When output handshake is selected, port C lines obey the data direction register while STRA is high, but port C is forced to output when STRA is low.

INVB - Invert Strobe B

0 = Active level is logic zero

1 = Active level is logic one

5. SERIAL COMMUNICATIONS INTERFACE (SCI)

This section contains a description of the serial communication interface (SCI).

5.1 OVERVIEW AND FEATURES

A full-duplex asynchronous Serial Communications Interface (SCI) is provided with a standard NRZ format (one start bit, eight or nine data bits, and one stop bit) and a variety of baud rates. The SCI transmitter and receiver are functionally independent, but use the same data format and bit rate. "Baud" and "bit rate" are used synonymously in the following description.

SCI Two-Wire System Features

- Standard NRZ (mark/space) format.
- Advanced error detection method includes noise detection for noise duration of up to 1/16 bit time.
- Full-duplex operation.
- Software programmable for one of 32 different baud rates.
- Software selectable word length (eight or nine bit words).
- Separate transmitter and receiver enable bits.
- Capable of being interrupt driven.
- Four separate enable bits available for interrupt control.

SCI Receiver Features

- Receiver wake-up function (idle or address bit).
- Idle line detect.
- Framing error detect.
- Noise detect.
- Overrun detect.
- Receiver data register full flag.

SCI Transmitter Features

- Transmit data register empty flag.
- Transmit complete flag.
- Send break.

5.2 DATA FORMAT

Receive data or transmit data is the serial data which is transferred to the internal data bus from the receive data input pin (RxD), or from the internal bus to the transmit data output pin (TxD). The non-return-to-zero (NRZ) data format shown in Figure 5.1 is used and must meet the following criteria:

(1) The idle line is brought to a logic one state prior to transmission/reception of a character.

(2) A start bit (logic zero) is used to indicate the start of a frame.

(3) The data is transmitted and received least-significant-bit first.

(4) A stop bit (logic one) is used to indicate the end of a frame. A frame consists of a start bit, a character of eight or nine data bits, and a stop bit.

(5) A break is defined as the transmission or reception of a low (logic zero) for at least one complete frame time.

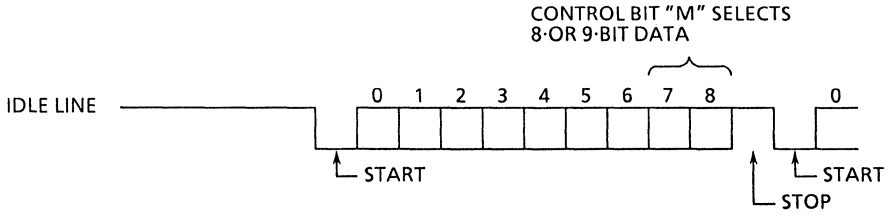


Figure 5.1 Data Format

5.3 WAKE-UP FEATURE

The receiver wake-up feature reduces SCI service overhead in multiple receiver systems. Software in each receiver evaluates the first character(s) of each message. If the message is intended for a different receiver, the SCI can be placed in a sleep mode so that the rest of the message will not generate requests for service. Whenever a new message is started, logic in the sleeping receivers causes them to wake up so they can evaluate the initial character(s) of the new message.

A sleeping SCI receiver can be configured (using the WAKE control bit in serial communications control register 1 (SCCR1)) to wake up using either of two methods:idle line wake up or address mark wake up.

In idle line wake up, a sleeping receiver wakes up as soon as the RxD line becomes idle. Idle is defined as a continuous logic high on the RxD line for ten (or eleven) full bit times. Systems using this type of wake up must provide at least one character time of

idle between messages to wake up sleeping receivers but must not allow any idle time between characters within a message.

In address mark wake up, the most significant bit (MSB) in a character is used to indicate that the character is an address (1) or a data (0) character. Sleeping receivers will wake up whenever an address character is received. Systems using this method for wake up would set the MSB of the first character in each message and leave it clear for all other characters in the message. Idle periods may be present within messages and no idle time is required between messages for this wake up method.

5.4 RECEIVE DATA (RxD)

Receive data is the serial data which is applied through the input line and the serial communications interface to the internal bus. The receiver circuitry clocks the input at a rate equal to 16 times the baud rate and this time is referred to as the RT clock.

Once a valid start bit is detected, the start bit, each data bit, and the stop bit are sampled three times at RT intervals 8 RT, 9 RT, and 10 RT (1 RT is the position where the bit is expected to start), as shown in Figure 5.2 The value of the bit is determined by voting logic which takes the value of the majority of samples.

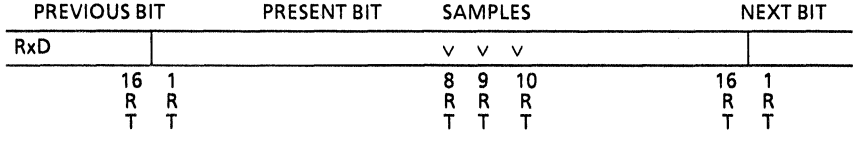


Figure 5.2 Sampling Technique Used on All Bits

5.5 START BIT DETECTION

When the RxD input is detected low, it is tested for three more sample times (referred to as the start edge verification samples in Figure 5.3). If at least two of these three verification samples detect a logic zero, a valid start bit has been detected, otherwise the line is assumed to be idle. A noise flag is set if all three verification samples do not detect a logic zero. A valid start bit could be assumed with a set noise flag present.

If there has been a framing error without detection of a break (10 zeros for 8-bit format or 11 zeros for 9-bit format), the circuit continues to operate as if there actually was a stop bit and the start edge will be placed artificially. The last bit received in the data shift register is inverted to a logic one, and the three logic one start qualifiers (shown in Figure 5.3) are forced into the sample shift register during the interval when detection of a start bit is anticipated (see Figure 5.4);therefore, the start bit will be accepted no sooner than it is anticipated.

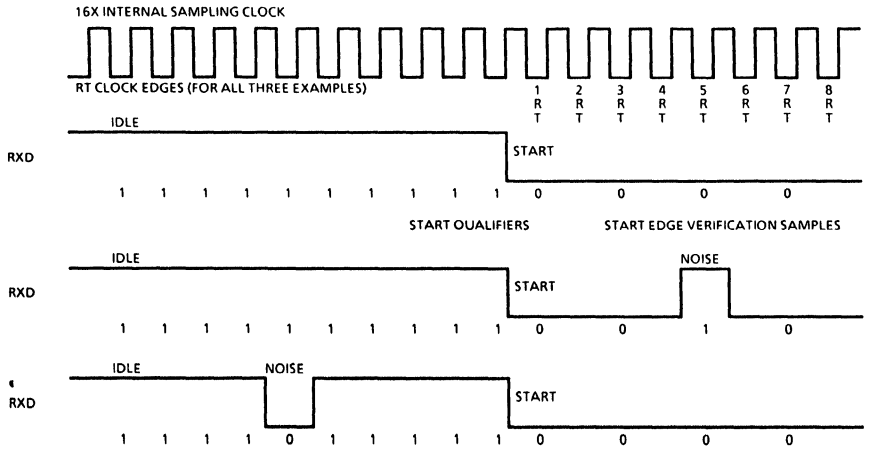
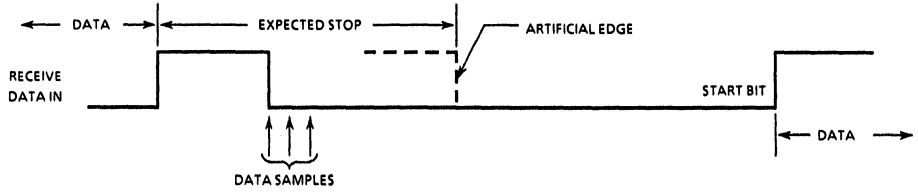
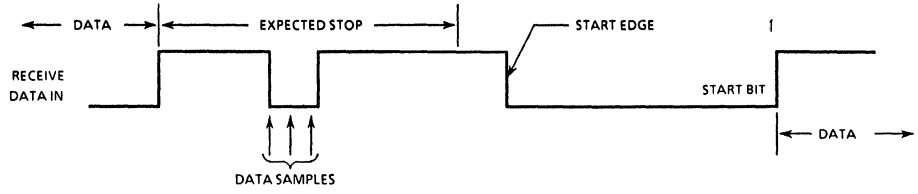


Figure 5.3 Examples of Start Bit Sampling Techniques



(a) Case1, Receive Line Low During Artificial Edge



(b) Case2, Receive Line High During Expected Start Edge

Figure 5.4 SCI Artifical Start Following a Framing Error

If the receiver detects that a break produced the framing error, the start bit will not be artificially induced and the receiver must actually detect a logic one before the start bit can be recognized. See Figure 5.5.

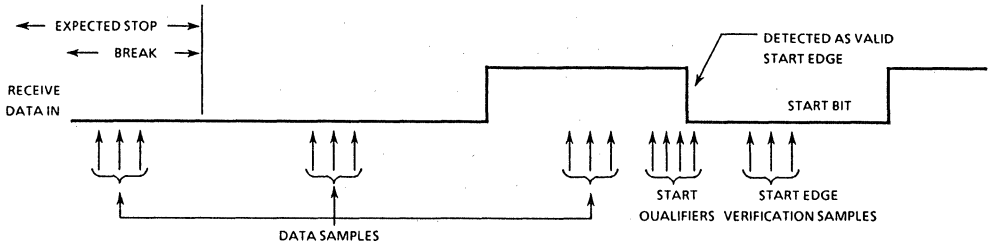


Figure 5.5 SCI Start Bit Following a Break

5.6 TRANSMIT DATA (TxD)

Transmit data is the serial data from the internal data bus which is applied through the serial communications interface to the output line. The transmitter generates a bit time by using a derivative of the RT clock, thus producing a transmission rate equal to 1/16 that of the receiver sample clock.

5.7 FUNCTIONAL DESCRIPTION

A block diagram of the SCI is shown in Figure 5.6. The user has option bits in serial communications control register 1 (SCCR1) to determine the "Wake-up" method (WAKE bit) and data word length (M bit) of the SCI. Serial communications control register 2 (SCCR2) provides control bits which individually enable/disable the transmitter or receiver (TE and RE, respectively), enable system interrupts (TIE, TCIE, ILIE) and provide the wake-up enable bit (RWU) and the send break code bit (SBK). The baud rate register (BAUD) bits allow the user to select different baud rates which may be used as the rate control for the transmitter and receiver.

Data transmission is initiated by a write to the serial communications data register (SCDR). Provided the transmitter is enabled, data stored in the SCDR register is transferred to the transmit data shift register. This transfer of data sets the TDRE bit of the SCI status register (SCSR) and may generate an interrupt if the transmit interrupt is enabled. The transfer of data to the transmit data shift register is synchronized with the bit rate clock (Figure 5.7). All data is transmitted LSB first. Upon completion of data transmission, the transmission complete (TC) bit of the SCSR register is set (provided no pending data, preamble, or break is to be sent), and an interrupt may be generated if the transmit complete interrupt is enabled. If the transmitter is disabled, and the data, preamble, or break (in the transmit shift register) has been sent, the TC bit will also be set. This will also generate an interrupt if the TCIE bit is set. If the transmitter is disabled in the middle of a transmission, that character will be completed before the transmitter gives up control of the TxD pin.

When the SCDR register is read, it contains the last data byte received, provided that the receiver is enabled. The RDRF bit of the SCSR register is set to indicate that a data byte has been transferred from the input serial shift register to the SCDR register, which can cause an interrupt if the receiver interrupt is enabled. The data transfer from the input serial shift register to the SCDR register is synchronized by the receiver bit rate clock. The OR (over-run), NF (noise), or FE (framing) error bits of the SCSR register may be set if data reception errors occurred.

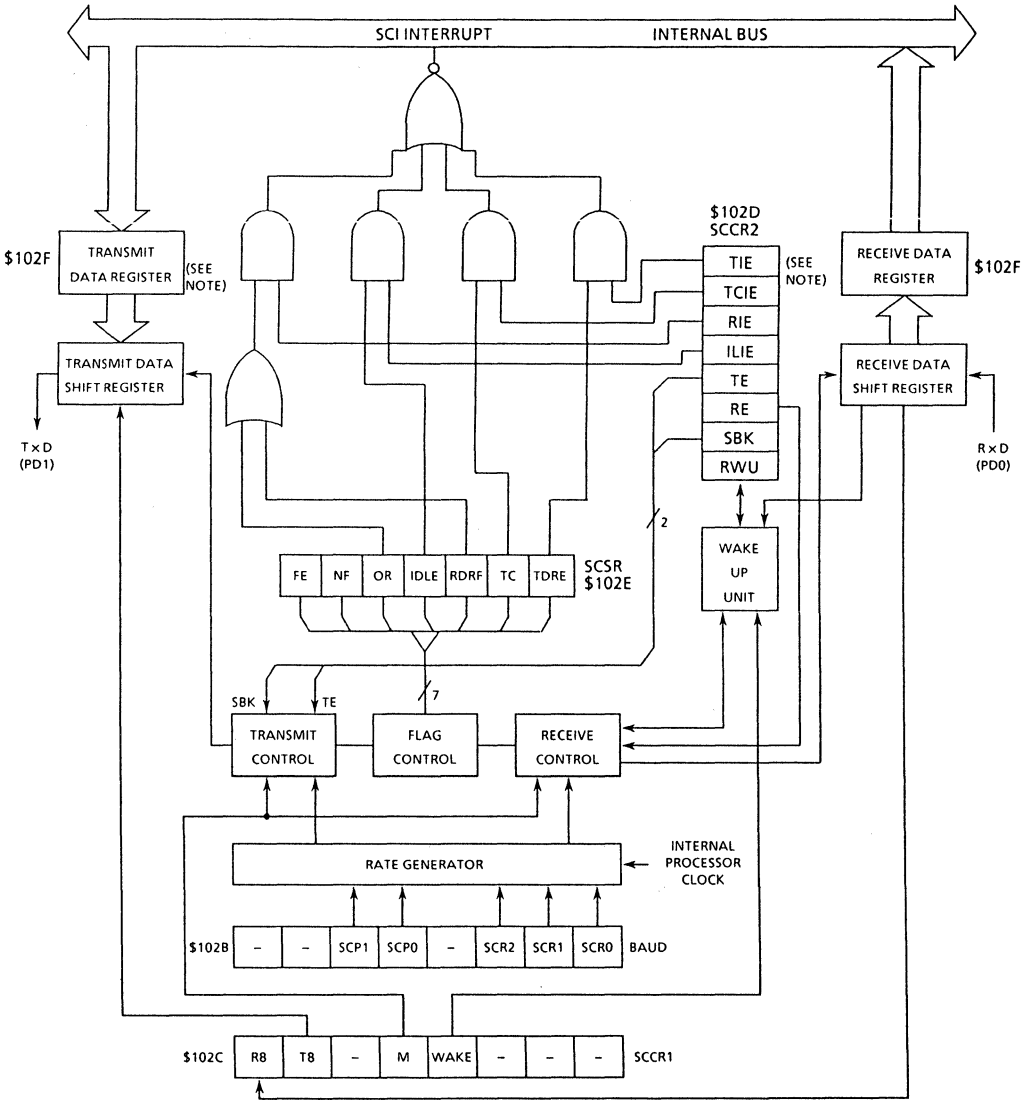
An idle line interrupt is generated if the idle line interrupt is enabled and the IDLE bit (which detects idle line transmission) of SCSR register is set. This allows a receiver that is not in the wake-up mode to detect the end of a message, the preamble of a new message, or to resynchronize with the transmitter. A valid character must be received before the idle line condition or the IDLE bit will not be set and an idle interrupt will not be generated.

5.8 SCI REGISTERS

There are five registers used in the serial communications interface and the operation of these registers is discussed in the following paragraphs. Reference should be made to the block diagram shown in Figure 5.6.

5.8.1 Serial Communications Data Register (SCDR)

The serial communications data register performs two functions;i.e., it acts as the receive data register when it is read and as the transmit data register when it is written. Figure 5.6 shows this register as two separate registers, namely:the receive data register and the transmit data register.



Note : The Serial Communications Data Register (SCDR) is controlled by the internal R/$\overline{W}$ signal. It is the transmit data register when witten and receive data register when read.

Figure 5.6 Serial Communications Interface Block Diagram

5.8.2 Serial Communications Control Register 1 (SCCR1)

The serial communications control register 1 (SCCR1) provides the control bits which: (1) determine the word length, and (2) select the method used for the wake-up feature.

	7	6	5	4	3	2	1	0	
$102C	R8	T8	0	M	WAKE	0	0	0	SCCR1
RESET	U	U	0	0	0	0	0	0	

R8 - Receive Data Bit 8

If the M bit is set, this bit provides a storage location for the ninth bit in the receive data character.

T8 - Transmit Data Bit 8

If the M bit is set, this bit provides a storage location for the ninth bit in the transmit data character. It is not necessary to write to this bit for every character transmitted, only when the sense is to be different than that for the previous character.

Bit 5 - Not Implemented

This bit always reads zero.

M - CI Character Length

0 = 1 start bit, 8 data bits, 1 stop bit

1 = 1 start bit, 9 data bits, 1 stop bit

WAKE - Wake Up Method Select

0 = Idle Line

1 = Address Mark

Bits 2, 1 - Not Implemented

These bits always read zero.

5.8.3 Serial Communications Control Register 2 (SCCR2)

The serial communications control register 2 (SCCR2) provides the control bits which enable/disable individual SCI functions.

	7	6	5	4	3	2	1	0	
$102D	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK	SCCR2
RESET	0	0	0	0	0	0	0	0	

TIE - Transmit Interrupt Enable

0 = TDRE interrupts disabled

1 = SCI interrupt if TDRE = 1

TCIE - Transmit Complete Interrupt Enable

0 = TC interrupts disabled

1 = SCI interrupt if TC = 1

REI - Receive Interrupt Enable

0 = RDRF and OR interrupts disabled

1 = SCI interrupt if RDRF or OR = 1

ILIE - Idle Line Interrupt Enable

0 = IDLE interrupts disabled

1 = SCI interrupt if IDLE = 1

TE - Transmit Enable

When the transmit enable (TE) bit is set, the transmit shift register output is applied to the TxD line. Depending on the state of control bit M (SCCR1), a preamble of 10 (M = 0) or 11(M = 1) consecutive ones is transmitted when software sets the TE bit from a cleared state. After loading the last byte in the serial communications data register and receiving the TDRE flag, the user can clear TE bit. Transmission of the last byte will then be completed before the transmitter gives up control of the TxD pin. While the transmitter is active, the data direction register control for port D bit 1 is overridden and the line is forced to be an output.

RE - Receive Enable

When the receive enable (RE) bit is set, the receiver is enabled. When RE bit is clear, the receiver is disabled and all of the status bits associated with the receiver (RDRF, IDLE, OR, NF and FE) are inhibited. While the receiver is enabled, the data direction register control for port D bit 0 is overridden and the line is forced to be an input.

RWU - Receiver Wake Up

When the receiver wake-up (RWU) bit is set by the user's software, it puts the receiver to sleep and enables "wake up" function. If the WAKE bit is cleared, RWU bit is cleared by the SCI logic after receiving 10 (M=0) or 11 (M=1) consecutive ones. If the WAKE bit is set, RWU bit is cleared by the SCI logic after receiving a data word whose MSB is set.

SBK - Send Break

If the send break (SBK) bit is toggled and cleared, the transmitter sends 10 (M=0) or 11 (M=1) zeros and then reverts to idle or sending data. If SBK bit remains set, the transmitter will continually send whole blocks of zeros (sets of 10 or 11) until cleared. At the completion of the break code, the transmitter sends at least one high bit to guarantee recognition of a valid start bit. If the transmitter is currently empty and idle, setting and clearing SBK bit is likely to queue two character times of break because the first break transfers almost immediatly to the shift register and the second is then queued into the parallel transmit buffer.

5.8.4 Serial Communications Status Register (SCSR)

The serial communications status register (SCSR) provides inputs to the interrupt logic circuits for generation of the SCI system interrupt.

	7	6	5	4	3	2	1	0	
$102E	TDRE	TC	RDRF	IDLE	OR	NF	FE	0	SCSR
RESET	1	1	0	0	0	0	0	0	

TDRE - Transmit Data Register Empty

The transmit data register empty (TDRE) bit is set to indicate that the content of the serial communications data register have been transferred to the transmit serial shift register. This bit is cleared by reading the SCSR register (with TDRE=1) followed by a write to the SCDR register .

TC - Transmit Complete

The transmit complete (TC) bit is set at the end of a data frame, preamble, or break condition if:

1) TE=1, TDRE=1, and no pending data, preamble, or break is to be transmitter; or

2) TE=0, and the data, preamble, or break in the transmit shift register has been transmitted.

The TC bit is a status flag which indicates that one of the above conditions have occurred. The TC bit is cleared by reading the SCSR register (with TC set) followed by a write to the SCDR register.

RDRF - Receive Data Register Full

The receive data register full (RDRF) bit is set when the receiver serial shift register is transferred to the SCDR register. The RDRF bit is cleared when the SCSR register is read (with RDRF set) followed by a read of the SCDR register.

IDLE - Idle Line Detect

The idle line detect (IDLE) bit, when set, indicates the receiver has detected an idle line. The IDLE bit is cleared by reading the SCSR register with IDLE bit set followed by reading SCDR register. Once the IDLE status flag is cleared, it will not be set again until after the RxD line has been active and becomes idle again.

OR - Overrun Error

The overrun error (OR) bit is set when the next byte is ready to be transferred from the receive shift register to the SCDR register which is already full (RDRF bit is set). When an overrun error occurs, the data which caused the overrun is lost and the data which was already in SCDR register is not disturbed. The OR bit is cleared when the SCSR register is read (with OR bit set), followed by a read of the SCDR register.

NF - Noise Flag

The noise flag (NF) bit is set if there is noise on any of the received bits, including the start and stop bits. The NF bit is not set until the RDRF flag is set. The NF bit is cleared when the SCSR register is read (with NF bit set), followed by a read of the SCDR register.

FE - Framing Error

The framing error (FE) bit is set when no stop bit was detected in the received data character. The FE bit is set at the same time as the RDRF bit is set. If the byte received causes both framing and overrun errors, the processor will only recognize the overrun error. The framing error flag inhibits further transfer of data into the SCDR register until it is cleared. The FE bit is cleared when the SCSR register is read (with FE bit equal to one) followed by a read of the SCDR register.

Bit 0 - Not Implemented

This bit always reads zero.

5.8.5 Baud Rate Register (BAUD)

The baud rate register selects the different baud rates which may be used as the rate control for the transmitter and receiver. The SCP0-SCP1 bits function as a prescaler

for the SCR0-SCR2 bits. Together, these five bits provide multiple baud rate combinations for a given crystal frequency.

	7	6	5	4	3	2	1	0	
$102B	TCLR	0	SCP1	SCP0	RCKB	SCR2	SCR1	SCR0	BAUD
RESET	0	0	0	0	0	U	U	U	

TCLR - Clear Baud Rate Counters (Test)

This bit is used to clear the baud rate counter chain during factory testing. TCLR bit is zero and cannot be set while in normal operating modes.

SCP1 and SCP0 - SCI Baud Rate Prescaler Selects

The E clock is divided by the factors shown in Table 5.1. This prescaled output provides an input to a divider which is controlled by the SCR2-SCR0 bits.

Table 5.1 Second Prescaler Stage

SCR1	SCR0	Internal Processor Clock Divide By
0	0	1
0	1	3
1	0	4
1	1	13

SCR2, SCR1, and SCR0 - SCI Baud Rate Selects

These three bits select the baud rates for both the transmitter and the receiver. The prescaler output described above is further divided by the factors shown in Table 5.2.

Table 5.2 Second Prescaler Stage

SCR2	SCR1	SCR0	Prescaler Output Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

RCKB - SCI Baud Rate Clock Check (Test)

This bit is used during factory testing to enable the exclusive-OR of the receiver clock and transmitter clock to be driven out the TxD pin. RCKB bit is zero and cannot be set while in normal operating modes.

The diagram shown in Figure 5.7 and the data given in Tables 5.3 and 5.4 illustrate the divider chain used to obtain the baud rate clock. Note that there is a fixed rate divide-by-16 between the receive clock (RT) and the transmit clock (Tx). The actual divider chain is controlled by the combined SCP0-SCP1 and SCR0-SCR2 bits in the baud rate register as illustrated.

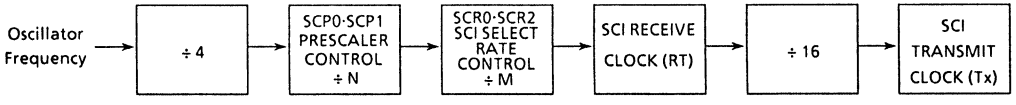


Figure 5.7 Rate Generator Division

Table 5.3 Prescaler Highest Baud Rate Frequency Output

SCP Bit		Clock* Divided By	Crystal Frequency (MHz)				
1	0		8.3886	8.0	4.9152	4.0	3.6864
0	0	1	131.072 K Baud	125.000 K Baud	76.80 K Baud	62.50 K Baud	57.60 K Baud
0	1	3	43.691 K Baud	41.666 K Baud	25.60 K Baud	20.833 K Baud	19.20 K Baud
1	0	4	32.768 K Baud	31.250 K Baud	19.20 K Baud	15.625 K Baud	14.40 K Baud
1	1	13	10.082 K Baud	9600 Baud	5.907 K Baud	4800 Baud	4430 Baud

*The cloick in the "Clock Divided By" column is the internal processor clock.

Note: The divided frequencies shown in Table5.3 represent baud rates which are the highest transmit baud rate (Tx) that can be obtained by a specific crystal frequency and only using the prescaler division. Lower baud rates may be obtained by providing a further division using the SCI rate select bits as shown below for some representative prescaler outputs.

Table 5.4 Transmit Baud Rate Output for a Given Prescaler Output

SCR Bits			Divided By	Representative Highest Prescaler Baud Rate Output				
2	1	0		131.072 K Baud	32.768 K Baud	76.80 K Baud	19.20 K Baud	9600 Baud
0	0	0	1	131.072 K Baud	32.768 K Baud	76.80 K Baud	19.20 K Baud	9600 Baud
0	0	1	2	65.536 K Baud	16.384 K Baud	38.40 K Baud	9600 Baud	4800 Baud
0	1	0	4	32.768 K Baud	8.192 K Baud	19.20 K Baud	4800 Baud	2400 Baud
0	1	1	8	16.384 K Baud	4.096 K Baud	9600 Baud	2400 Baud	1200 Baud
1	0	0	16	8.192 K Baud	2.048 K Baud	4800 Baud	1200 Baud	600 Baud
1	0	1	32	4.096 K Baud	1.024 K Baud	2400 Baud	600 Baud	300 Baud
1	1	0	64	2.048 K Baud	512 Baud	1200 Baud	300 Baud	150 Baud
1	1	1	128	1.024 K Baud	256 Baud	600 Baud	150 Baud	75 Baud

Note: Table5.4 illustrates how the SCI select bits can be used to provide lower transmitter baud rates by further dividing the prescaler output frequency. The five examples are only representative samples. In all cases, the baud rates shown are transmit baud rates (transmit clock) and the receiver clock is 16 times higher in frequency than the actual baud rate.

6. SERIAL PERIPHERAL INTERFACE (SPI)

This section contains a description on the serial peripheral interface (SPI).

6.1 OVERVIEW AND FEATURES

The serial peripheral interface (SPI) is a synchronous interface which allows several SPI microcontrollers or SPI-type peripherals to be interconnected. 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. The TMP68HC11E9 SPI system may be configured either as a master or as a slave.

Features 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
- Easily Interfaces to Simple Expansion Parts (PLLs, D/As, Latches, Display Drivers, etc.)

6.2 SPI SIGNAL DESCRIPTIONS

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

Any SPI output line has to have its corresponding data direction register bit set. If this bit is clear, the line is disconnected from the SPI logic and becomes a general-purpose input line. Any SPI input line is forced to act as an input regardless of what is in the corresponding data direction register bit.

6.2.1 Master In Slave Out (MISO)

The MISO line is configured as an input in a master device and as an output in a slave device. It is one of the two lines that transfer serial data in one direction, with the most significant bit sent first. The MISO line of a slave device is placed in the high-impedance state if the slave is not selected.

6.2.2 Master Out Slave In (MOSI)

The MOSI line is configured as an output in a master device and an input in a slave device. It is one of the two lines that transfer serial data in one direction with the most significant bit sent first.

6.2.3 Serial Clock (SCK)

The serial clock is used to synchronize data movement both in and out of the device through its MOSI and MISO lines. The master and slave devices are capable of exchanging a byte of information during a sequence of eight clock cycles. Since SCK is generated by the master device, this line becomes an input on a slave device.

As shown in Figure 6.1, four possible timing relationships may be chosen by using control bits CPOL and CPHA in the serial peripheral control register (SPCR). Both master and slave devices must operation with the same timing. The master device always places data on the MOSI line a half-cycle before the clock edge (SCK), in order for the slave device to latch the data.

Two bits (SPR0 and SPR1) in the SPCR of the master device select the clock rate. In a slave device, SPR0 and SPR1 have no effect on the operation on the SPI.

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

The slave select ($\overline{SS}$) input line is used to select a slave device. It has to be low prior to data transactions and must stay low for the duration of transaction.

The $\overline{SS}$ line on the master must be tied high. If it goes low, a mode fault error flag (MODF) is set in the serial peripheral status register (SPSR). The $\overline{SS}$ pin can be selected to be a general-purpose output by writing a one in bit 5 of the port D data direction register, thus disabling the mode fault circuit. The other three SPI lines are dedicated to the SPI whenever the SPI is on.

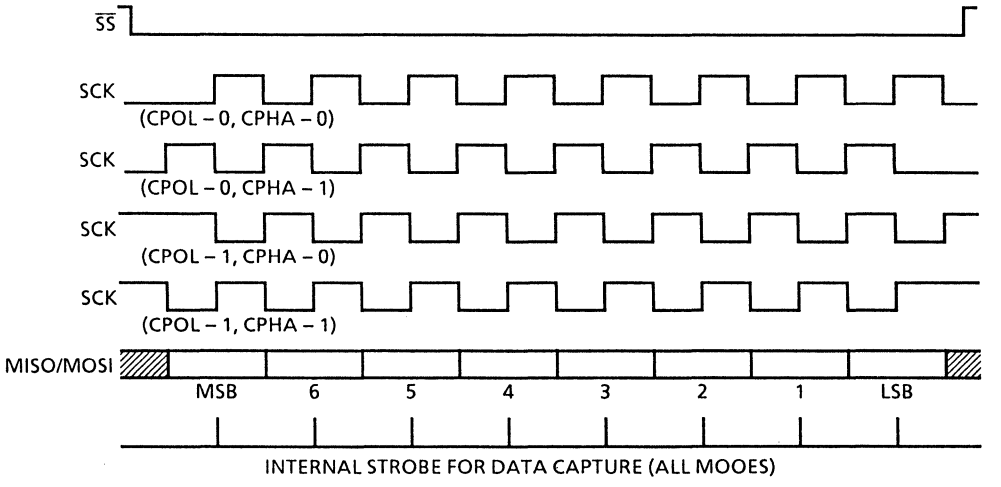


Figure 6.1 Data Clock Timing Diagram

When CPHA = 0, the shift clock is the OR of $\overline{SS}$ with SCK. In this clock phase mode, $\overline{SS}$ must go high between successive characters in an SPI message. When CPHA = 1, $\overline{SS}$ may be left low for several SPI characters. In cases where there is only one SPI slave MCU, its $\overline{SS}$ line could be tied to V_{SS} as long as CPHA = 1 clock modes are used.

6.3 FUNCTIONAL DESCRIPTION

Figure 6.2 shows a block diagram of the serial peripheral interface circuitry. When a master device transmits data to a slave device via the MISO line, the slave device responds by sending data to the master device via the master's MISO line. This implies full duplex transmission with both data out and data in synchronized with the same clock signal. 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) is used to signify that the I/O operation has been completed.

The SPI is double buffered on read, but not on write. If a write is performed during data transfer, the transfer occurs uninterrupted, and the write will be unsuccessful. This condition will cause the write collision (WCOL) status bit in the SPSR to be set. After a data byte is shifted, the SPIF flag of the SPSR is set.

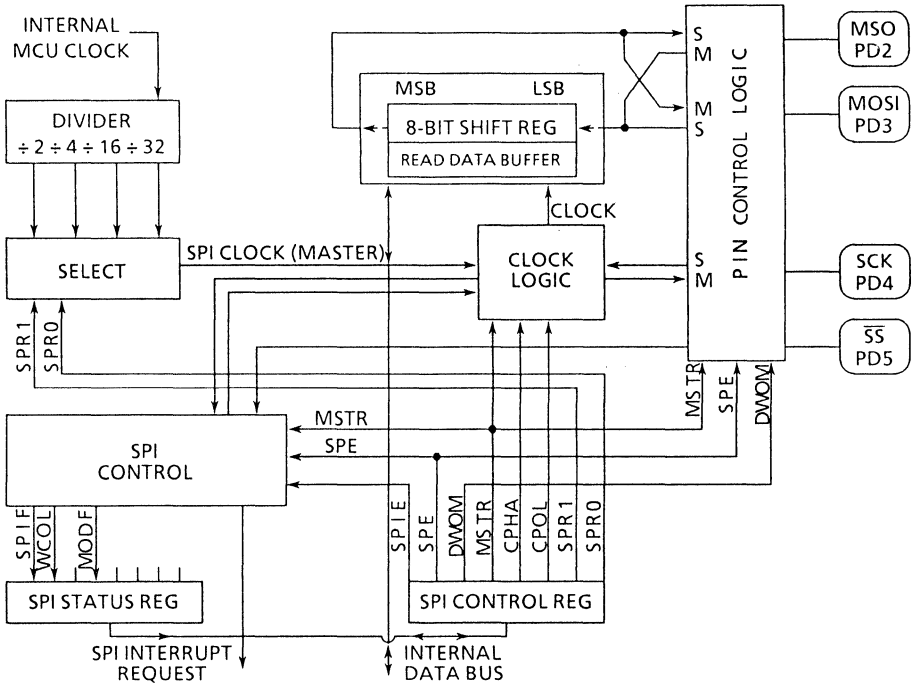


Figure 6.2 Serial Peripheral Interface Block Diagram

In the master mode, the SCK pin is an output. It idles high or low, depending on the CPOL bit in the SPCR, until data is written to the shift register, at which point eight clocks are generated to shift the eight bits of data and then SCK goes idle again.

In a slave mode, the slave start logic receives a logic low at the $\overline{SS}$ pin and a clock input at the SCK pin. Thus, the slave is synchronized with the master. Data from the master is received serially at the slave MOSI line and loads the 8-bit shift register. After the 8-bit shift register is loaded, its data is parallel transferred to the read buffer. During a write cycle, data is written into the shift register, then the slave waits for a clock train from the master to shift the data out on the slave's MISO line.

Figure 6.3 illustrates the MOSI, MISO, SCK and $\overline{SS}$ master-slave interconnections.

Due to data direction register control of SPI outputs and the port D write-OR mode (DWOM) option, the SPI system can be configured in a variety of ways. System with a single bidirectional data path rather than seperate MISO and MOSI paths can be accommodated. Since TMP68HC11E9 SPI slaves can selectively disable their MISO output, a broadcast message protocol is also possible.

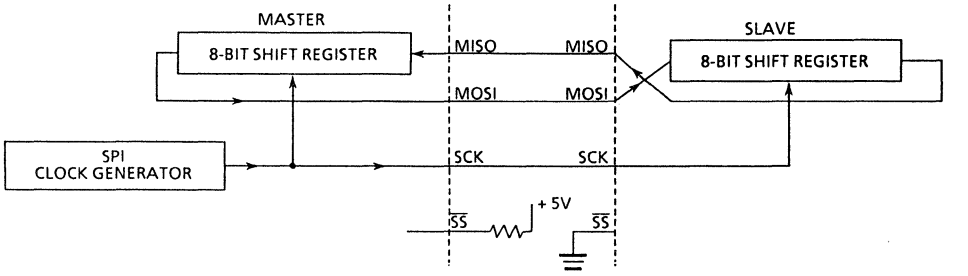


Figure 6.3 Serial Peripheral Interface Master-Slave Interconnection

6.4 SPI REGISTERS

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

6.4.1 Serial Peripheral Control Register (SPCR)

	7	6	5	4	3	2	1	0	
$1028	SPIE	SPE	DWOM	MSTR	CPOL	CPHA	SPR1	SPR0	SPCR
RESET	0	0	0	0	0	1	U	U	

SPIE - Serial Peripheral Interrupt Enable

0 = SPIF interrupts disabled

1 = SPI interrupt if SPIF = 1

SPE - Serial Peripheral System Enable

0 = SPI system off

1 = SPI system on

DWOM - Port D Write-OR Mode Option

DWOM affects all six port D pins together.

0 = Port D outputs are normal CMOS outputs

1 = Port D outputs act as open-drain outputs

MSTR - Master Mode Select

0 = Slave mode

1 = Master mode

CPOL - Clock Polarity

When the clock polarity (CPOL) bit is cleared and data is not being transferred, a steady state low value is produced at the SCK pin of the master device. Conversely, if this bit is set, the SCK pin will idle high. This bit is also used in conjunction with the clock phase control bit to produce the desired clock-data relationship between master and slave. See Figure 6.1.

CPHA - Clock Phase

The clock phase (CPHA) bit, in conjunction with the CPOL bit, controls the clock-data relationship between master and slave. The CPOL bit can be thought of as simply inserting an inverter in series with the SCK line. The CPHA bit selects one of two fundamentally different clocking protocols. When CPHA = 0, the shift clock is the OR of SCK with $\overline{SS}$. As soon as $\overline{SS}$ goes low the transaction begins and the first edge on SCK invokes the first data sample. When CPHA = 1, the $\overline{SS}$ pin may be thought of as a simple output enable control. Refer to Figure 6.1.

SPR1 and SPR0 - SPI Clock Rate Selects

These two serial peripheral rate bits (SPR1, SPR0) select one four baud rates (Table 6.1) to be used as SCK if the device is a master; however, they have no effect in the slave mode.

Table 6.1 Serial Periperal Rate Selection

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

6.4.2 Serial Peripheral Status Register (SPSR)

	7	6	5	4	3	2	1	0	
$1029	SPIF	WCOL	0	MODF	0	0	0	0	SPSR
RESET	0	0	0	0	0	0	0	0	

SPIF - SPI Transfer Complete Flag

The serial peripheral data transfer flag (SPIF) bit is set upon completion of data transfer between the processor and external device. If SPIF bit goes high, and if SPIE bit is set, a serial peripheral interrupt is generated. Clearing the SPIF bit is accomplished by reading the SPSR register (with SPIF bit set) followed by an access of the SPDR register. Unless SPSR is read (with SPIF bit set) first, attempts to write to SPDR register are inhibited.

WCOL - Write Collision

The write collision (WCOL) bit is set when an attempt is made to write to the serial peripheral data register while data transfer is taking place. If CPHA bit is zero a transfer is said to begin when $\overline{SS}$ goes low and the transfer ends when $\overline{SS}$ goes high after eight clock cycles on SCK. When CPHA bit is one a transfer is said to begin the first time SCK becomes active while $\overline{SS}$ is low and the transfer ends when the SPIF flag gets set. Clearing the WCOL bit is accomplished by reading the SPSR register (with WCOL bit set) followed by an access to SPDR register.

Bit 5 - Not Implemented

This bit always reads zero.

MODF - Mode Fault

The mode fault flag (MODF) bit indicates that there may 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 normally clear, and is set only when the master device has its $\overline{SS}$ pin pulled low. Setting the MODF bit affects the internal serial peripheral interface system in the following ways.

1) An SPI interrupt is generated if SPIE = 1.

2) The SPE bit is cleared. This disables the SPI.

3) The MSTR bit is cleared, thus forcing the device into the slave mode.

4) DDRD bits for the four SPI pins are forced to zeros.

Clearing the MODF bit is accomplished by reading the SPSR register (with MODF set), followed by a write to the SPCR register. Control bits SPE and MSTR may be restored by user software to their original state after the MODF bit has been cleared. It is also necessary to restore DDRD bit after a mode fault.

Bits 3-0 - Not Implemented

These bits always read zero.

6.4.3 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. At the completion of transmitting a byte of data, the SPIF status bit is set in both the master and slave devices.

When the user reads the serial peripheral data I/O register, a buffer is actually being read. The first SPIF bit 98must 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. In cases of overrun the byte which causes the overrun is lost.

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

7. ANALOG-TO-DIGITAL CONVERTER

The TMP68HC11E9 includes an 8-channel, multiplexed-input, successive approximation, analog-to-digital (A/D) converter with sample and hold to minimize conversion errors caused by rapidly changing input signals. Two dedicated lines (V_{RL}, V_{RH}) are provided for the reference voltage inputs. These pins may be connected to a separate or isolated power supply to ensure full accuracy of the A/D conversion. The 8-bit A/D converter has a total error of ± 1 LSB which includes $\pm 1/2$ LSB of quantization error and accepts analog inputs which range from V_{RL} to V_{RH}. Smaller analog input ranges can also be obtained by adjusting V_{RH} and V_{RL} to the desired upper and lower limits. Conversion is specified and tested for $V_{RL}=0V$ and $V_{RH}=5V\pm 10\%$;however, laboratory characterization over the full temperature range indicates little or no degradation with V_{RH}-V_{RL} as low as 2.5 to 3V. The A/D system can be operated with V_{RH} below V_{DD} and/or V_{RL} above V_{SS} as long as V_{RH} is above V_{RL} by enough to support the conversions (2.5 to 5.0V). Each conversion is accomplished in 32 MCU E clock cycles, provided the E clock rate is greater than 750 kHz. For systems which operate at clock rates less than 750 kHz, an internal R-C oscillator must be used to clock the A/D system. The internal R-C oscillator is selected by setting the CSEL bit in the OPTION register.

7.1 CONVERSION PROCESS

The A/D converter is ratiometric. An input voltage equal to V_{RL} converts to \$00 and an input voltage equal to V_{RH} converts to \$FF (full scale), with no overflow indication. For ratiometric conversions, the source of each analog input should use V_{RH} as the supply voltage and be referenced to V_{RL}.

Figure 7.1 shows the detailed sequence for a set of four conversions. This sequence begins one E clock cycle after a write to the A/D control/status register (ADCTL). Figure 7.2 shows a model of the port E A/D channel inputs. This model is useful for understanding the effects of external circuitry on the accuracy of A/D conversions.

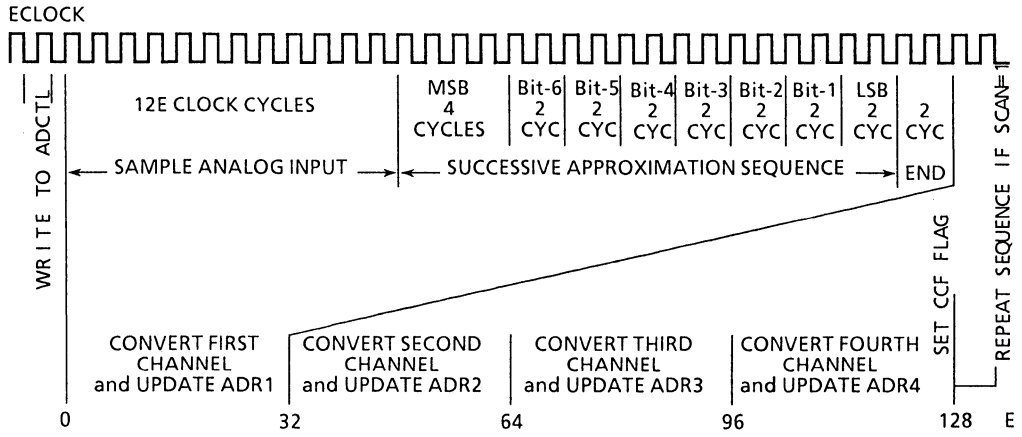
7.2 CHANNEL ASSIGNMENTS

A multiplexer allows the single A/D converter to select one of sixteen analog signals. Eight of these channels correspond to port E input lines to the MCU, four of the channels are for internal reference points or test functions, and four channels are reserved for future use. Table 7.1 shows the signals selected by the four channel control bits.

7.3 SINGLE-CHANNEL OPERATION

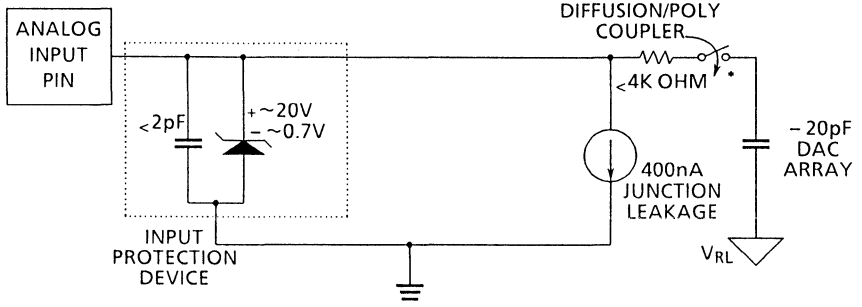
There are two variations of single-channel operation. In the first variation (SCAN = 0), the single selected channel is converted four consecutive times with the first result being stored in A/D result register 1 (ADR1) and the fourth result being stored in register ADR4. After the fourth conversion is complete, all conversion activity is halted

until a new conversion command is written to the ADCTL register. In the second variation (SCAN = 1), conversions continue to be performed on the selected channel with the fifth conversion being stored in register ADR1 (overwriting the first conversion result), the sixth conversion overwrites ADR2, and so on.



Notes: Conversion results are built up in the SAR and transferred into ADRx during the END period. The CCF status flag is set during the END period of the fourth conversion after a write to ADCTL. This figure assumes CSEL in the OPTION register is 0 so the E clock is acling as the conversion clock. If MULT=0 all four conversions in the sequence are performed on the same analog channel.

Figure 7.1 A/D Conversion Sequence



* This analog switch is closed only during the 12 cycle sample time

Figure 7.2 A/D Pin Model

7.4 MULTIPLE-CHANNEL OPERATION

There are two variations in multiple-channel operation. In the first variation (SCAN = 0), the selected group of four channels are converted, one time each, with the

first result being stored in register ADR1 and the fourth result being stored in register ADR4. After the fourth conversion is complete,all conversion activity is halted until a new conversion command is written to the ADCTL register. In the second variation (SCAN = 1), conversions continue to be performed on the selected group of channels with the fifth conversion being stored in register ADR1 (replacing the earlier conversion result for the first channel in the group), the sixth conversion overwrites ADR2, and so on.

Table 7.1 Analog-to-Digital Channel Assignments

CD	CC	CB	CA	Channel Signal	Result in ADRx if MULT = 1
0	0	0	0	AN0	ADR1
0	0	0	1	AN1	ADR2
0	0	1	0	AN2	ADR3
0	0	1	1	AN3	ADR4
0	1	0	0	AN4	ADR1
0	1	0	1	AN5	ADR2
0	1	1	0	AN6	ADR3
0	1	1	1	AN7	ADR4
1	0	0	0	Reserved	ADR1
1	0	0	1	Reserved	ADR2
1	0	1	0	Reserved	ADR3
1	0	1	1	Reserved	ADR4
1	1	0	0	V_{RH} Pin*	ADR1
1	1	0	1	V_{RL} Pin*	ADR2
1	1	1	0	$(V_{RH})/2$*	ADR3
1	1	1	1	Reserved*	ADR4

*This group of channels used during factory test.

7.5 OPERATION IN STOP AND WAIT MODES

If a conversion sequence is still in process when either the STOP or WAIT mode is enered, the conversion of the current channel is suspended. When the MCU resumes normal operation, that channel will be re-sampled and the conversion sequence resumed. As the MCU exits the WAIT mode, the A/D circuits are stable and valid results can be obtained on the first conversion. However, in STOP mode, all analog bias currents are disabled and it becomes necessary to allow a stabilization period when leaving the STOP mode. If the STOP mode is exited with a delay, there will be enough time for these circuits to stabilize before the first conversion. If the STOP mode is exited with no delay (DLY bit in OPTION register equal to zero), sufficient time must be allowed for the A/D circuitry to stabilize to avoid invalid results (see 7.8 A/D POWER UP AND CLOCK SELECT).

7.6 A/D CONTROL/STATUS REGISTER (ADCTL)

All bits in this register may be read or written, except bit 7 which is a ready-only status indicator and bit 6 which always reads as a zero.

	7	6	5	4	3	2	1	0	
$1030	CCF	0	SCAN	MULT	CD	CC	CB	CA	ADCTL
RESET	0	0	U	U	U	U	U	U	

CCF - Conversions Complete Flag

This read-only status indicator is set when all four A/D result registers contain valid conversion result. Each time the ADCTL register is written, this bit is automatically cleared to zero and a conversion sequence is started. In the continuous modes, conversions continue in a round-robin fashion and the result registers continue to be updated with current data even though the CCF bit remains set.

Note : The user must write to register ADCTL to initiate conversion. To abort a conversion in progress, write to the ADCTL register and a new conversion sequence is initiated immediately.

Bit 6 - Not Implemented

This bit always reads zero.

SCAN - Continuous Scan Control

When this control bit is clear, the four requested conversions are performed once to fill the four result registers. When this control bit is set, conversions continue in a round-robin fashion with the result registers being updated as data becomes available.

MULT - Multiple - Channel/Single Channel Control

When this bit is clear, the A/D system is configured to perform four consecutive conversions on the single channel specified by the four channel select bits CD through CA (bits 3-0 of the ADCTL register). When this bit is set, the A/D system is configured to perform a conversion on each of four channels where each result register corresponds to one channel.

Note : When the multiple channel continuous scan mode is used, extra care is needed in the design of circuitry driving the A/D inputs. Refer to the A/D Pin Model and A/D Conversion Sequence figures in addition to the following discussion. The charge on the capacitive DAC array prior to the sample time is related to the voltage on the previously converted channel. A charge share situation exists between the internal DAC capacitance and the external circuit capacitance. Although the amount of charge involved is small the rate at which it is repeated is every 64 microseconds for an E clock of 2 MHz. The RC

charging rate of the external circuit must be balanced against this charge sharing effect to avoid accuracy errors.

CD - Channel Select D

CC - Channel Select C

CB - Channel Select B

CA - Channel Select A

These four bits are used to select one of 16 A/D channels (see Table 7.1). When a multiple channel mode is selected (MULT=1), the two least-significant channel select bits (CB and CA) have no meaning and the CD and CC bits specify which group of four channels are to be converted. The channels selected by the four channel control bits are shown in Table 7.1.

7.7 A/D RESULT REGISTERS 1, 2, 3, AND 4 (ADR1, ADR2, ADR3, and ADR4)

The A/D result registers are ready-only registers used to hold an 8-bit conversion result. Writes to these registers have no effect. Data in the A/D result registers is valid when the CCF flag bit in the ADCTL register is set, indicating a conversion sequence is complete. If conversion results are needed sooner refer to Figure 7.1. For example the ADR1 result is valid 33 cycles after an ADCTL write. Refer to the A/D channelassignments in Table 7.1 for the relationship between the channel and the result registers.

7.8 A/D POWER UP AND CLOCK SELECT

A/D power up is controlled by bit 7 (ADPU) of the OPTION register. When ADPU is cleared, power to the A/D system is disabled. When ADPU is set, the A/D system is enabled. A delay of as much as 100 microseconds is required after turning on the A/D converter to allow the analog bias voltages to stabilize.

Clock select is controlled by bit 6 (CSEL) of the OPTION register. When CSEL is cleared, the A/D system uses the system E clock. When CSEL is set, the A/D system uses an internal R-C clock source, which runs at about 1.5 MHz. The MCU E clock is not suitable to drive the A/D system if it is operating below 750 kHz, in which case the R-C internal clock should be selected. A delay of 10 ms is required after changing CSEL from zero to one to allow the R-C oscillator to start and internal bias voltages to settle. Refer to 9.1.5 Configuration Options Register (OPTION) for additional information. Note thatthe CSEL control bit also enables a separate R-C oscillator to drive the EEPROM charge pump.

When the A/D system is operating with the MCU E clock, all switching and comparator operations are synchronized to the MCU clocks. This allows the comparator results to be sampled at quiet clock times to minimize noise errors. The internal R-C

oscillator is asynchronous to the MCU clock so noise will affect A/D results more while CSEL=1.

8. PROGRAMMABLE TIMER, REAL TIME INTERRUPT, AND PULSE ACCUMULATOR

This section describes the 16-bit programmable timer, the real time interrupt, and the pulse accumulator system.

8.1 PROGRAMMABLE TIMER

The timer has a single 16-bit free-running counter which is clocked by the output of a four-stage prescaler (divide by 1, 4, 8, or 16), which is in turn driven by the MCU E clock. Input functions are called input captures. These input captures record the count form the free-running counter in response to a detected edge on an input line. Output functions, called output compares, cause an output action when there is a match between a 16-bit output-compare register and the free-running counter. This timer system has three input capture registers and five output compare registers.

8.1.1 Counter

The key element in the timer system is a 16-bit free-running counter, or timer counter register. After reset, the MCU is configured to use the E clock as the input to the free-running counter. Initialization software may optionally reconfigure the system to use one of the three prescaler values. The prescaler control bits can only be written once during the first 64 cycles after a reset. Software can read the counter at any time without affecting its value because it is clocked and read during opposite phases of the E clock.

A counter read should first address the most significant byte. An MPU read of this address causes the least significant byte to be transferred to a buffer. This buffer is not affected by reset and is accessed when reading the least significant byte of the counter. For double byte read instructions, the two accesses occur on consecutive bus cycles.

The counter is cleared to $0000 during reset and is a read-only register with one exception. In test modes only, any MPU write to the most significant byte presets the counter to $FFF8 regardless of the value involved in the write.

When the count changes from $FFFF to $0000, the timer overflow flag (TOF) bit is set in timer interrupt flag register 2 (TFLG2). An interrupt can be enabled by setting the interrupt enable bit (TOI) in timer interrupt mask register 2 (TMSK2).

8.1.2 Input Capture

The input capture registers are 16-bit read-only registers which are not affected by reset and are used to latch the value of the counter when a defined transition is sensed by the corresponding input capture edge detector. The level transition which triggers counter transfer is defined by the corresponding input edge bits (EDGxB, EDGxA) in TCTL2.

The result obtained by an input capture corresponds to the value of the counter one E clock cycle after the transition which triggered the edge-detection logic. The selected edge transition sets the ICxF bit in timer interrupt flag register 1 (TFLG1) and can cause an interrupt if the corresponding ICxI bit(s) is (are) set in the timer interrupt mask register 1 (TMSK1). A read of the input capture register's most significant byte inhibits captures for one E cycle to allow a double-byte read of the full 16-bit register.

8.1.2.1 Input Capture4.

Port A pin 3 serves multiple functions. After reset, data direction bit 3 (DDRA3), in the PACTL register is cleared to zero configuring Port A pin 3 as an input. Port A pin 3 can then be used as an input capture4 (IC4), by setting I4/O5 to "one" in the PACTL register. The I4/O5 bit is configured to OC5 (cleared to zero) on reset. If DDRA3 is configured as an output (set to one) and IC4 is enabled, writes to Port A bit 3 causes edges on the PA3 pin to result in input captures. All other aspects of using IC4 remain the same as the other input captures, weith the exception that the 16-bit timer output compare 5 register now also server as the 16-bit timer input capture 4 register. When the TI4O5 register is acting as the IC4 capture register it cannot be written to. Upon reset, I4/O5 is configured as . The OC5 function overrides DDRA3 to force the Port A pin 3 to be an output whenever OM5:OL5 bit are not 0:0. In all other aspects, OC5 works the same as the other output compares.

8.1.3 Output Compare

All output compare registers are 16-bit read/write registers which are initialized to $FFFF by reset. They can be used as output waveform controls or as elapsed time indicators. If an output compare register is not used, it may be used as a storage location.

All output compare registers have a separate dedicated comparator for comparing against the free-running counter. If a match is found, the corresponding output compare flag (OCxF) bit in TFLG1 is set and a specified action is automatically taken. For output compare functions two through five the automatic action is controlled by pairs of bits (OMx and OLx) in the timer control register 1 (TCTL1). Each pair of control bits are encoded to specify the output action to be taken as a result of a successful OCx compare. The output action is taken on each succesful compare regardless of wheather or not the OCxF flag was previously clear.

An interrupt can also accompany a successful output compare, provided that the corresponding interrupt enable bit (OCx1) is set in TMSK1.

After an MPU write cycle to the most significant byte, output compares are inhibited for one E cycle in order to allow writing two consecutive bytes before making the next comparison. If both bytes of the register are to be changed, a double-byte write instruction should be used in order to take advantage of the compare inhibit feature.

MPU write can be made to either byte of the output compare register without affecting the other byte.

A write-only register, timer compare force (CFORC), allows forced mcompares. Five of the bit positions in the CFORC register correspond to the five output compares. To force a compare, or compares, a write is done to CFORC register withe the associated bits set for each output compare that is to be forced. The action taken as a result of a forced compare is the same as if there was a match between the OCx register and the free-running counter, except that the corresponding interrupt status flag bits are not set. Output actions are synchronized to the prescaled timer clock so there could be as much as 16 E clock cycles of delay between the write to CFORC and the output action.

8.1.4 Output Compare 1 I/O Pin Control

Unlike the other four output compares, output compare 1 can automatically affect any or all of the five output pins (bits 3-7) in port A as a result of a successful compare between the OC1 register and the 16-bit free-running counter. The two 5-bit registers used in conjunction with this function are the output compare 1 mask register (OC1M) and the output compare 1 data register (OC1D).

Register OC1M is used to specify the bits of port A (I/O and timer port) which are to be affected as a result of a successful OC1 compare. Register OC1D is used to specify the data which is to be stored to the affected bits of port A as the result of a successful OC1 compare. If an OC1 compare and another output compare occur during the same E cycle and both attempt to alter the same port A line, the OC1 compare prevails.

This function allows control of multiple I/O pins automatically with a single output compare.

Another intended use for the special I/O pin control on output compare 1 is to allow more than one output compare to control a single I/O pin. This allows pulses as short as one E clock cycle to be generated.

8.1.5 Timer Compare Force Register (CFORC)

The timer compare force register is used to force early output compare actions. The CFORC register is an 8-bit write-only register. Reads of this location have no meaning and always return logic zeros. Note that the compare force function is not generally recommended for use with the output toggle function because a normal compare occuring immediately before or after the force may result in undesirable operation.

	7	6	5	4	3	2	1	0	
$100B	FOC1	FOC2	FOC3	FOC4	FOC5	0	0	0	CFORC
RESET	0	0	0	0	0	0	0	0	

FOC1-FOC5 - Force Output Compare x Action

0= Has no meaning

1= Causes action programmed for output compare x, except the OCxF flag bit is not set.

Bits 2-0 - Not Implemented

These bits always read zero.

8.1.6 Output Compare 1 Mask Register (OC1M)

This register is used in conjunction with output compare 1 to specify the bits of port A which are affected as a result of a successful OC1 compare.

	7	6	5	4	3	2	1	0	
$100C	OC1M7	OC1M6	OC1M5	OC1M4	OC1M3	0	0	0	OC1M
RESET	0	0	0	0	0	0	0	0	

The bit of the OC1M register correspond bit-for-bit with the lines of port A (lines 7 through 3 only). For each bit that is affected by the successful compare, the corresponding bit in OC1M should be set to one.

Note that the pulse accumulator function shares line 7 of port A. If the DDRA7 bit in the pulse accumulator control register (PACTL) is set, then port A line 7 is configured as an output and OC1 can obtain access by setting OC1M bit 7. In this condition if the PAEN bit in the PACTL register is set, enabling the pulse accumulator input, then OC1 compares cause a write of OC1D bit 7 to an internal latch, and the output of that latch drives the pin and the pulse accumulator input. This action can then cause the pulse accumulator to take the appropriate action (pulse count or gate modes).

8.1.7 Output Compare 1 Data Register (OC1D)

This register used in conjunction with output compare 1 to specify the data which is to be stored to the affected bits of port A as the result of a successful OC1 compare.

	7	6	5	4	3	2	1	0	
$100D	OC1D7	OC1D6	OC1D5	OC1D4	OC1D3	0	0	0	OC1D
RESET	0	0	0	0	0	0	0	0	

The bits of the OC1D register correspond bit-for-bit with the lines of port A (lines 7 thru 3 only) . When a successful OC1 compare occurs, for each bit that is set in OC1M, the corresponding data bit in OC1D is stored in the corresponding bit of port A. If there

is a conflicting situation where an OC1 compare and another output compare function occur during the same E cycle with both attempting to alter the same port A line, the OC1 action prevails.

8.1.8 Timer Control Register 1 (TCTL1)

	7	6	5	4	3	2	1	0	
$1020	OM2	OL2	OM3	OL3	OM4	OL4	OM5	OL5	TCTL1
RESET	0	0	0	0	0	0	0	0	

OM2, OM3, OM4, and OM5 - Output Mode

OL2, OL3, OL4, and OL5 - Output Level

These two control bits (OMx and OLx) are encoded to specify the output action taken as a result of a successful OCx compare.

OMx	OLx	Action Taken Upon Successful Compare
0	0	Timer disconnected from output pin logic
0	1	Toggle OCx output line
1	0	Clear OCx output line to zero
1	1	Set OCx output line to one

8.1.9 Timer Control Register 2 (TCTL2)

	7	6	5	4	3	2	1	0	
$1021	EDG4B	EDG4B	EDG1B	EDG1A	EDG2B	EDG2A	EDG3B	EDG3A	TCTL2
RESET	0	0	0	0	0	0	0	0	

EDGxB and EDGxA - Input Caputre x Edge Control.

These two bits (EDGxB and EDGxA) are cleared to zero by reset and are encoded to configure the input sensing logic for input capture x as follow:

EDGxB	EDBxA	Configuration
0	0	Capture disabled
0	1	Capture on rising edges only
1	0	Capture on falling edges only
1	1	Capture on any (rising or falling)edge

8.1.10 Timer Interrupt Mask Register 1 (TMSK1)

	7	6	5	4	3	2	1	0	
$1022	OC1I	OC2I	OC3I	OC4I	OC5I	IC1I	IC2I	IC3I	TMSK1
RESET	0	0	0	0	0	0	0	0	

OCxI - Output compare x Interrupt

If the OCxI enable bit is set when the OCx flag bit is set, a hardware interrupt sequence is requested.

ICxI - Input Capture x Interrupt

If the ICxI enable bit is set when the ICxF flag bit is set, a hardware interrupt sequence is requested.

Note : When the I4/O5 bit in the PACTL register is one, the I405I bit behaves as the Input Capture 4 Interrupt bit, and when I4/O5 is zero, the I405I bit acts as the Output Compare 5 Interrupt control bit.

8.1.11 Timer Interrupt Flag Register 1 (TFLG1)

Timer interrupt flag register 1 is used to indicate the occurrence of timer system events, and together with the TMSK1 register allows the timer subsystem to oprate in a polled or interrupt driven system. For each bit in TFLG1, there is a corresponding bit in TMSK1 in the same bit position. If the mask bit is set, each time the condition for the corresponding flag are met, a hardware interrupt sequence is requested as well as the flg bit being set.

These timer system status flags are cleared by writing a one to the bit positions corresponding to the flag(s) which are to be cleared. Bit manipulation instructions would be inappropriate for flag clearing because they are read-modify-write instruction. Even though the instruction mask implies that the programmer is only interested in some of the bits in the manipulated location, the entire location is actually read and rewritten which may clear other bits in the register.

	7	6	5	4	3	2	1	0	
$1023	OC1F	OC2F	OC3F	OC4F	I4O5F	IC1F	IC2F	IC3F	TFLG1
RESET	0	0	0	0	0	0	0	0	

OCxF - Output Compare xFlag

This flag bit is set each time the timer counter matches the output compare register x value. A write of a zero does not affect this bit. A write of a one causes this bit to be cleared.

ICxF - Input Caputure x Flag

This flag is set each time a selected active edge is detected on the ICx input line. A write of a zero does not affect this bit. A write of a one causes this bit to be cleared.

Note : When the I4/O5 bit in the PACTL register is one, the I405F bit behaves as the Input Capyure 4 Flag bit, and when I4/O5 is zero, the I4O5I bit acts as the Output Compare 5 Flag.

8.1.12 Timer Interrupt Mask Register 2 (TMSK2)

Timer interrupt mask register 2 is used to control whether or not a hardware interrupt sequence is requested as a result of a status bit being set in timer interrupt flag register 2. In addition, two timer prescaler bits are included in this register. For each of the four most significant bits in timer flag register 2, (TFLG2), there is a correponding bit in the timer mask register 2 (TMSK2) in the same bit position.

	7	6	5	4	3	2	1	0	
$1024	TOI	RTII	PAOVI	PAII	0	0	PR1	PR0	TMSK2
RESET	0	0	0	0	0	0	0	0	

TOI - Timer Overflow Interrupt Enable

0 = TOF interrupts disabled

1 = Interrupt requested when TOF = 1

RTII - RTI Interrupt Enable

0 = RTIF interrupts disabled

1 = Interrupt requested when RTIF = 1

PAOVI - Pulse Accumulator Overflow Interrupt Enable

0 = PAOVF interrupts disabled

1 = Interrupt requested when PAOVF = 1

PAII - Pulse Accumulator Input Interrupt Enable

0 = PAIF interrupts disabled

1 = Interrupt requested when PAIF = 1

Bits 3 and 2 - Not Implemented

These bits always read zero.

PR1 and PRO - Timer Prescaler Selects

These two bits may be read at any time but may only be written during initalization. Writes are disabled after the first write or after 64 E cycles out of reset.

If the MCU is in special test or special bootstrap mode, then these two bits may be written any time.

Bits3 and 2 - Not Implemented

These bits always read zero.

PR1 and PRO - Timer Prescaler Selects

These two bits may be read at any time but may only be written during initalization. Writes are disabled after the first write or after 64 E cycles out of reset. If the MCU is in special test or special bootstrap mode, then these two bits may be written any time.

These two bits specify the timer prescaler divide factor.

PR1	PR0	Divide-by-Factor
0	0	1
0	1	4
1	0	8
1	1	16

8.1.13 Timer Interrupt Flag Register 2 (TFLG2)

Timer interrupt flag register 2 is used to indicate the occurrence of timer system events, and together with the TMSK2 register, allows the timer subsystems to operate in a polled or interrupt driven system. For each bit in timer flag register 2 (TFLG2), there is a corresponding bit in timer mask register 2 (TMSK2) in the same bit position. If the enable bit is set each time the conditions for the corresponding flag are met, a hardware interrupt sequence is requested as well as the flag bit being set.

The timer system status flags are cleared by writing a one to the bit positions corresponding to the flag(s) which are to be cleared. Bit manipulation instructions would be inappropriate for flag clearing because they are read-modify-write instructions. Even though the instruction mask implies that the programmer is only interested in some of the bits in the manipulated location, the entire location is actually read and rewritten which may clear other bits in the register.

	7	6	5	4	3	2	1	0	
$1025	TOF	RTIF	PAOVF	PAIF	0	0	0	0	TFLG2
RESET	0	0	0	0	0	0	0	0	

TOF - Timer Overflow

This bit is cleared by reset. It is set to one each time the 16-bit free-running counter advances from a value of $FFFF to $0000. This bit is cleared by a write to the TFLG2 register with bit 7 set.

RTIF - Real Time Interrupt Flag

This bit is set at each rising edge of the selected tap point. This bit is cleared by a write to the TFLG2 register with bit 6 set.

PAOVF - Pulse Accumulator Overflow Interrupt Flag

This bit is set when the count in the pulse accumulator rolls over from $FF to $00. This bit is cleared by a write to the TFLG2 register with bit 5 set.

PAIF - Pulse Accumulator Input Edge Interrupt Flag

This bit is set when an active edge is detected on the PAI input pin. This bit is cleared by a write to the TFLG2 register with bit 4 set.

Bits 3-0 - Not Implemented

These bits always read zero.

8.2 REAL TIME INTERRUPT

The real time interrupt feature on the MCU is configured and controlled by using two bits (RTR1 and RTR0) in the PACTL register to select one of four interrupt rates. The RTII bit in the TMSK2 register enables the interrupt capability. Every timeout causes the RTIF bit in TFLG2 to be set, and if RTII is set, and interrupt request is generated. After reset, one entire real time interrupt period elapses before the RTIF flag is set for the first time.

8.3 PULSE ACCUMULATOR

The pulse accumulator is an 8-bit read/write counter which can operate in either of two modes (external event counting or gated time accumulation) depending on the state of the PAMOD control bit in the PACTL register. In the event counting mode, the 8-bit counter is clocked to increasing values by an external pin. The maximum clocking rate for the external event counting mode is E clock divided by two. In the gated time accumulation mode, a free-running E clock/64 signal drives the 8-bit counter, but only while the external PAI input pin is activated.

The pulse accumulator uses port A bit 7 as its PAI input, but this pin also shares function as a general purpose I/O pin and as a timer output compare pin. Normally port A bit 7 would be configure as a input when being used for the pulse accumulator. Note that even when port A bit 7 is configured for output, this pin still drives the input to the pulse accumulator.

8.4 PULSE ACCUMULATOR CONTROL REGISTER (PACTL)

Four bits in this register are used to control an 8-bit pulse accumulator system and two other bits are used to select the rate for the real time interrupt system.

	7	6	5	4	3	2	1	0	
$1026	DDRA7	PAEN	PAMOD	PEDGE	DDRA3	I4/O5	RTR1	RTR0	PACTL
RESET	0	0	0	0	0	0	0	0	

DDRA7 - Data Direction for Port A Bit 7

0 = Input only

1 = Output

PAEN - Pulse Accumulator System Enable

0 = Pulse accumulator off

1 = Pulse accumulator on

PAMOD - Pulse Accumulator Mode

0 = External event counting

1 = Gated time accumulation

PEDGE - Pulse Accumulator Edge Control

This bit has different meanings depending on the state of the PAMOD bit.

PAMOD	PEDGE	Action on Clock
0	0	PAI Falling Edge Increments the Counter
0	1	PAI Rising Edge Increments the Counter
1	0	A '0' on PAI Inhibits Counting
1	1	A '1' on PAI Inhibits Counting

DRRA3 - Data Direction for Port A Bit 3

0 = Input only

1 = Output

I4/O5 - Input Capture 4/Output Compare5

0 = Output compare 5 function enable (No IC4)

1 = Input capture4 function enable (No OC5)

These bits always read zero.

RTR1 and RTR0 - RTI Interrupt Rate Selects

These two bits select one of four rates for the real rime periodic interrupt circuit (see Table 8.1). Reset clears these two bits and after reset, a full RTI period elapses before the first RTI interrupt.

Table 8.1 Real Time Interrupt Rate versus RTR1 and RTR0

RTR1	RTR0	Divide E By	XTAL = 2^{23}	XTAL = 8.0 MHz	XTAL = 4.9152 MHz	XTAL = 4.0 MHz	XTAL = 3.6864 MHz
0	0	2^{13}	3.91 ms	4.10 ms	6.67 ms	8.19 ms	8.89 ms
0	1	2^{14}	7.81 ms	8.19 ms	13.33 ms	16.38 ms	17.78 ms
1	0	2^{15}	15.62 ms	16.38 ms	26.67 ms	32.77 ms	35.56 ms
1	1	2^{16}	31.25 ms	32.77 ms	53.33 ms	65.54 ms	71.11 ms
		E =	2.1 MHz	2.0 MHz	1.2288 MHz	1.0 MHz	921.6 MHz

9. RESETS, INTERRUPTS, AND LOW POWER MODES

This section provides a description of the resets, interrupts, and low power modes. The computer operating properly (COP) watchdog system and clock monitor are described as part of the reset system. The interrupt description includes a flowchart to illustrate how interrupts are executed.

9.1 RESETS

The MCU has four possible types of reset:an active low external reset pin ($\overline{\text{RESET}}$), a power-on reset, a computer operating properly (COP) watchdog timer reset, and a clock monitor reset.

9.1.1 External $\overline{\text{RESET}}$ Pin

The $\overline{\text{RESET}}$ pin is used to reset the MCU and allow an orderly software startup procedure. When a reset condition is sensed, this pin is driven low by an internal device for four E clock cycles, then released, and two E clock, cycles later it is sampled. If the pin is still low, it means that an external reset has occurred. If the pin is high, it implies that the reset was initiated internally by either the watchdog timer (COP) or the clock monitor (refer to Figure 9.1). This method of differentiation between internal and external reset conditions assumes that the reset pin will rise to a logic one in less than two E clock cycles once it is released and that an externally generated reset should stay active for at least eight E clock cycles.

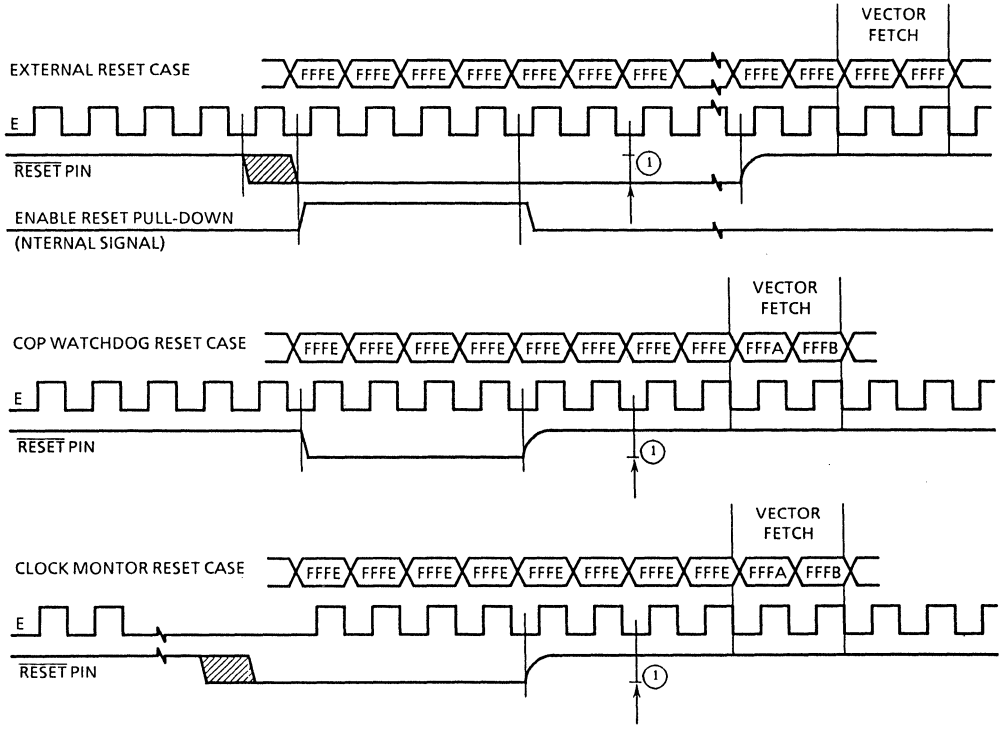
Since there is EEPROM on chip, it is very important to control reset during power transitions. If the reset line is not held low while V_{DD} is below its minimum operating level, the EEPROM contents could be corrupted. Corruption occurs due to improper instruction execution when there is not sufficient voltage to execute instructions correctly. Both EEPROM memories and the EEPROM based CONFIG register are subject to this potential problem.

A low voltage inhibit (LVI) circuit which holds reset low whenever V_{DD} is below its minimum operating level is required to protect against EEPROM corruption. Figures 9.2 and 9.3 show two examples of reset circuits with LVI capabilities. The best circuit for a particular application may be different from either of these suggested circuits.

The circuit in Figure 9.2 includes an R-C turn on delay. In older dynamic MCU designs this delay was required to allow the crystal oscillator to stabilize. Since the TMP68HC11E9 is a fully static CMOS design, it is capable of operating at clock rates down to DC and therefore a turn on delay is not required. Though not required for proper MCU operation, a turn on delay could be dictated by the system requirements of a particular application. For example, if incorrect time period measurements during the first few tens of milliseconds of operation cannot be tolerated, a turn on delay is probably needed.

9.1.2 Power-On Reset

The power-on reset occurs when a positive transition is detected on V_{DD}. The power-on reset is used strictly for power turn-on conditions and should not be used to detect any drops in power supply voltage. The power-on circuitry provides a 4064 cycle time delay from the time of the first oscillator operation. In a system where E = 2 MHz, power on reset lasts about 2 milliseconds. If the external $\overline{\text{RESET}}$ pin is low at the end of the power-on delay time, the MCU remains in the reset condition until the $\overline{\text{RESET}}$ pin goes high.



Note (1) $\overline{\text{RESET}}$ pin is sampled at this time. Low implies an extemal reset. High implies clock monitor or COP watchdog caused reset.

Figure 9.1 Reset Timing

9.1.2.1 CPU.

After reset, the CPU fetches the restart vector from locations $FFFE and $FFFF($BFFE and $BFFF if in special bootstrap or special test operating mode) during the first three cycles, and begins executing instructions. The stack pointer and other CPU registers are indeterminate immediately after reset; however, the X and I

interrupt mask bits in the condition code register are set to mask any interrupt requests. Also, the S bit in the condition code register is set to disable the STOP mode.

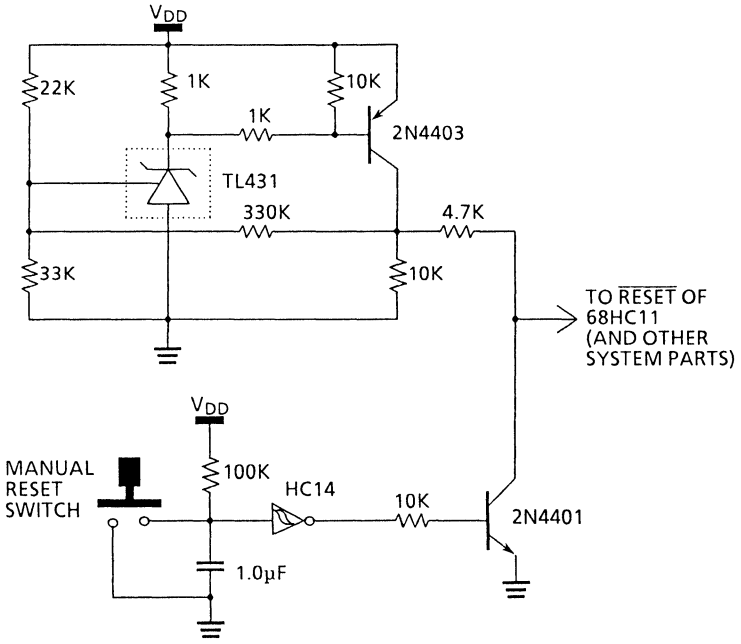


Figure 9.2 Reset Circuit with LVI and RC Delay

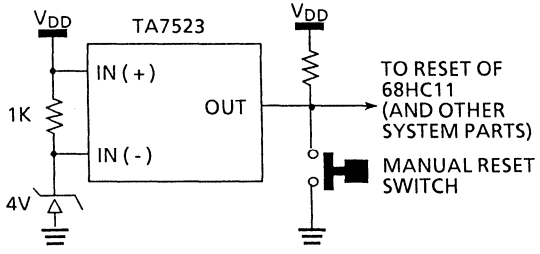


Figure 9.3 Simple LVI Reset Circuit

9.1.2.2 Memory Map.

After reset, the INIT register is initialized to $01, putting the 512 bytes of RAM at locations $0000 through $01FF and the control registers at locations $1000 through $103F. The 12K-bite ROM and/or the 512-byte EEPROM may or may not be present in the memory map because the two bits that enable them in the CONFIG register are EEPROM cells and are not affected by reset or power down.

9.1.2.3 Parallel I/O.

When a reset occurs in expanded multiplexed operating mode, the 18 pins used for parallel I/O are dedicated to the expansion bus. If a reset occurs in the single-chip operating mode, the STAF, STAI, and HNDS bits in the parallel input/output control register (PIOC) are cleared so that no interrupt is pending or enabled, and the simple strobed mode (rather than full handshake mode) of parallel I/O is selected. The CWON bit in PIOC is cleared so port C is not in wired-OR mode. Port C is initialized as an input port (DDRC = $00), port B is general purpose output port with all bits cared. STRA is the edge-sensitive strobe A input and the active edge is initially configured to detect rising edges (EGA bit in the PIOC set), and SARB is the strobe B output and is initially a logic zero (the INVB bit in the PIOC is set). Port C, port D bits 0 through 5, port A bits 0, 1, 2, 3, and 7, and port E are configured as general purpose high-impedance inputs. Port B and bits 4 through 6 of port A have their directions fixed as outputs and their reset state is a logic zero.

9.1.2.4 Timer.

During reset, the timer system is initialized to a count of $0000. The prescaler bits are cleared, and all output compare registers are initialized to $FFFF. All input capture registers are indeterminate after reset. The output compare 1 mask (OC1M) register is cleared so that successful OC1 compares do not affect any I/O pins. The other four output compares are configured to not affect any I/O pins on successful compares. All three input capture edge-detector circuits are configured for "capture disabled" operation. The timer overflow interrupt flag and all eight timer function interrupt flags are cleared. All nine timer interrupts are disabled since their mask bits are cleared.

9.1.2.5 Real Time Interrupt.

The real time interrupt flag is cleared and automatic handware interrupts are masked. The rate control bits are cleared after reset and may be initialized by software before the real time interrupt system is used.

9.1.2.6 Pulse Accumulator.

The pulse accumulator system is disabled at reset so that the PAI input pin defaults to being a general purpose input pin.

9.1.2.7 COP.

The COP watchdog system is enabled if the NOCOP control bit in the system configuration control register (EEPROM cell) is clear, and disabled if NOCOP is set. The COP rate is set for the shortest duration timeout.

9.1.2.8 SCI Serial I/O.

The reset condition of the SCI system is independent of the operating mode. At reset, the SCI baud rate is indeterminate and must be established by a software write to the BAUD register. All transmit and receive interrupts area masked and both the transmitter and receiver are disabled so the port pins default to being general purpose I/O lines. The SCI frame format is initialized to an 8-bit character size. The send break and receiver wake up functions are disabled. The TDRE and TC status bits in the SCI status register are both set, indicating that there is no transmit data in either the transmit data register of the transmit serial shift register. The RDRF, IDLE, OR NF, and FE receive-related status bits are all cleared.

Note that upon reset in special bootstrap mode execution begins in the 192 byte boot ROM. This firmware sets port D to wire-OR mode, establishes a baud rate, and enables the SCI receiver and transmitter.

9.1.2.9 SPI Serial I/O.

The SPI system is desabled by reset. The port pins associated with this function default to being general purpose I/O lines.

9.1.2.10 A/D Converter.

The A/D converter system configuration is indeterminate after reset. The conversion complete flag is cleared by reset. The ADPU bit is cleared by reet thus disabling the A/D system.

9.1.2.11 System.

The EEPROM programming controls are all disabled so the memory system is configured for normal read operation. The highest priority I interrupt defaults to being the external $\overline{IRQ}$ pin by PSEL3-PSEL0 equal to 0:1:0:1. The $\overline{IRQ}$ interrupt pin is configured for level sensitive operation (for wire-OR systems). The RBOOT, SMOD, and MDA bits in the HPRIO register reflect the status of the MODB and MODA inputs at the rising edge of reset. The DLY control bit is set to specify that an oscillator start-up delay is imposed upon recovery from STOP mode. The clock monitor system is disabled by CME equal zero.

9.1.3 Computer Operating Properly (COP) Reset

The MCU includes a computer operating properly watchdog system to help protect against software failures. To use a COP watchdog timer, a watchdog timer reset sequence must be executed on a regular periodic basis so that the watchdog timer is never allowed to time out.

The internal COP function includes special control bits which permit specification of one of four time out periods and even allows the function to be disabled completely. The COP system has a separate reset vector.

The NOCOP control bit, which determines whether or not a wathcdog timeout causes a system reset, is implemented in an EEPROM cell in the CONFIG register. Once programmed, this bit remains set (or cleared) even when no power is applied, and the COP function is enabled or disabled independent of resident software. The NOCOP control bit may be preempted while in special test modes to prevent the COP system from causing a hardware reset.

Two other control bits in the OPTION register select one of four timeout durations for the COP timer. The actual timeout period is dependent on the system E clock frequency, but for reference purposes, Table 9.1 shows the relationship between the CR1 and CR0 control bits and the COP timeout period for various system clock frequencies.

Table 9.1 COP Timeout Period versus CR1 and CR0

CR1	CR0	E/2^{15} Divided By	XTAL = 2^{23} Timeout – 0/ + 15.6ms	XTAL = 8.0 MHz Timeout – 0/ + 16.4ms	XTAL = 4.9152 MHz Timeout – 0/ + 26.7ms	XTAL = 4.0 MHz Timeout – 0/ + 32.8ms	XTAL = 3.6864 MHz Timeout – 0/ + 35.6ms
0	0	1	15.625 ms	16.384 ms	26.667 ms	32.768 ms	35.556 ms
0	1	4	62.5 ms	65.536 ms	106.67 ms	131.07 ms	142.22 ms
1	0	16	250 ms	262.14 ms	426.67 ms	524.29 ms	568.89 ms
1	1	64	1 s	1.049 s	1.707 s	2.1 s	2.276 s
		E =	2.1MHz	2.0MHz	1.2288MHz	1.0MHz	921.6kHz

The default reset condition of the CR1 and CR0 bits is cleared which corresponds to the shortest timeout period.

The sequence required to reset the watchdog timer is:

1) Write $55 to the COP reset register (COPRST) as $103A, followed by

2) Write $AA to the same address.

Both writes must occur in correct order prior to timeout but, any number of instructions may be executed between the writes. The elapsed time between adjacent software reset sequence must never be greater than the COP time out period. Reading the COPRST register does not return meaningful data and does not affect the watchdog timer.

9.1.4 Clock Monitor Reset

The clock monitor function is enabled by the CME control bit in the OPTION register. When CME bit is clear, the monitor function is disabled. When the CME bit is set, the clock monitor function detects the absence of an E clock for more than a certain period of time. The timeout period is dependent on processing parameters and will be between 5

and 100 microseconds. This means that an E-clock rate of 200 kHz or more will never cause a clock monitor failure and an E-clock rate of 10 kHz or less will definitely cause a clock

monitor failure. This implies that systems operating near or below an E-clock rate of 200 kHz should not use the clock monitor function.

Upon detection of a slow or absent clock, the clock monitor circuit will cause a system reset. This reset is issued to the external system via the bidirectional $\overline{\text{RESET}}$ pin. The clock monitor system has a separate reset vector.

Special considerations are needed when using a STOP function and clock monitor in the same system. Since the STOP function causes the clocks to be halted, the clock monitor function will generate a reset sequence if it is enabled at the time the STOP mode is entered.

The clock monitor is useful as a backup for the COP watchdog timer. Since the watchdog timer requires a clock to function, it will not indicate any failure if the system clocks fail. The clock monitor would detect such a failure and force the MCU to its reset state. Note that clocks are not required for the MCU to reach its reset configuration, although clocks are required to sequence through reset back to the run condition.

9.1.5 Configuration Options Register (OPTION)

This is a special purpose 8-bit register that is used (optionally) during initialization to configure internal system configuration options. With the exception of bits 7, 6, and 3 (ADPU, CSEL, and CME) which may be read or written at any time, this register may be written to only once after a reset and thereafter is a read-only register. If no write is performed to this location within 64 E-clock cycles after reset, then bits 5, 4, 1, and 0 (IRQE, DLY, CR1, and CR0) will become read-only to minimize the possibility of any accidential changes to the system configuration (writes will be ignored). While in special test modes, the protection mechanism on this register is preempted and all bits in the OPTION register may be written repeatedly.

	7	6	5	4	3	2	1	0	
$1039	ADPU	CSEL	IRQE	DLY	CME	0	CR1	CR0	OPTION
RESET	0	0	0	1	0	0	0	0	

ADPU - A/D Powerup

This bit controls operations of the on-chip analog-to-digital converter. When ADPU bit is clear, the A/D system is powered down and conversion requests will not return meaningful information. To use the A/D system, this bit should be set. A 100 microsecond delay is required after ADPU bit is turned on to allow the A/D system to stabilize.

CSEL - A/D/EE Charge Pump Clock Source Select

This bit determines the clocking source for the on-chip A/D and EEPROM charge pump. When this bit is zero, the MCU E clock drives the A/D system and the EEPROM charge pump. When CSEL bit is one, on-chip separate R-C oscillators are enabled and clock the systems at about 2 MHz. When running with an E clock below 1 MHz, CSEL bit must be high to program or erase EEPROM. When operating below 750 kHz E clock rate, CSEL bit should be high for A/D conversions. A delay of 10 milliseconds is required after CSEL bit is turned on to allow the A/D system to stabilize.

IRQE - IRQ Edge/Level Sensitive

This bit may only be written under special circumstances as described above. When this bit is clear, the $\overline{IRQ}$ pin is configuredfor level sensitive wired-OR operation (low level) and when it is set, the $\overline{IRQ}$ pin is configured for edge-only sensitivity (falling edges).

DLY - STOP Exit Turn-On Delay

This bit may only be written under special circumstances as described above. This bit is set during reset and controls whether or not a relatively long turn-on delay will be imposed before processing can resume after a STOP period. If an external clock source is supplied this delay can be inhibited so that processing can resume within a few cycles of a wake up from STOP mode. When DLY bit is set, a 4064 E clock cycle delay is imposed to allow oscillator stabilization and when DLY bit is clear, this delay is bypassed.

CME - Clock Monitor Enable

This control bit may be read or written at any time and controls whether or not the internal clock monitor circuit will trigger a reset sequence when a slow or absent system clock is detected. When it is clear, the clock monitor circuit is disabled and when it is set, the clock monitor circuit is enabled. Systems operating at or below 200 kHz should not use the clock monitor function. Reset clears the CME bit.

Bit 2 - Not Implemented

This bit always reads zero.

CR1 and CR0 - COP Timer Rate Selects

These bits may only be written under special circumstances as described above. Refer to Table 9.1 for the relationship between CR1:CR0 and the COP timeout period.

9.2 INTERRUPTS

When an external or internal (hardware) interrupt occurs, the interrupt is not serviced until the current instruction being executed is completed. Until the current instruction is complete, the interrupt is considered pending. After completion of current instruction execution, unmasked interrupts may be serviced in accordance with an

established fixed hardware priority circuit;however, one I-bit related interrupt source may be dynamically elevated to the highest I bit priority position in the heirarchy (see 9.2.5 Highest Priority I Interrupt Register (HPRIO)).

Seventeen hardware interrupts and one software interrupt (excluding reset type interrupts) can be generated from all of the possible sources. The interrupts can be divided into two basic categories, maskable and non-maskable. In the TMP68HC11E9 fifteen of the interrupts can be masked using the condition code register I bit. In addition to being maskable by the I bit in the condition code register, all of the on-chip interrupt sources are individually maskable by local control bits.

The software interrupt (SWI instruction) is a non-maskable instruction rather than a maskable interrupt source. The illegal opcode interrupt is a non-maskable interrupt. The last interrupt source, external input to the $\overline{\text{XIRQ}}$ pin, is considered a non-maskable interrupt because once enabled, it cannot be masked by software;however, it is masked during reset and upon receipt of an interrupt at the $\overline{\text{XIRQ}}$ pin. Tables 9.2, 9.3, and 9.4 provide a list of each interrupt, its vector location in memory, and the actual condition code and control bits that mask it. A discussion of the various interrupts is provided below. Figure 9.4 shows the interrupt stacking order.

Table 9.2 Interrupt Vector Assignments

Vector Address	Interrupt Source	CC Register Mask	Local Mask
FFC0, C1	Reserved	–	–
•	•		
•	•		
FFD4, D5	Reserved	–	–
FFD6, D7	SCI Serial System	I Bit	See Table 9.3
FFD8, D9	SPI Serial Transfer Complete	I Bit	SPIE
FFDA, DB	Pulse Accumulator Input Edge	I Bit	PAII
FFDC, DD	Pulse Accumulator Overflow	I Bit	PAOVI
FFDE, DF	Timer Overflow	I Bit	TOI
FFE0, E1	Timer Output Compare 5	I Bit	OC5I
FFE2, E3	Timer Output Compare 4	I Bit	OC4I
FFE4, E5	Timer Output Compare 3	I Bit	OC3I
FFE6, E7	Timer Output Compare 2	I Bit	OC2I
FFE8, E9	Timer Output Compare 1	I Bit	OC1I
FFEA, EB	Timer Input Compare 3	I Bit	OC3I
FFEC, ED	Timer Input Compare 2	I Bit	OC2I
FFEE, EF	Timer Input Compare 1	I Bit	OC1I
FFF0, F1	Real Timer Interrupt	I Bit	RTII
FFF2, F3	$\overline{IRQ}$ (External Pin or Parallel I/O)	I Bit	See Table 9.4
FFF4, F5	$\overline{XIRQ}$ Pin (Pesudo Non-Maskable Interrupt)	X Bit	None
FFF6, F7	SWI	None	None
FFF8, F9	Illegal Opcode Trap	None	None
FFFA, FB	COP Failure (Reset)	None	NOCOP
FFFC, FD	COP Clock Monitor Fail (Reset)	None	CME
FFFE, FF	$\overline{RESET}$	None	None

Table 9.3 SCI Serial System Interrupts

Interrupt Cause	Local Mask
Receive Data Register Full	RIE
Receiver Overrun	RIE
Idle Detect	ILIE
Transmit Data Register Empty	TIE
Transmit Complete	TCIE

Table 9.4 IRQ Vector Interrupts

Interrupt Cause	Local Mask
External Pin	None
Parallel I/O Handshake	STAI

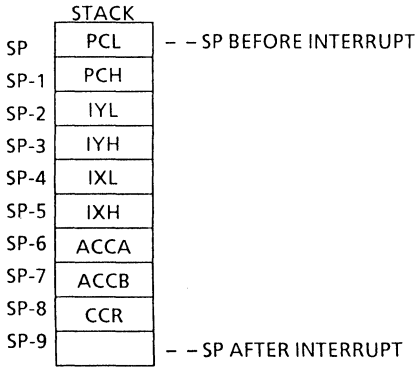


Figure 9.4 Interrupt Stacking Order

9.2.1 Software Interrupt (SWI)

The software interrupt is executed in the same manner as any other instruction and will take precedence over interrupts only if the orher interrupts are masked (I and X bits in the condition code register set). The SWI instruction is executed in a manner simslar to other maskable interrupts in that it sets the 1bit, CPU registers are stacked, etc.

Note : The SWI instruction will not be fetched if another interrupt os pending. However, once an SWI instruction has begun, no other interrupt can be honored until the SWI vector has been fetched.

9.2.2 Illegal Opcode Trap

Since not all possible opcodes or opcode sequences are defined, an illegal opcode detection circuit has been included. When an illegalopcode is detected, an interrupt is requested to the illegal opcode vector. The illegal opcode vector should never be left unitialized. It is a good idea to reinitialize the stack pointer as a result of an illegal opcode interrupt so repeated execution of illegal opcodes does not cause stack overruns.

9.2.3 Interrupt Mask Bits in Condition Code Register

Upon reset, both the X bit and the I bit are set to inhibit all maskable interrupts and $\overline{\text{XIRQ}}$. After minimum system initialization, software may clear the X bit by a TAP instruction, thus enabling $\overline{\text{XIRQ}}$ interrupts. Thereafter software cannot set the X bit so an $\overline{\text{XIRQ}}$ interrupt is effectively a nonmaskable interrupt. Since the operation of the I bit related interrupt structure has no effect on the X bit, the external $\overline{\text{XIRQ}}$ pin remains effectively non-masked. In the interrupt priority logic, the $\overline{\text{XIRQ}}$ interrupt is a higher priority than any source that is maskable by the 1bit. All I bit reated interrupts operate normally with their own priority relationship. When an I bit related interrupt occurs, the I bit is automatically set by hardware after stacking the condition code register byte, but the X bit is not affected.

When an X bit related interrupt occurs, both the X bit and the I bit are automatically set by hardware after stacking the condition code register. An RTI (return from interrupt) instruction restores the X and I bits to their pre-interrupt request state.

9.2.4 Priority Structure

Interrupts obey a fixed hardware priority circuit to resolve simultaneous requests; however, one 1bit related interrupt source may be elevated to the highest 1bit priority position in the resolution circuit. The first six interrupt sources are not masked by the 1bit in the condition code register and have the fixed priority interrupt relationship of: reset, clock monitor fail, COP fail, illegal opcode, and $\overline{\text{XIRQ}}$. (SWI is actually an instruction and has highest priority other than reset in the sense that once the SWI opcode is fetched, no other interrupt can be honored until the SWI vector has been fetched). Each of these sources is an input to the priority resolution circuit. The highest 1bit masked priority input to the resolution circuit is assigned under software control (of the HPRIO register) to be connected to any one of the remaining I bit related interrupt sources. In order to avoid timing races, the HPRIO register may only be written while the 1bit related interrupts are inhibited (1bit in condition code register is a logic one). An interrupt that is assigned to this high priority position is still subject to masking by any associated control bits or the 1bit in the condition code register. The interrupt vector address is not affected by assigning a source to this higher priority position.

Figures 9.5, 9.6, and 9.7 illustrate the interrupt process as it relates to normal processing. Figure 9.5 shows how the CPU begins from a reset and how interrupt detection relates to normal opcode fetches. Figure 9.6 is an expansion of a block in Figure 9.5 and shows how interrupt priority is resolved. Figure 9.7 is an expansion of the SCI interrupt block in Figure 9.7 shows the resolution of interrupt sources within the SCI subsystem.

9.2.5 Highest Priority I Interrupt Register (HPRIO)

This register is used to select one of the I bit related interrupt sources to be elevated to the highest I bit masked position in the priority resolution circuit. In addition, four miscelleneous system control bits are included in this register.

	7	6	5	4	3	2	1	0	
$103C	RBOOT	SMOD	MDA	IRV	PSEL3	PSEL2	PSEL1	PSEL0	HPRIO
RESET	-	-	-	-	0	1	0	1	

RBOOT - Read Bootstrap ROM

The read bootstrap ROM bit only has meaning when the SMOD bit is a one (special bootstrap mode or special test mode). At all other times, this bit is clear and may not be written. When set, upon reset in bootstrap mode only, the small bootstrap loader program is enabled. When clear, by reset in the other three modes, this ROM is disabled and accesses to this area are treated as external accesses.

SMOD - Special Mode

The special mode bit reflects the inverse of the MODB input pin at the rising edge of reset. It is set if the MODB pin is low during reset. If MODB is high during reset, it is cleared. This bit may be cleared under software control from the special modes, thus, changing the operating mode of the MCU, but may never be set by software.

MDA - Mode Select A

The mode select A bit reflects the status of the MODA input pin at the rising edge of reset. While the SMOD bit is set (special bootstrap or special test mode in effect), the MDA bit may be written, thus, changing the operating mode of the MCU. When the SMOD bit is clear, the MODA bit is a read-only bit and the operating mode cannot be changed without going through a reset sequence.

Table 9.5 summarizes the relationship between the SMOD and MDA bits and the MODB and MODA input pins at the rising edge of reset.

IRV - Internal Read Visibility

The TMP68HC11E9 IRV function differs from the other TMP68HC11x versions. The internal read visibility bit is used in the special modes (SMOD=1) to affect visibility of internal reads on the expansion data bus. IRV is writeable anytime if

SMOD=1 and one time only between resets if SMOD=0. If IRV is clear, visibility of internal reads is blocked. If the bit is set, internal reads are visible on the external bus. The user must ensure that bus conflicts do not occur by disabling all external devices from driving the data bus during any internal access.

Table 9.5 Mode Bits Relationship

Inputs		Mode Description	Latched at Reset	
MODB	MODA		SMOD	MDA
1	0	Single Chip	0	0
1	1	Expanded Multiplexed	0	1
0	0	Special Bootstrap	1	0
0	1	Special Test	1	1

1 = Logic High 0 = Logic Low

PSEL3, PSEL2, PSEL1, and PSEL0 - Priority Select

These four priority select bits are used to specify one 1bit related interrupt source which becomes the highest priority 1bit related source (Table 9.6). These bits may be written only while the 1bit in CCR=1 (interrupts masked).

Table 9.6 Highest Priority I Interrupt versus PSEL3-PSEL0

PSEL3	PSEL2	PSEL1	PSEL0	Interrupt Source Promoted
0	0	0	0	Timer Overflow
0	0	0	1	Pulse Accumulator Overflow
0	0	1	0	Pulse Accumulator Input Edge
0	0	1	1	SPI Serial Transfer Complete
0	1	0	0	SCI Serial System
0	1	0	1	Reserved (Default to $\overline{IRQ}$)
0	1	1	0	$\overline{IRQ}$ (External Pin or Parallel I/O)
0	1	1	1	Real Time Interrupt
1	0	0	0	Timer Input Capture 1
1	0	0	1	Timer Input Capture 2
1	0	1	0	Timer Input Capture 3
1	0	1	1	Timer Output Compare 1
1	1	0	0	Timer Output Compare 2
1	1	0	1	Timer Output Compare 3
1	1	1	0	Timer Output Compare 4
1	1	1	1	Timer IC4 OC5

Note : During reset, PSEL3, PSEL2, PSEL1, and PSEL0 are initialized to 0:1:0:1 which corre-sponds to "Reserved (default to $\overline{IRQ}$)" being the highest priority I bit related interupt source.

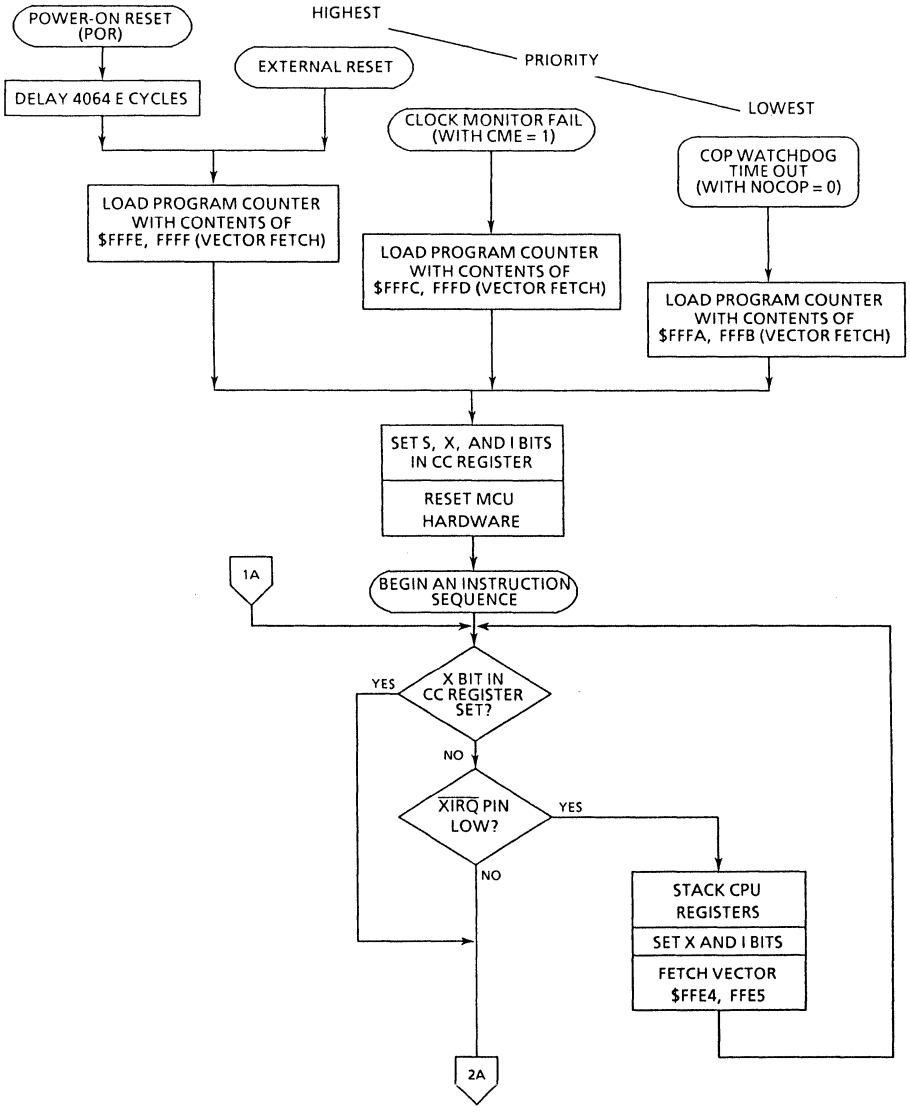


Figure 9.5 Processing Flow Out of Resets (Sheet 1of 2)

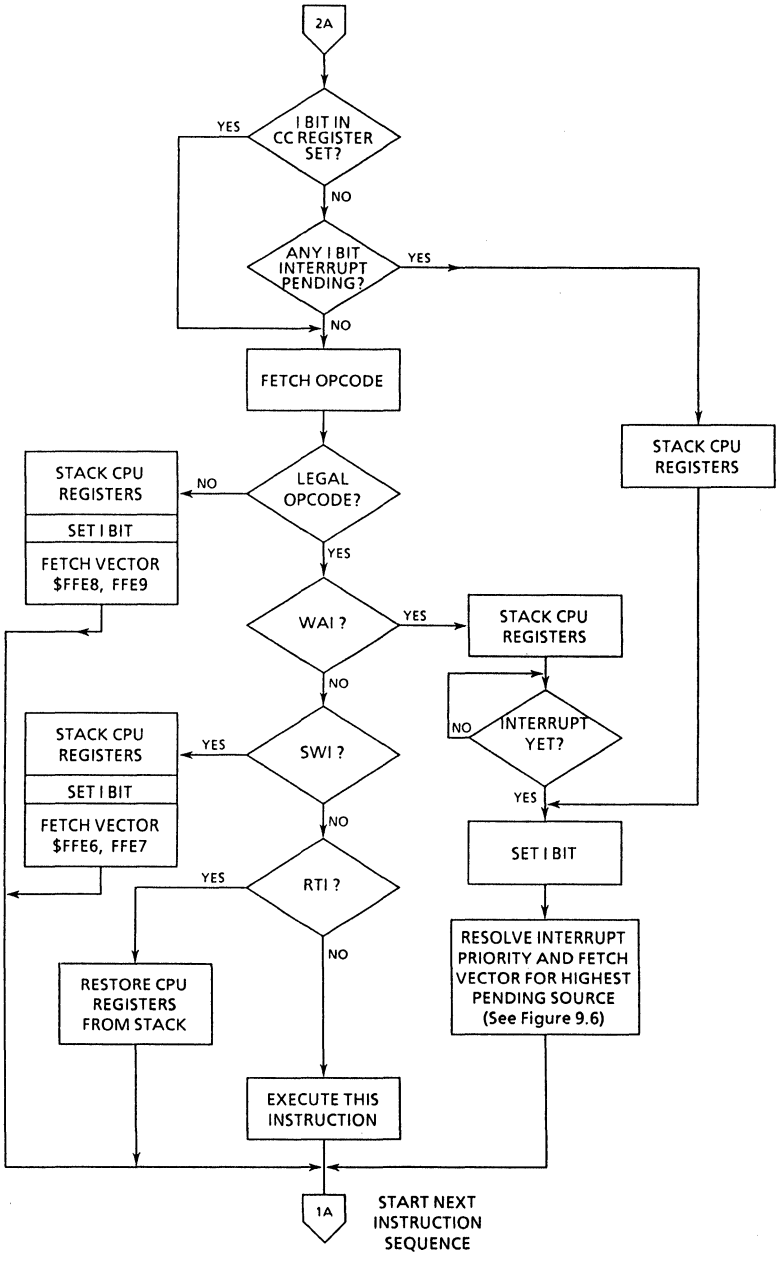


Figure 9.5 Processing Flow Out of Resets (Sheet 2 of 2)

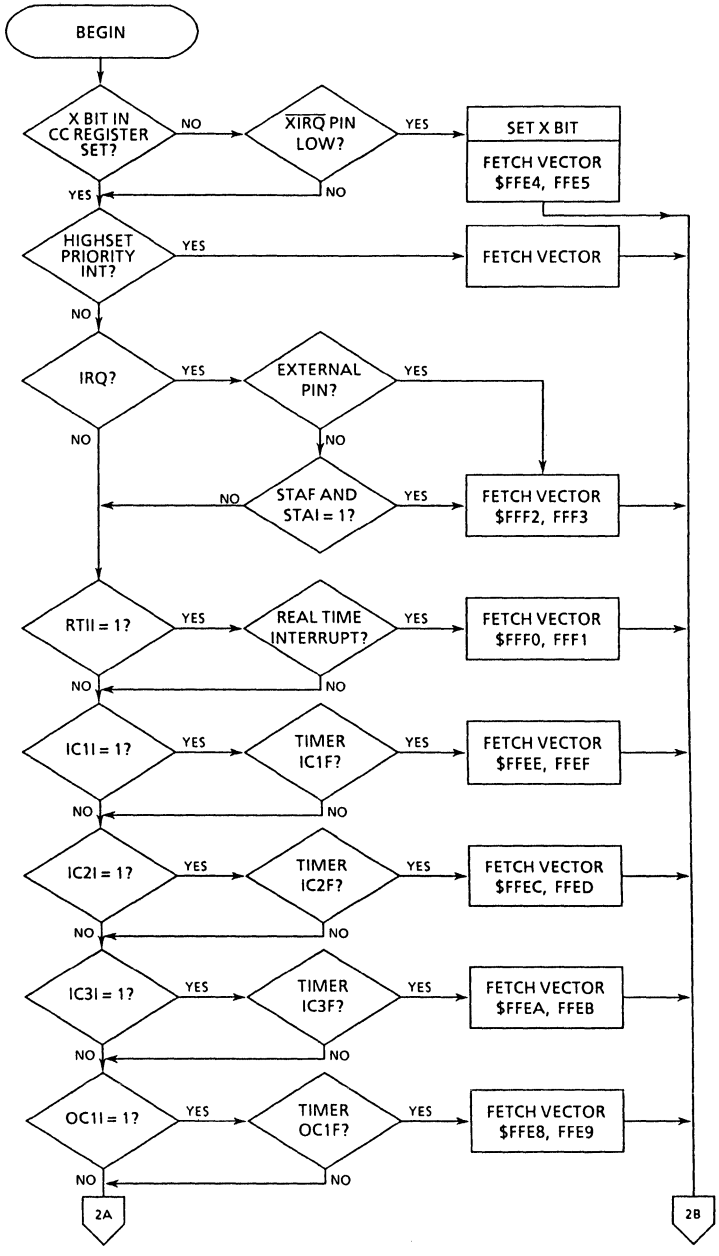
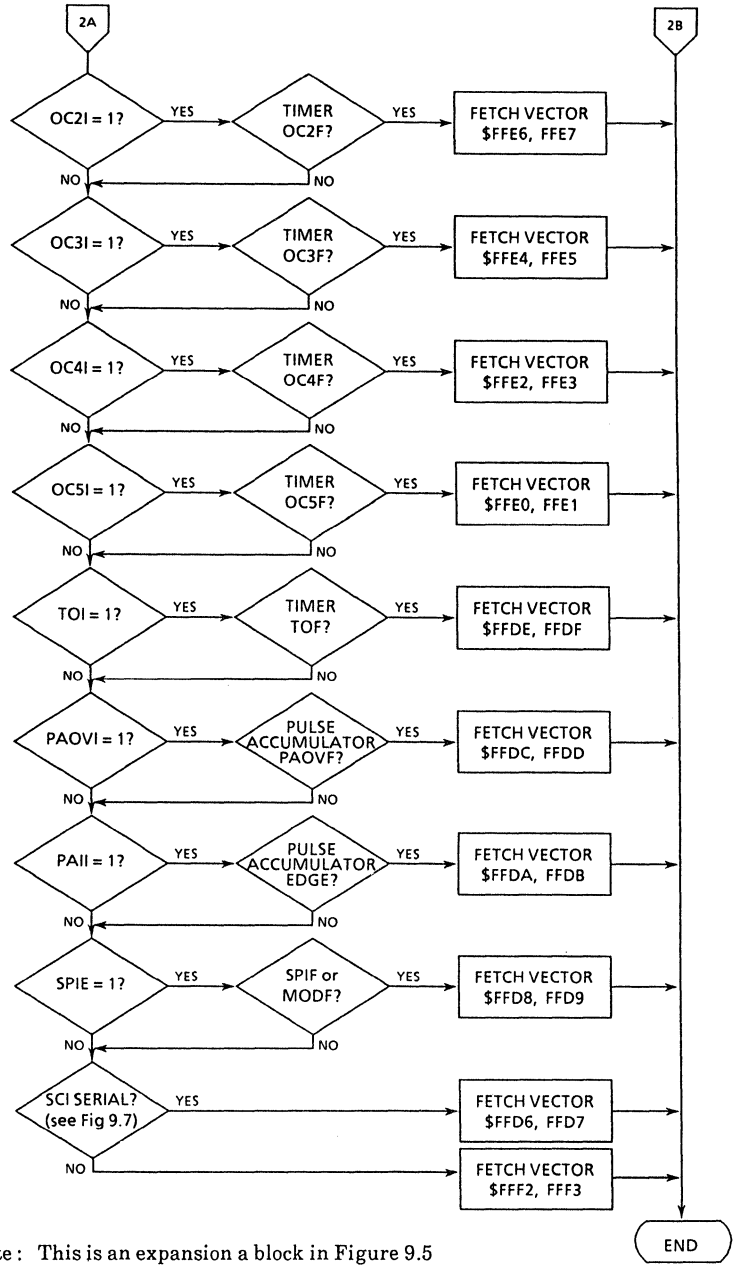
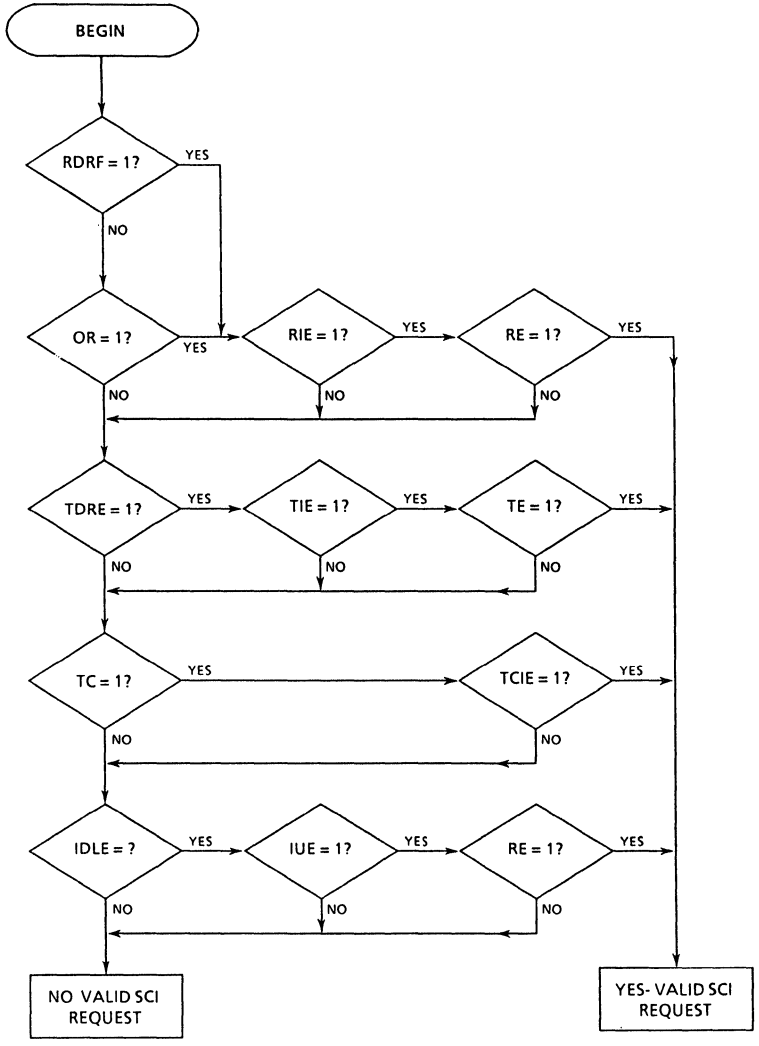


Figure 9.6 Interrupt Priority Resolution (Sheet 1 of 2)



Note : This is an expansion a block in Figure 9.5

Figure 9.6 Interrupt Priority Resolution (Sheet 2 of 2)



Note : This is an expansion a block in Figure 9.6

Figure 9.7 Interrupt Source Resolution Within SCI

9.3 LOW POWER MODES

The MCU contains two programmable low power consumption modes;WAIT and STOP. These two instructions are discussed below. Table 9.7 summarizes the activity on all pins of the MCU for all operating conditions.

9.3.1 WAIT Instruction

The WAI instruction puts the MCU in a low power consumption mode, keeping the oscillator running. Upon execution of a WAI instruction, the machine state is stacked and program execution stops. The wait state can be exited only by an unmasked interrupt or $\overline{\text{RESET}}$. If the I bit is set (interrupts masked) and the COP is disabled, the timer system will be turned off to additionally reduce power consumption. The amount of power savings is application dependent and depends upon circuitry connected to the MCU pins as well as which subsystems (i.e., timer, SPI, SCI) are active when the WAIT mode is entered. Turning off the A/D subsystem by clearing ADPU further reduces WAIT mode current.

Table 9.7 Pin State Summary for RESET, STOP, and WAIT

Pins	Single Chip Modes			Expanded Modes		
	RESET	WAIT	STOP	RESET	WAIT	STOP
Output Only						
E	Active E	Active E	0	Active E	Active E	0
XTAL!!!	Active	Active	1	Active	Active E	1
STRB RW	0	SS	SS	1	1	1
PA4-PA6	0	SS	SS	0	SS	SS
PB0-PB7	0	SS	SS	HI ADD	HI ADD	HI ADD
Input/Output						
RESET	I (0)	I	I	I (0)	I	I
MODA LIR	I (0)	OD (1)	OD (1)	I (1)	OD (1)	OD (1)
MODB V_{STBY}	I (MODB)	I (V_{STBY})	I (V_{STBY})	I (MODES)	I (V_{STBY})	I (V_{STBY})
STRA AS	I (STRA)	I (STRA)	I (STRA)	Active AS	Active AS	0
PA3, PA7	I	I/O	I/O	I	I/O	I/O
PC0-PC7	I	I/O	I/O	ADD/DATA	SP-8/DATA	I
PD0-PD5	I	I/O	I/O	I	I/O	I/O
Input Only	Input Clock or Connect to Crystal with XTAL					
EXTAL	Terminate Unused Inputs to V_{DD}					
$\overline{IRQ}$	Terminate Unused Inputs to V_{DD}					
$\overline{XIRQ}$	Terminate Unused Inputs to V_{DD} or V_{SS}					
PA0-PA2	If Not Used, External Drive Not Required					
PE0-PE7	If Not Used, External Drive Not Required					
V_{RH}-V_{RL}						

SYMBOLS:

DATA	=	Current data present.
I	=	Input pin, if ( ) associated then this is required input state.
IO	=	Input output pin, state determined by data direction register (or configuration of OC5).
LO ADD	=	Low byte of address.
HI ADD	=	High byte of the address.
ADD DATA	=	Low byte of the address multiplexed with data.
OD	=	Open drain output, ( ) current output state.
SS	=	Steady state, output pin stays in current state.
SP-8	=	Address output during WAI period following WAI instruction, stack pointer value, at time of WAI, minus 8.
!!!	=	XTAL is output but not normally usable for any output function beyond crystal drive.

9.3.2 STOP Instruction

The STOP instruction places the MCU in its lowest power consumption mode provided the S bit in the condition code register is clear. If the S bit is set, the STOP mode is disabled and STOP instructions are treated as NOPs (no operation). In the STOP mode, all clocks including the internal oscillator are stopped causing all internal processing to be halted. Recovery from the STOP mode may be accomplished by $\overline{RESET}$,

$\overline{\text{XIRQ}}$, or an unmasked $\overline{\text{IRQ}}$. When the $\overline{\text{XIRQ}}$ is used, the MCU exits from the STOP mode regardless of the state of the X bit in the condition code register;however, the actual recovery sequence differs depending on the state of the X bit. If the X bit is clear, the MCU starts up with the stacking sequence leading to normal service of the $\overline{\text{XIRQ}}$ request. if the X bit is set, then processing will continue with the instruction immediately following the STOP instruction and no $\overline{\text{XIRQ}}$ interrupt service routine is requested. A $\overline{\text{RESET}}$ will always result in an exit from the STOP mode, and the start of MCU operation is determined by the reset vector.

Since the oscillator is stopped in the STOP mode, a restart delay of 4064 clock cycle times may be required to allow oscillator stabilization. If the internal oscillator is being used, this delay is required; however, if a stable external oscillator is being used, a control bit in the OPTION register may be used (DLY = 0) to give a delay of four cycle.

10. CPU, ADDRESSING MODES, AND INSTRUCTION SET

This section provides a description of the CPU registers, addressing modes, and a summary of the M6811 instruction set. Special operations such as subroutine calls and interrupts are described and cycle-by-cycle operations for all instructions are presented.

10.1 CPU REGISTERS

In addition to being able to execute all M6800 and M6801 instructions, the TMP68HC11E9 uses a 4-page opcode map to allow execution of 91 new opcodes (see 10.2.7 Prebyte). Seven registers, discussed in the following paragraphs, are available to programmers as shown in Figure 10.1.

10.1.1 Accumulators A and B

Accumulator A and accumulator B are general-purpose 8-bit registers used to hold operands and results of arithmetic calculations or data manipulations. These two accumulators can be concatenated into a single 16-bit accumulator called the D accumulator.

10.1.2 Index Register X (IX)

The 16-bit IX register is used for indexed mode addressing. It provides a 16-bit indexing value which is added to an 8-bit offset provided in an instruction to create an effective address. The IX register can also be used as a counter or as a temporary storage register.

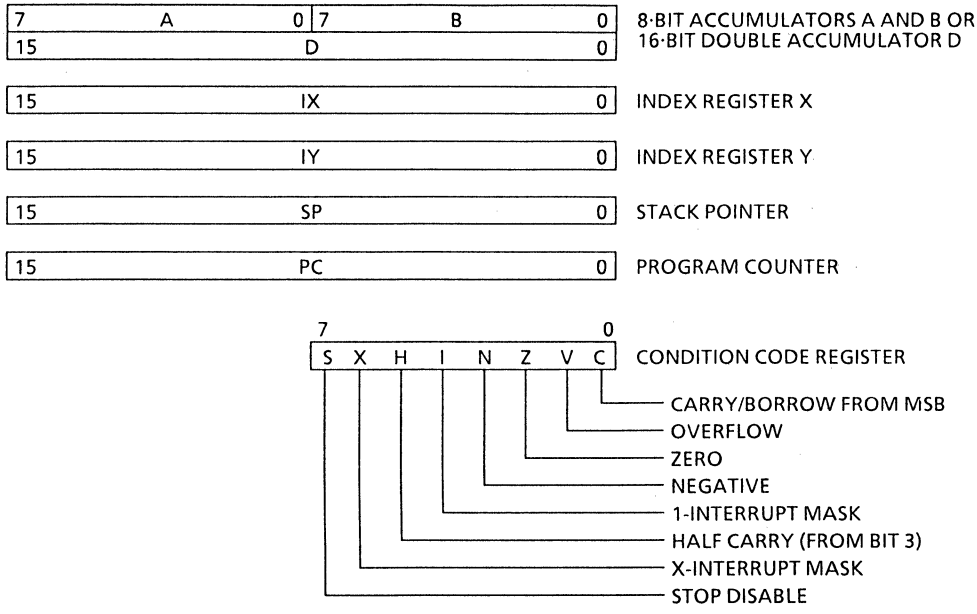


Figure 10.1 Programming Model

10.1.3 Index Register Y (IY)

The 16-bit IY register is also used for indexed mode addressing similar to the IX register;however, all instructions using the IY register require and extra byte of machine code and an extra cycle of execution time since they are two byte opcodes.

10.1.4 Stack Pointer (SP)

The stack pointer (SP) is a 16-bit register that contains the address of the next free location on the stack. The stack is configured as a sequence of last-in-first-out read/write registers which allow important data to be stored during interrupts and subroutine calls. Each time a new byte is added to the stack (a push), the SP is decremented;whereas, each time a byte is removed from the stack (a pull) the SP is incremented.

10.1.5 Program Counter (PC)

The program counter is a 16-bit register that contains the address of the next instruction to be executed.

10.1.6 Condition Code Register (CCR)

The condition code register is an 8-bit register in which each bit signifies the results of the instruction just executed. These bits can be individually tested by a program and a specific action can be taken as a result of the test. Each individual condition code register bit is explained below.

10.1.6.1 Carry/Borrow (C).

The C bit is set if there was a carry or borrow out of the arithmetic logic unit (ALU) during the last arithmetic operation. The C bit is also affected during shift and rotate instructions.

10.1.6.2 Overflow (V).

The overflow bit is set if there was an arithmetic overflow as a result of the operation; otherwise, the V bit is cleared.

10.1.6.3 Zero (Z).

The zero bit is set if the result of the last arithmetic, logic, or data manipulation operation was zero;otherwise, the Z bit is cleared.

10.1.6.4 Negative (N).

The negative bit is set if the result of the last arithmetic, logic, or data manipulation operation was negative;otherwise, the N bit is cleared. A result is said to be negative if its most significant bit is a one.

10.1.6.5 Interrupt Mask (I).

The I interrupt mask bit is set either by hardware or program instruction to disable (mask) all maskable interrupt sources (both external and internal).

10.1.6.6 Half Carry (H).

The half carry bit is set to a logic one when a carry occurs between bits 3 and 4 of the arithmetic logic unit during an ADD, ABA, or ADC instruction;otherwise, the H bit is cleared.

10.1.6.7 X interrupt Mask (X).

The X interrupt mask bit is set only by hardware ($\overline{\text{RESET}}$ or $\overline{\text{XIRQ}}$ acknowledge);and it is cleared only by program instruction (TAP or RTI).

10.1.6.8 Stop Disable (S).

The stop disable bit is set to disable the STOP instruction, and cleared to enable the STOP instruction. The S bit is program controlled. The STOP instruction is treated as no operation (NOP) if the S bit is set.

10.2 ADDRESSING MODES

Six addressing modes can be used to reference memory;they include:immediate, direct, extended, indexed (with either of two 16-bit index registers and an 8-bit offset), inherent and relative. Some instructions require and additional byte before the opcode to accommodate a multi-page opcode map;this byte is called a prebyte.

The following paragraphs provide a description of each addressing mode plus a discussion of the prebyte. In these descriptions the term effective address is used to indicate the address in memory from which the argument is fetched or stored, or from which execution is to proceed.

10.2.1 Immediate Addressing

In the immediate addressing mode, the actual argument is contained in the byte (s) immediately following the instruction, where the number of bytes matches the size of the register. These are two, three, or four (if prebyte is required) byte instructions.

10.2.2 Direct Addressing

In the direct addressing mode (sometimes called zero page addressing), the least significant byte of the operand address is contained in a single byte following the opcode and the most significant byte is assumed to be $00. Direct addressing allows the user to access $0000 through $00FF using two byte instructions and execution time is reduced by eliminating the additional memory access. In most applications, this 512-byte area is reserved for frequently referenced data. In the TMP68HC11E9, software can configure the memory map so that internal RAM, and/or internal registers, or external memory space can occupy these addresses.

10.2.3 Extended Addressing

In the extended addressing mode, the second and third bytes (following the opcode) contain the absolute address of the operand. These are three or four (if prebyte is required) byte instructions: one or two for the opcode, and two for the effective address.

10.2.4 Indexed Addressing

In the indexed addressing mode, one of the index registers (X or Y) is used in calculating the effective address. In this case, the effective address is variable and depends on two factors: 1) the current contents of the index register (X or Y) being used, and 2) the 8-bit unsigned offset contained in the instruction. This addressing mode allows referencing any memory location in the 64 K byte address space. These are usually two or three (if prebyte is required) byte instructions, the opcode plus the 8-bit offset.

10.2.5 Inherent Addressing

In the inherent addressing mode, all of the information is contained in the opcode. The operands (if any) are registers and no memory reference is required. These are usually one or two byte instructions.

10.2.6 Relative Addressing

The relative addressing mode is used for branch instructions. If the branch condition is true, the contents of the 8-bit signed byte following the opcode (the offset) is added to the contents of the program counter to form the effective branch address;otherwise, control proceeds to the next instruction. These are usually two byte instructions.

10.2.7 Prebyte

In order to expand the number of instructions used in the TMP68HC11E9, a prebyte instruction has been added to certain instructions. The instructions affected are usually associated with index register Y. The instruction opcodes which do not require a prebyte could be considered as page 1 of the overall opcode map. The remaining opcodes could be considered as pages 2, 3, and 4 of the opcode map and would require a prebyte;$18 for page 2, $1A for page 3, and $CD for page 4.

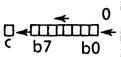
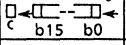
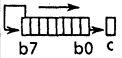
10.3 INSTRUCTION SET

The central processing unit (CPU) in the TMP68HC11E9 is basically a proper extension of the MC6801 CPU. In addition to its ability to execute all M6800 and M6801 instructions, the TMP68HC11E9 CPU has a paged operation code (opcode) map with a total of 91 new opcodes. Major functional additions include a second 16-bit index register (Y register), two types of 16-by-16 divide instruction, STOP and WAIT instructions, and bit manipulation instructions.

Table 10.1 shows all TMP68HC11E9 instructions in all possible addressing modes. For each instruction, the operand construction is shown as well as the total number of machine code bytes and execution time in CPU E-clock cycles. Notes are provided at the end of Table 10.1 which explain the letters in the Operand and Execution Time columns for some instructions. Definitions of "Special Ops" found in the Boolean Expression column are found in Figure 10.2.

Table 10.2 through 10.8 prdovide a detailed description of the information present on the address bus, data bus, and the read/write ($R/\overline{W}$) line during each cycle of each instruction. The information is useful in comparing actual with expected results during debug of both software and hardware as the program is executed. The information is categorized in groups according to addressing mode and number of cycles per instruction. In general, instructions with the same address mode and number of cycles execute in the same manner. Exceptions are indicated in the table.

Table 10.1 TMP68HC11E9 Instructions, Addressing Modes, and Execution Times (Sheet 1 of 7)

Source Form (s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand (s)	Bytes	Cycle	Cycle by Cycle*	Condition Codes S X H I N Z V C
ABA	Add Accumulators	A + B→A	INH	1B		1	2	2-1	- - ↕ - ↕ ↕ ↕ ↕
ABX	Add B to X	IX + 00 : B→IX	INH	3A		1	3	2-2	- - - - - - - -
ABY	Add B to Y	IY + 00 : B→IY	INH	18 3A		2	4	2-4	- - - - - - - -
ADCA (opr)	Add with Carry to A	A + M + C→A	A IMM	89	ii	2	2	3-1	- - ↕ - ↕ ↕ ↕ ↕
			A DIR	99	dd	2	3	4-1	
			A EXT	B9	hh ll	3	4	5-2	
			A IND, X	A9	ff	2	4	6-2	
			A IND, Y	18 A9	ff	3	5	7-2	
ADCB (opr)	Add with Carry to B	B + M + C→B	B IMM	C9	ii	2	2	3-1	- - ↕ - ↕ ↕ ↕ ↕
			B DIR	D9	dd	2	3	4-1	
			B EXT	F9	hh ll	3	4	5-2	
			B IND, X	E9	ff	2	4	6-2	
			B IND, Y	18 E9	ff	3	5	7-2	
ADDA (opr)	Add Memory to A	A + M→A	A IMM	8B	ii	2	2	3-1	- - ↕ - ↕ ↕ ↕ ↕
			A DIR	9B	dd	2	3	4-1	
			A EXT	BB	hh ll	3	4	5-2	
			A IND, X	AB	ff	2	4	6-2	
			A IND, Y	18 AB	ff	3	5	7-2	
ADDB (opr)	Add Memory to B	B + M→B	B IMM	CB	ii	2	2	3-1	- - ↕ - ↕ ↕ ↕ ↕
			B DIR	DB	dd	2	3	4-1	
			B EXT	FB	hh ll	3	4	5-2	
			B IND, X	EB	ff	2	4	6-2	
			B IND, Y	18 EB	ff	3	5	7-2	
ADDD (opr)	Add 16-Bit to D	D + M : M + 1 →D	IMM	C3	jj kk	3	4	3-3	- - - - ↕ ↕ ↕ ↕
			DIR	D3	dd	2	5	4-7	
			EXT	F3	hh ll	3	6	5-10	
			IND, X	E3	ff	2	6	6-10	
			IND, Y	18 E3	ff	3	7	7-8	
ANDA (opr)	AND A with Memory	A • M→A	A IMM	84	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			A DIR	94	dd	2	3	4-1	
			A EXT	B4	hh ll	3	4	5-2	
			A IND, X	A4	ff	2	4	6-2	
			A IND, Y	18 A4	ff	3	5	7-2	
ANDB (opr)	AND B with Memory	B • M→B	B IMM	C4	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			B DIR	D4	dd	2	3	4-1	
			B EXT	F4	hh ll	3	4	5-2	
			B IND, X	E4	ff	2	4	6-2	
			B IND, Y	18 E4	ff	3	5	7-2	
ASL (opr)	Arithmetic Shift Left		EXT	78	hh ll	3	6	5-8	- - - - ↕ ↕ ↕ ↕
			IND, X	68	ff	2	6	6-3	
			IND, Y	18 68	ff	3	7	7-3	
ASLA			A INH	48		1	2	2-1	
ASLB			B INH	58		1	2	2-1	
ASLD	Arithmetic Shift Left Double		INH	05		1	3	2-2	- - - - ↕ ↕ ↕ ↕
ASR (opr)	Arithmetic Shift Right		EXT	77	hh ll	3	6	5-8	- - - - ↕ ↕ ↕ ↕
			IND, X	67	ff	2	6	6-3	
			IND, Y	18 67	ff	3	7	7-3	
ASRA			A INH	47		1	2	2-1	
ASRB			B INH	57		1	2	2-1	
BCC (rel)	Branch if Carry Clear	?C = 0	REL	24	rr	2	3	8-1	- - - - - - - -
BCLR (opr) (msk)	Clear Bit (s)	M • (mm) →M	DIR	15	dd mm	3	6	4-10	- - - - ↕ ↕ 0 -
			IND, X	1D	ff mm	3	7	6-13	
			IND, Y	18 1D	ff mm	4	8	7-10	
BCS (rel)	Branch if Carry Set	?C = 1	REL	25	rr	2	3	8-1	- - - - - - - -
BEQ (rel)	Branch if = Zero	?Z = 1	REL	27	rr	2	3	8-1	- - - - - - - -

* Cycle-by-cycle number provides a reference to Table 10.2 through 10.8 which detail cycle-by-cycle operation.
Example: Table 10.1 cycle-by-cycle column reference number 2.4 equals Table 10.2 line item 2.4.

Table 10.1 TMP68HC11E9 Instructions, Addressing Modes, and Execution Times (Sheet 2 of 7)

Source Form (s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes
				Opcode	Operand (s)				S X H I N Z V C
BGE (rel)	Branch if ≥ Zero	?N⊕V = 0	REL	2C	rr	2	3	8-1	- - - - - - - -
BGT (rel)	Branch if > Zero	?Z + (N⊕V) = 0	REL	2E	rr	2	3	8-1	- - - - - - - -
BHI (rel)	Branch if Higher	?C + Z = 0	REL	22	rr	2	3	8-1	- - - - - - - -
BHS (rel)	Branch if Higher or Same	?C = 0	REL	24	rr	2	3	8-1	- - - - - - - -
BITA (opr)	Bit (s) Test A with Memory	A • M	A IMM	85	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			A DIR	95	dd	2	3	4-1	
			A EXT	B5	hh ll	3	4	5-2	
			A IND, X	A5	ff	2	4	6-2	
			A IND, Y	18 A5	ff	3	5	7-2	
BITB (opr)	Bit (s) Test B with Memory	B • M	B IMM	C5	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			B DIR	D5	dd	2	3	4-1	
			B EXT	F5	hh ll	3	4	5-2	
			B IND, X	E5	ff	2	4	6-2	
			B IND, Y	18 E5	ff	3	5	7-2	
BLE (rel)	Branch if ≤ Zero	?Z + (N⊕V) = 1	REL	2F	rr	2	3	8-1	- - - - - - - -
BLO (rel)	Branch if Lower	?C = 1	REL	25	rr	2	3	8-1	- - - - - - - -
BLS (rel)	Branch if Lower or Same	?C + Z = 1	REL	23	rr	2	3	8-1	- - - - - - - -
BLT (rel)	Branch if < Zero	?N⊕V = 1	REL	2D	rr	2	3	8-1	- - - - - - - -
BMI (rel)	Branch if Minus	?N = 1	REL	2B	rr	2	3	8-1	- - - - - - - -
BNE (rel)	Branch if Not = Zero	?Z = 0	REL	26	rr	2	3	8-1	- - - - - - - -
BPL (rel)	Branch if Plus	?N = 0	REL	2A	rr	2	3	8-1	- - - - - - - -
BRA (rel)	Branch Always	?1 = 1	REL	20	rr	2	3	8-1	- - - - - - - -
BRCLR (opr) (msk) (rel)	Branch if Bit (s) Clear	?M • mm = 0	DIR	13	dd mm rr	4	6	4-11	- - - - - - - -
			IND, X	1F	ff mm rr	4	7	6-14	
			IND, Y	18 1F	ff mm rr	5	8	7-11	
BRN (rel)	Branch Never	?1 = 0	REL	21	rr	2	3	8-1	- - - - - - - -
BRSET (opr) (msk) (rel)	Branch if Bit (s) Set	? ($\overline{M}$) • mm = 0	DIR	12	dd mm rr	4	6	4-11	- - - - - - - -
			IND, X	1E	ff mm rr	4	7	6-14	
			IND, Y	18 1E	ff mm rr	5	8	7-11	
BSET (opr) (msk)	Set Bit (s)	M + mm→M	DIR	14	dd mm	3	6	4-10	- - - - ↕ ↕ 0 -
			IND, X	1C	ff mm	3	7	6-13	
			IND, Y	18 1C	ff mm	4	8	7-10	
BSR (rel)	Branch to Subroutine	See Special Ops	REL	8D	rr	2	6	8-2	- - - - - - - -
BVC (rel)	Branch if Overflow Clear	?V = 0	REL	28	rr	2	3	8-1	- - - - - - - -
BVS (rel)	Branch if Overflow Set	?V = 1	REL	29	rr	2	3	8-1	- - - - - - - -
CBA	Compare A to B	A-B	INH	11		1	2	2-1	- - - - ↕ ↕ ↕ ↕
CLC	Clear Carry Bit	0→C	INH	0C		1	2	2-1	- - - - - - - 0
CLI	Clear Interrupt Mask	0→I	INH	0E		1	2	2-1	- - - 0 - - - -
CLR (opr)	Clear Memory Byte	0→M	EXT	7F	hh ll	3	6	5-8	- - - - 0 1 0 0
			IND, X	6F	ff	2	6	6-3	
			IND, Y	18 6F	ff	3	7	7-3	
CLRA	Clear Accumulator A	0→A	A INH	4F		1	2	2-1	- - - - 0 1 0 0
CLRB	Clear Accumulator B	0→B	B INH	5F		1	2	2-1	- - - - 0 1 0 0
CLV	Clear Overflow Flag	0→V	INH	0A		1	2	2-1	- - - - - - 0 -
CMPA (opr)	Compare A to Memory	A-M	A IMM	81	ii	2	2	3-1	- - - - ↕ ↕ ↕ ↕
			A DIR	91	dd	2	3	4-1	
			A EXT	B1	hh ll	3	4	5-2	
			A IND, X	A1	ff	2	4	6-2	
			A IND, Y	18 A1	ff	3	5	7-2	

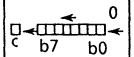
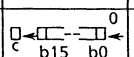
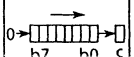
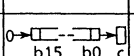
* Cycle-by-cycle number provides a reference to Table 10.2 through 10.8 which detail cycle-by-cycle operation.
Example: Table 10.1 cycle-by-cycle column reference number 2.4 equals Table 10.2 line item 2.4.

Table 10.1 TMP68HC11E9 Instructions, Addressing Modes, and Execution Times (Sheet 3 of 7)

Source Form (s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Machine Coding (Hexadecimal) Operand (s)	Bytes	Cycle	Cycle by Cycle*	Condition Codes S X H I N Z V C
CMPB (opr)	Compare B to Memory	B-M	B IMM	C1	ii	2	2	3-1	- - - - ↕ ↕ ↕ ↕
			B DIR	D1	dd	2	3	4-1	
			B EXT	F1	hh ll	3	4	5-2	
			B IND, X	E1	ff	2	4	6-2	
			B IND, Y	18 E1	ff	3	5	7-2	
COM(opr)	1's Complement Memory Byte	$FF-M→M	EXT	73	hh ll	3	6	5-8	- - - - ↕ ↕ 0 1
			IND, X	63	ff	2	6	6-3	
			IND, Y	18 63	ff	3	7	7-3	
COMA	1's Complement A	$FF-A→A	A INH	43		1	2	2-1	- - - - ↕ ↕ 0 1
COMB	1's Complement B	$FF-B→B	B INH	53		1	2	2-1	- - - - ↕ ↕ 0 1
CPD (opr)	Compare D to Memory 16-Bit	D-M : M + 1	IMM	1A 83	jj kk	4	5	3-5	- - - - ↕ ↕ ↕ ↕
			DIR	1A 93	dd	3	6	4-9	
			EXT	1A B3	hh ll	4	7	5-11	
			IND, X	1A A3	ff	3	7	6-11	
			IND, Y	CD A3	ff	3	7	7-8	
CPX (opr)	Compare X to Memory 16-Bit	IX-M : M + 1	IMM	8C	jj kk	3	4	3-3	- - - - ↕ ↕ ↕ ↕
			DIR	9C	dd	2	5	4-7	
			EXT	BC	hh ll	3	6	5-10	
			IND, X	AC	ff	2	6	6-10	
			IND, Y	CD AC	ff	3	7	7-8	
CPY (opr)	Compare Y to Memory 16-Bit	IY-M : M + 1	IMM	18 8C	jj kk	4	5	3-5	- - - - ↕ ↕ ↕ ↕
			DIR	18 9C	dd	3	6	4-9	
			EXT	18 BC	hh ll	4	7	5-11	
			IND, X	1A AC	ff	3	7	6-11	
			IND, Y	18 AC	ff	3	7	7-8	
DAA	Decimal Adjust A	Adjust Sum to BCD	INH	19		1	2	2-1	- - - - ↕ ↕ ↕ ↕
DEC (opr)	Decrement Memory Byte	M-1→M	EXT	7A	hh ll	3	6	5-8	- - - - ↕ ↕ ↕ -
			IND, X	6A	ff	2	6	6-3	
			IND, Y	18 6A	ff	3	7	7-3	
DECA	Decrement Accumulator A	A-1→A	A INH	4A		1	2	2-1	- - - - ↕ ↕ ↕ -
DECB	Decrement Accumulator B	B-1→B	B INH	5A		1	2	2-1	- - - - ↕ ↕ ↕ -
DES	Decrement Stack Pointer	SP-1→SP	INH	34		1	3	2-3	- - - - - - - -
DEX	Decrement Index Register X	IX-1→IX	INH	09		1	3	2-2	- - - - - ↕ - -
DEY	Decrement Index Register Y	IY-1→IY	INH	18 09		2	4	2-4	- - - - - ↕ - -
EORA (opr)	Exclusive OR A with Memory	A ⊕ M→A	A IMM	88	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			A DIR	98	dd	2	3	4-1	
			A EXT	B8	hh ll	3	4	5-2	
			A IND, X	A8	ff	2	4	6-2	
			A IND, Y	18 A8	ff	3	5	7-2	
EORB (opr)	Exclusive OR B with Memory	B ⊕ M→B	B IMM	C8	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			B DIR	D8	dd	2	3	4-1	
			B EXT	F8	hh ll	3	4	5-2	
			B IND, X	E8	ff	2	4	6-2	
			B IND, Y	18 E8	ff	3	5	7-2	
FDIV	Fractional Divide 16 by 16	D / IX→IX ; r→D	INH	03		1	41	2-17	- - - - - ↕ ↕ ↕
IDIV	Integer Divide 16 by 16	D / IX→IX ; r→D	INH	02		1	41	2-17	- - - - - ↕ 0 ↕
INC (opr)	Increment Memory Byte	M + 1→M	EXT	7C	hh ll	3	6	5-8	- - - - ↕ ↕ ↕ -
			IND, X	6C	ff	2	6	6-3	
			IND, Y	18 6C	ff	3	7	7-3	
INCA	Increment Accumulator A	A + 1→A	A INH	4C		1	2	2-1	- - - - ↕ ↕ ↕ -
INCB	Increment Accumulator B	B + 1→B	B INH	5C		1	2	2-1	- - - - ↕ ↕ ↕ -
INS	Increment Stack Pointer	SP + 1→SP	INH	31		1	3	2-3	- - - - - - - -

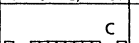
* Cycle-by-cycle number provides a reference to Table 10.2 through 10.8 which detail cycle-by-cycle operation.
Example:Table 10.1 cycle-by-cycle column reference number 2.4 equals Table 10.2 line item 2.4.

Table 10.1 TMP68HC11E9 Instructions, Addressing Modes, and Execution Times (Sheet 4 of 7)

Source Form (s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes
				Opcode	Operand (s)				S X H I N Z V C
INX	Increment Index Register X	IX + 1→IX	INH	08		1	3	2-2	- - - - - ↕ - -
INY	Increment Index Register Y	IY + 1→IY	INH	18 08		2	4	2-4	- - - - - ↕ - -
JMP (opr)	Jump	See Special Ops	EXT	7E	hh ll	3	3	5-1	- - - - - - - -
			IND, X	6E	ff	2	3	6-1	
			IND, Y	18 6E	ff	3	4	7-1	
JSR (opr)	Jump to Subroutine	See Special Ops	DIR	9D	dd	2	5	4-8	- - - - - - - -
			EXT	BD	hh ll	3	6	5-12	
			IND, X	AD	ff	2	6	6-12	
			IND, Y	18 AD	ff	3	7	7-9	
LDAA (opr)	Load Accumulator A	M→A	A IMM	86	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			A DIR	96	dd	2	3	4-1	
			A EXT	B6	hh ll	3	4	5-2	
			A IND, X	A6	ff	2	4	6-2	
			A IND, Y	18 A6	ff	3	5	7-2	
LDAB (opr)	Load Accumulator B	M→B	B IMM	C6	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			B DIR	D6	dd	2	3	4-1	
			B EXT	F6	hh ll	3	4	5-2	
			B IND, X	E6	ff	2	4	6-2	
			B IND, Y	18 E6	ff	3	5	7-2	
LDD (opr)	Load Double Accumulator D	M→A, M + 1→B	IMM	CC	jj kk	3	3	3-2	- - - - ↕ ↕ 0 -
			DIR	DC	dd	2	4	4-3	
			EXT	FC	hh ll	3	5	5-4	
			IND, X	EC	ff	2	5	6-6	
			IND, Y	18 EC	ff	3	6	7-6	
LDS (opr)	Load Stack Pointer	M : M + 1→SP	IMM	8E	jj kk	3	3	3-2	- - - - ↕ ↕ 0 -
			DIR	9E	dd	2	4	4-3	
			EXT	BE	hh ll	3	5	5-4	
			IND, X	AE	ff	2	5	6-6	
			IND, Y	18 AE	ff	3	6	7-6	
LDX (opr)	Load Index Register X	M : M + 1→IX	IMM	CE	jj kk	3	3	3-2	- - - - ↕ ↕ 0 -
			DIR	DE	dd	2	4	4-3	
			EXT	FE	hh ll	3	5	5-4	
			IND, X	EE	ff	2	5	6-6	
			IND, Y	CD EE	ff	3	6	7-6	
LDY (opr)	Load Index Register Y	M : M + 1→IY	IMM	18 CE	jj kk	4	4	3-4	- - - - ↕ ↕ 0 -
			DIR	18 DE	dd	3	5	4-5	
			EXT	18 FE	hh ll	4	6	5-6	
			IND, X	1A EE	ff	3	6	6-7	
			IND, Y	18 EE	ff	3	6	7-6	
LSL (opr)	Logical Shift Left		EXT	78	hh ll	3	6	5-8	- - - - ↕ ↕ ↕ ↕
			IND, X	68	ff	2	6	6-3	
			IND, Y	18 68	ff	3	7	7-3	
LSLA			A INH	48		1	2	2-1	
LSLB			A INH	58		1	2	2-1	
LSLD	Logical Shift Left Double		INH	05		1	3	2-2	- - - - ↕ ↕ ↕ ↕
LSR (opr)	Logical Shift Right		EXT	74	hh ll	3	6	5-8	- - - - 0 ↕ ↕ ↕
			IND, X	64	ff	2	6	6-3	
			IND, Y	18 64	ff	3	7	7-3	
LSRA			A INH	44		1	2	2-1	
LSRB			B INH	54		1	2	2-1	
LSRD	Logical Shift Right Double		INH	04		1	3	2-2	- - - - 0 ↕ ↕ ↕
MUL	Multiply 8 by 8	A x B→D	INH	3D		1	10	2-13	- - - - - - - ↕

* Cycle-by-cycle number provides a reference to Table 10.2 through 10.8 which detail cycle-by-cycle operation.
Example:Table 10.1 cycle-by-cycle column reference number 2.4 equals Table 10.2 line item 2.4.

Table 10.1 TMP68HC11E9 Instructions, Addressing Modes, and Execution Times (Sheet 5 of 7)

Source Form (s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes
				Opcode	Operand (s)				S X H I N Z V C
NEG (opr)	2's Complement Memory Byte	0 – M→M	EXT	70	hh ll	3	6	5-8	- - - - ↕ ↕ ↕ ↕
			IND, X	60	ff	2	6	6-3	
			IND, Y	18 60	ff	3	7	7-3	
NEGA	2's Complement A	0 – A→A	A INH	40		1	2	2-1	- - - - ↕ ↕ ↕ ↕
NEGB	2's Complement B	0 – B→B	B INH	50		1	2	2-1	- - - - ↕ ↕ ↕ ↕
NOP	No Operation	No Operation	INH	01		1	2	2-1	- - - - - - - -
ORAA (opr)	OR Accumulator A (Inclusive)	A + M→A	A IMM	8A	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			A DIR	9A	dd	2	3	4-1	
			A EXT	BA	hh ll	3	4	5-2	
			A IND, X	AA	ff	2	4	6-2	
			A IND, Y	18 AA	ff	3	5	7-2	
ORAB (opr)	OR Accumulator B (Inclusive)	B + M→B	B IMM	CA	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			B DIR	DA	dd	2	3	4-1	
			B EXT	FA	hh ll	3	4	5-2	
			B IND, X	EA	ff	2	4	6-2	
			B IND, Y	18 EA	ff	3	5	7-2	
PSHA	Push A onto Stack	A→Stk, SP = SP – 1	A INH	36		1	3	2-6	- - - - - - - -
PSHB	Push B onto Stack	B→Stk, SP = SP – 1	B INH	37		1	3	2-6	- - - - - - - -
PSHX	Push X onto Stack (Lo First)	IX→Stk, SP = SP – 2	INH	3C		1	4	2-7	- - - - - - - -
PSHY	Push Y onto Stack (Lo First)	IY→Stk, SP = SP – 2	INH	18 3C		2	5	2-8	- - - - - - - -
PULA	Pull A from Stack	SP = SP + 1, A←Stk	A INH	32		1	4	2-9	- - - - - - - -
PULB	Pull B from Stack	SP = SP + 1, B←Stk	B INH	33		1	4	2-9	- - - - - - - -
PULX	Pull X from Stack (Hi First)	SP = SP + 2, IX←Stk	INH	38		1	5	2-10	- - - - - - - -
PULY	Pull Y from Stack (Hi First)	SP = SP + 2, IY←Stk	INH	18 38		2	6	2-11	- - - - - - - -
ROL (opr)	Rotate Left		EXT	79	hh ll	3	6	5-8	- - - - ↕ ↕ ↕ ↕
			IND, X	69	ff	2	6	6-3	
			IND, Y	18 69	ff	3	7	7-3	
ROLA		C b7 ← b0	A INH	49		1	2	2-1	
ROLB			B INH	59		1	2	2-1	
ROR (opr)	Rotate Right		EXT	76	hh ll	3	6	5-8	- - - - ↕ ↕ ↕ ↕
			IND, X	66	ff	2	6	6-3	
			IND, Y	18 66	ff	3	7	7-3	
RORA		C b7 → b0	A INH	46		1	2	2-1	
RORB			B INH	56		1	2	2-1	
RTI	Return from Interrupt	See Special Ops	INH	3B		1	12	2-14	↕ ↕ ↕ ↕ ↕ ↕ ↕ ↕
RTS	Return from Subroutine	See Special Ops	INH	39		1	5	2-12	- - - - - - - -
SBA	Subtract B from A	A – B→A	INH	10		1	2	2-1	- - - - ↕ ↕ ↕ ↕
SBCA (opr)	Subtract with Carry from A	A – M – C→A	A IMM	82	ii	2	2	3-1	- - - - ↕ ↕ ↕ ↕
			A DIR	92	dd	2	3	4-1	
			A EXT	B2	hh ll	3	4	5-2	
			A IND, X	A2	ff	2	4	6-2	
			A IND, Y	18 A2	ff	3	5	7-2	
SBCB (opr)	Subtract from Carry from B	B – M – C→B	B IMM	C2	ii	2	2	3-1	- - - - ↕ ↕ ↕ ↕
			B DIR	D2	dd	2	3	4-1	
			B EXT	F2	hh ll	3	4	5-2	
			B IND, X	E2	ff	2	4	6-2	
			B IND, Y	18 E2	ff	3	5	7-2	
SEC	Set Carry	1→C	INH	0D		1	2	2-1	- - - - - - - - 1
SEI	Set Interrupt Mask	1→I	INH	0F		1	2	2-1	- - - 1 - - - -
SEV	Set Overflow Flag	1→V	INH	0B		1	2	2-1	- - - - - - 1 -

* Cycle-by-cycle number provides a reference to Table 10.2 through 10.8 which detail cycle-by-cycle operation.

Example:Table 10.1 cycle-by-cycle column reference number 2.4 equals Table 10.2 line item 2.4.

Table 10.1 TMP68HC11E9 Instructions, Addressing Modes, and Execution Times (Sheet 6 of 7)

Source Form (s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes
				Opcode	Operand (s)				S X H I N Z V C
STAA (opr)	Store Accumulator A	A→M	A DIR	97	dd	2	3	4-2	- - - - ↕ ↕ 0 -
			A EXT	B7	hh ll	3	4	5-3	
			A IND, X	A7	ff	2	4	6-5	
			A IND, Y	18 A7	ff	3	5	7-5	
STAB (opr)	Store Accumulator B	B→M	B DIR	D7	dd	2	3	4-2	- - - - ↕ ↕ 0 -
			B EXT	F7	hh ll	3	4	5-3	
			B IND, X	E7	ff	2	4	6-5	
			B IND, Y	18 E7	ff	3	5	7-5	
STD (opr)	Store Accumulator D	A→M, B→M + 1	DIR	DD	dd	2	4	4-4	- - - - ↕ ↕ 0 -
			EXT	FD	hh ll	3	5	5-5	
			IND, X	ED	ff	2	5	6-8	
			IND, Y	18 ED	ff	3	6	7-7	
STOP	Stop Internal Clocks		INH	CF		1	2	2-1	- - - - - - - -
STS (opr)	Store Stack Pointer	SP→M : M + 1	DIR	9F	dd	2	4	4-4	- - - - ↕ ↕ 0 -
			EXT	BF	hh ll	3	5	5-5	
			IND, X	AF	ff	2	5	6-8	
			IND, Y	18 AF	ff	3	6	7-7	
STX (opr)	Store Index Register X	IX→M : M + 1	DIR	DF	dd	2	4	4-4	- - - - ↕ ↕ 0 -
			EXT	FF	hh ll	3	5	5-5	
			IND, X	EF	ff	2	5	6-8	
			IND, Y	CD EF	ff	3	6	7-7	
STY (opr)	Store Index Register Y	IY→M : M + 1	DIR	18 DF	dd	3	5	4-6	- - - - ↕ ↕ 0 -
			EXT	18 FF	hh ll	4	6	5-7	
			IND, X	1A EF	ff	3	6	6-9	
			IND, Y	18 EF	ff	3	6	7-7	
SUBA (opr)	Subtract Memory from A	A – M→A	A IMM	80	ii	2	2	3-1	- - - - ↕ ↕ ↕ ↕
			A DIR	90	dd	2	3	4-1	
			A EXT	B0	hh ll	3	4	5-2	
			A IND, X	A0	ff	2	4	6-2	
			A IND, Y	18 A0	ff	3	5	7-2	
SUBB (opr)	Subtract Memory from B	B – M→B	B IMM	C0	ii	2	2	3-1	- - - - ↕ ↕ ↕ ↕
			B DIR	D0	dd	2	3	4-1	
			B EXT	F0	hh ll	3	4	5-2	
			B IND, X	E0	ff	2	4	6-2	
			B IND, Y	18 E0	ff	3	5	7-2	
SUBD (opr)	Subtract Memory from D	D – M : M + 1→D	IMM	83	jj kk	3	4	3-3	- - - - ↕ ↕ ↕ ↕
			DIR	93	dd	2	5	4-7	
			EXT	B3	hh ll	3	6	5-10	
			IND, X	A3	ff	2	6	6-10	
			IND, Y	18 A3	ff	3	7	7-8	
SWI	Software Interrupt	See Special Ops	INH	3F		1	14	2-15	- - - 1 - - - -
TAB	Transfer A to B	A→B	INH	16		1	2	2-1	- - - - ↕ ↕ 0 -
TAP	Transfer A to CC Register	A→CCR	INH	06		1	2	2-1	↕ ↕ ↕ ↕ ↕ ↕ ↕ ↕
TBA	Transfer B to A	B→A	INH	17		1	2	2-1	- - - - ↕ ↕ 0 -
TEST	TEST (Only in Test Modes)	Address Bus Counts	INH	00		1	**	2-20	- - - - - - - -
TPA	Transfer CC Register to A	CCR→A	INH	07		1	2	2-1	- - - - - - - -
TST (opr)	Test for Zero or Minus	M – 0	EXT	7D	hh ll	3	6	5-9	- - - - ↕ ↕ 0 0
			IND, X	6D	ff	2	6	6-4	
			IND, Y	18 6D	ff	3	7	7-4	
TSTA		A – 0	A INH	4D		1	2	2-1	- - - - ↕ ↕ 0 0
TSTB		B – 0	B INH	5D		1	2	2-1	- - - - ↕ ↕ 0 0
TSX	Transfer Stack Pointer to X	SP + 1→IX	INH	30		1	3	2-3	- - - - - - - -
TSY	Transfer Stack Pointer to Y	SP + 1→IY	INH	18 30		2	4	2-5	- - - - - - - -

* Cycle-by-cycle number provides a reference to Table 10.2 through 10.8 which detail cycle-by-cycle operation.
Example:Table 10.1 cycle-by-cycle column reference number 2.4 equals Table 10.2 line item 2.4.

Table 10.1 TMP68HC11E9 Instructions, Addressing Modes, and Execution Times (Sheet 7 of 7)

Source Form (s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes							
				Opcode	Operand (s)				S	X	H	I	N	Z	V	C
TXS	Transfer X to Stack Pointer	IX – 1→SP	INH	35		1	3	2-2	-	-	-	-	-	-	-	-
TYS	Transfer Y to Stack Pointer	IY – 1→SP	INH	18 35		2	4	2-4	-	-	-	-	-	-	-	-
WAI	Wait for Interrupt	Stack Regs & WAIT	INH	3E		1	***	2-16	-	-	-	-	-	-	-	-
XGDX	Exchange D with X	IX→D, D→IX	INH	8F		1	3	2-2	-	-	-	-	-	-	-	-
XGDY	Exchange D with Y	IY→D, D→IY	INH	18 8F		2	4	2-4	-	-	-	-	-	-	-	-

* Cycle-by-cycle number provides a reference to Table 10.2 through 10.8 which detail cycle-by-cycle operation.
Example:Table 10.1 cycle-by-cycle column reference number 2.4 equals Table 10.2 line item 2.4.

** Infinity or Until Reset Occurs

*** 12 Cycles are used beginning with the opcode fetch. A wait state is entered which remains in effect for an integer number of MPUE-clock cycles (n) until an interrupt is recognized. Finally, two additional cycles are used to fetch the appropriate interrupt vector (14 + n total).

dd = 8-Bit Direct Address ($0000-$00FF) (High Byte Assumed to be $00)
ff = 8-Bit Positive Offset $00 (0) to $FF (255) (Is Added to Index)
hh = High Order Byte of 16-Bit Extended Address
ii = One Byte of Immediate Data
jj = High Order Byte of 16-Bit Immediate Data
kk = Low Order Byte of 16-Bit Immediate Data
ll = Low Order Byte of 16-Bit Extended Address
mm = 8-Bit Mask (Set Bits to be Affected)
rr = Signed Relative Offset $80 (– 128) to $7F (+ 127)
(Offset Relative to the Address Following the Machine Code Offset Byte)

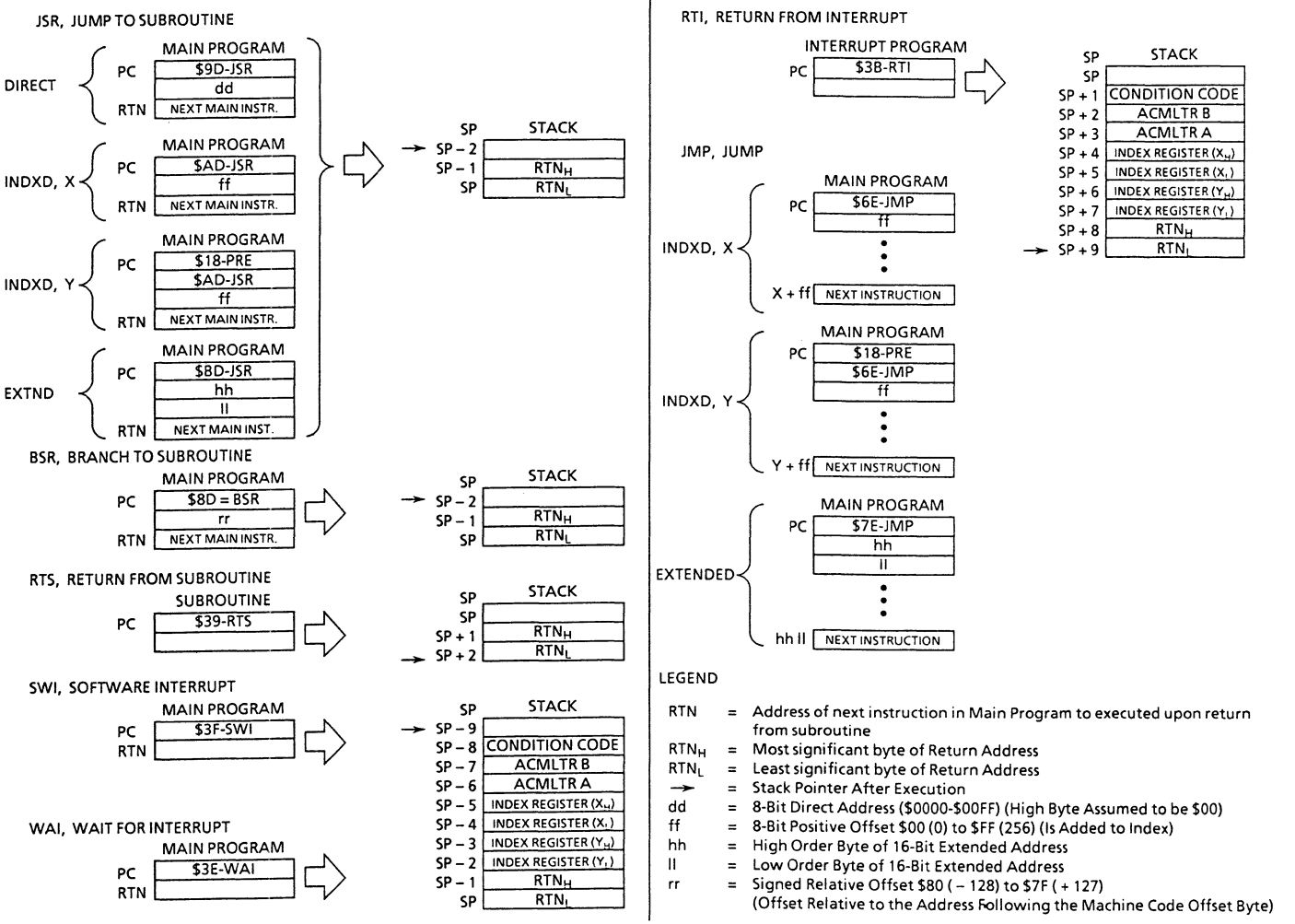


Figure 10.2 Special Operations

Table 10.2 Cycle-by-Cycle Operation — Inherent Mode (Sheet 1 of 4)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
2-1	ABA, ASLA, ASLB, ASRA, ASRB, CBA, CLC, CLI, CLRA, CLRB, CLV, COMA, COMB, DAA, DECA, DECB, INCA, INCB, LSLA, LSLB, LSRA, LSRB, NEGA, NEGB, NOP, ROLA, ROLB, RORA, RORB, SBA, SEC, SEI, SEV, STOP, TAB, TAP, TBA, TPA, TSTA, TSTB	2	1 2	Opcode Address Opcode Address + 1	1 1	Opcode Irrelevant Data
2-2	ABX, ASLD, DEX, INX, LSLD, LSRD, TXS, XGDX	3	1 2 3	Opcode Address Opcode Address + 1 $FFFF	1 1 1	Opcode Irrelevant Data Irrelevant Data
2-3	DES, INS, TSX	3	1 2 3	Opcode Address Opcode Address + 1 Previous SP Value	1 1 1	Opcode Irrelevant Data Irrelevant Data
2-4	ABY, DEY, INY, TYS, XGDY	4	1 2 3 4	Opcode Address Opcode Address + 1 Opcode Address + 2 $FFFF	1 1 1 1	Opcode (Page Select Byte) ($18) Opcode (Second Byte) Irrelevant Data Irrelevant Data
2-5	TSY	4	1 2 3 4	Opcode Address Opcode Address + 1 Opcode Address + 2 Stack Pointer	1 1 1 1	Opcode (Page Select Byte) ($18) Opcode (Second Byte) ($30) Irrelevant Data Irrelevant Data
2-6	PSHA, PSHB	3	1 2 3	Opcode Address Opcode Address + 1 Stack Pointer	1 1 0	Opcode Irrelevant Data Accumulator Data
2-7	PSHX	4	1 2 3 4	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer – 1	1 1 0 0	Opcode ($3C) Irrelevant Data IXL (Low Byte) to Stack IXH (High Byte) to Stack
2-8	PSHY	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Opcode Address + 2 Stack Pointer Stack Pointer – 1	1 1 1 0 0	Opcode (Page Select Byte) ($18) Opcode (Second Byte) ($3C) Irrelevant Data IXL (Low Byte) to Stack IXH (High Byte) to Stack

*The reference number is given to provide a cross-reference to Table 10.1.

Table 10.2 Cycle-by-Cycle Operation — Inherent Mode (Sheet 2 of 4)

Reference Number*	Address Mode and Instructions	Cycle	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
2-9	PULA, PULB	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	1	Irrelevant Data
			4	Stack Pointer + 1	1	Operand Data from Stack
2-10	PULX	5	1	Opcode Address	1	Opcode ($38)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	1	Irrelevant Data
			4	Stack Pointer + 1	1	IXH (High Byte) from Stack
			5	Stack Pointer + 2	1	IXL (Low Byte) from Stack
2-11	PULY	6	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($38)
			3	Opcode Address + 2	1	Irrelevant Data
			4	Stack Pointer	1	Irrelevant Data
			5	Stack Pointer + 1	1	IYH (High Byte) from Stack
			6	Stack Pointer + 2	1	IYL (Low Byte) from Stack
2-12	RTS	5	1	Opcode Address	1	Opcode ($39)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	1	Irrelevant Data
			4	Stack Pointer + 1	1	Address of Next Instruction (High Byte)
			5	Stack Pointer + 2	1	Address of Next Instruction (Low Byte)
2-13	MUL	10	1	Opcode Address	1	Opcode ($3D)
			2	Opcode Address + 1	1	Irrelevant Data
			3	$FFFF	1	Irrelevant Data
			4	$FFFF	1	Irrelevant Data
			5	$FFFF	1	Irrelevant Data
			6	$FFFF	1	Irrelevant Data
			7	$FFFF	1	Irrelevant Data
			8	$FFFF	1	Irrelevant Data
			9	$FFFF	1	Irrelevant Data
			10	$FFFF	1	Irrelevant Data
2-14	RTI	12	1	Opcode Adderess	1	Opcode ($3B)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	1	Irrelevant Data
			4	Stack Pointer + 1	1	Condition Code Register from Stack
			5	Stack Pointer + 2	1	B Accumulator from Stack
			6	Stack Pointer + 3	1	A Accumulator from Stack
			7	Stack Pointer + 4	1	IXH (High Byte) from Stack
			8	Stack Pointer + 5	1	IXL (Low Byte) from Stack
			9	Stack Pointer + 6	1	IYH (High Byte) from Stack

*The reference number is given to provide a cross-reference to Table 10.1.

Table 10.2 Cycle-by-Cycle Operation — Inherent Mode (Sheet 3 of 4)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
2-14 (Continued)	RTI	12	10	Stack Pointer + 7	1	IYL (Low Byte) from Stack
			11	Stack Pointer + 8	1	Address of Next Instruction (High Byte)
			12	Stack Pointer + 9	1	Address of Next Instruction (Low Byte)
2-15	SWI	14	1	Opcode Adderess	1	Opcode ($3F)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	0	Return Address (Low Byte)
			4	Stack Pointer – 1	0	Return Address (High Byte)
			5	Stack Pointer – 2	0	IYL (Low Byte) to Stack
			6	Stack Pointer – 3	0	IYH (High Byte) to Stack
			7	Stack Pointer – 4	0	IXL (Low Byte) to Stack
			8	Stack Pointer – 5	0	IXH (High Byte) to Stack
			9	Stack Pointer – 6	0	A Accumulator to Stack
			10	Stack Pointer – 7	0	B Accumulator to Stack
			11	Stack Pointer – 8	0	Condition Code Register to Stack
			12	Stack Pointer – 8	1	Irrelevant Data
			13	Address of SWI Vector (First Location)	1	SWI Service Routine Address (High Byte)
			14	Address of Vector + 1 (Second Location)	1	SWI Service Routine Address (Low Byte)
2-16	WAI	14 + n	1	Opcode Adderess	1	Opcode ($3E)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	0	Return Address (Low Byte)
			4	Stack Pointer – 1	0	Return Address (High Byte)
			5	Stack Pointer – 2	0	IYL (Low Byte) to Stack
			6	Stack Pointer – 3	0	IYH (High Byte) to Stack
			7	Stack Pointer – 4	0	IXL (Low Byte) to Stack
			8	Stack Pointer – 5	0	IXH (High Byte) to Stack
			9	Stack Pointer – 6	0	A Accumulator to Stack
			10	Stack Pointer – 7	0	B Accumulator to Stack
			11	Stack Pointer – 8	0	Condition Code Register to Stack
			12 to n + 12	Stack Pointer – 8	1	Irrelevant Data
			n + 13	Address of Vector (First Location)	1	Service Routine Address (High Byte)
			n + 14	Address of Vector (Second Location)	1	Service Routine Address (Low Byte)
2-17	FDIV, IDIV	41	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Irrelevant Data
			3 – 41	$FFFF	1	Irrelevant Data

*The reference number is given to provide a cross-reference to Table 10.1.

Table 10.2 Cycle-by-Cycle Operation — Inherent Mode (Sheet 4 of 4)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
2-18	Page 1 Illegal Opcodes	15	1	Opcode Address	1	Opcode (Illegal)
			2	Opcode Address + 1	1	Irrelevant Data
			3	$FFFF	1	Irrelevant Data
			4	Stack Pointer	0	Return Address (Low Byte)
			5	Stack Pointer – 1	0	Return Address (High Byte)
			6	Stack Pointer – 2	0	IYL (Low Byte) to Stack
			7	Stack Pointer – 3	0	IYH (High Byte) to Stack
			8	Stack Pointer – 4	0	IXL (Low Byte) to Stack
			9	Stack Pointer – 5	0	IXH (High Byte) to Stack
			10	Stack Pointer – 6	0	A Accumulator
			11	Stack Pointer – 7	0	B Accumulator
			12	Stack Pointer – 8	0	Condition Code Register to Stack
			13	Stack Pointer – 8	1	Irrelevant Data
			14	Address of Vector (First Location)	1	Service Routine Address (High Byte)
			15	Address of Vector + 1 (Second Location)	1	Service Routine Address (Low Byte)
2-19	Pages 2, 3, or 4 Illegal Opcodes	16	1	Opcode Address	1	Opcode (Legal Page Select)
			2	Opcode Address + 1	1	Opcode (Illegal Second Byte)
			3	Opcode Address + 2	1	Irrelevant Data
			4	$FFFF	1	Irrelevant Data
			5	Stack Pointer	0	Return Address (Low Byte)
			6	Stack Pointer – 1	0	Return Address (High Byte)
			7	Stack Pointer – 2	0	IYL (Low Byte) to Stack
			8	Stack Pointer – 3	0	IYH (High Byte) to Stack
			9	Stack Pointer – 4	0	IXL (Low Byte) to Stack
			10	Stack Pointer – 5	0	IXH (High Byte) to Stack
			11	Stack Pointer – 6	0	A Accumulator
			12	Stack Pointer – 7	0	B Accumulator
			13	Stack Pointer – 8	0	Condition Code Register to Stack
			14	Stack Pointer – 8	1	Irrelevant Data
			15	Address of Vector (First Location)	1	Service Routine Address (High Byte)
			16	Address of Vector + 1 (Second Location)	1	Service Routine Address (Low Byte)
2-20	TEST	Infinite	1	Opcode Address	1	Opcode ($00)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Opcode Address + 1	1	Irrelevant Data
			4	Opcode Address + 2	1	Irrelevant Data
			5 – n	Previous Address + 1	1	Irrelevant Data

*The reference number is given to provide a cross-reference to Table 10.1.

Table 10.3 Cycle-by-Cycle Operation — Immediate Mode

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
3-1	ADCA, ADCB, ADDA, ADDB, ANDA, ANDB, BITA, BITB, CMPA, CMPB, EORA, EORB, LDAA, LDAB, ORAA, ORAB, SBCA, SBCB, SUBA, SUBB	2	1 2	Opcode Address Opcode Address + 1	1 1	Opcode Operand Data
3-2	LDD, LDS, LDX	3	1 2 3	Opcode Address Opcode Address + 1 Opcode Address + 2	1 1 1	Opcode Operand Data (High Byte) Operand Data (Low Byte)
3-3	ADDD, CPX, SUBD	4	1 2 3 4	Opcode Address Opcode Address + 1 Opcode Address + 2 $FFFF	1 1 1 1	Opcode Operand Data (High Byte) Operand Data (Low Byte) Irrelevant Data
3-4	LDY	4	1 2 3 4	Opcode Address Opcode Address + 1 Opcode Address + 2 Opcode Address + 3	1 1 1 1	Opcode (Page Select Byte) ($18) Opcode (Second Byte) ($EC) Operand Data (High Byte) Operand Data (Low Byte)
3-5	CPD, CPY	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Opcode Address + 2 Opcode Address + 3 $FFFF	1 1 1 1 1	Opcode (Page Select Byte) Opcode (Second Byte) Operand Data (High Byte) Operand Data (Low Byte) Irrelevant Data

*The reference number is given to provide a cross-reference to Table 10.1.

Table 10.4 Cycle-by-Cycle Operation — Direct Mode (Sheet 1 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
4-1	ADCA, ADCB, ADDA, ADDB, ANDA, ANDB, BITA, BITB, CMPA, CMPB, EORA, EORB, LDAA, LDAB, ORAA, ORAB, SBCA, SBCB, SUBA, SUBB	3	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	1	Operand Data
4-2	STAA, STAB	3	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	0	Data from Accumulator
4-3	LDD, LDS, LDX	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	1	Operand Data (High Byte)
			4	Operand Address + 1	1	Operand Data (Low Byte)
4-4	STD, STS, STX	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	0	Register Data (High Byte)
			4	Operand Address + 1	0	Register Data (Low Byte)
4-5	LDY	5	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($DE)
			3	Opcode Address + 2	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			4	Operand Address	1	Operand Data (High Byte)
			5	Operand Address + 1	1	Operand Data (Low Byte)
4-6	STY	5	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($DE)
			3	Opcode Address + 2	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			4	Operand Address	0	Register Data (High Byte)
			5	Operand Address + 1	0	Register Data (Low Byte)
4-7	ADDD, CPX, SUBD	5	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	1	Operand Data (High Byte)
			4	Operand Address + 1	1	Operand Data (Low Byte)
			5	$FFFF	1	Irrelevant Data

*The reference number is given to provide a cross-reference to Table 10.1.

Table 10.4 Cycle-by-Cycle Operation — Direct Mode (Sheet 2 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
4-8	JSR	5	1	Opcode Address	1	Opcode ($9D)
			2	Opcode Address + 1	1	Subroutine Address (Low Byte) (High Byte Assumed to be $00)
			3	Subroutine Address	1	First Subroutine Opcode
			4	Stack Pointer	0	Return Address (Low Byte)
			5	Stack Pointer – 1	0	Return Address (High Byte)
4-9	CPD, CPY	6	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			4	Operand Address	1	Operand Data (High Byte)
			5	Operand Address + 1	1	Operand Data (Low Byte)
			6	$FFFF	1	Irrelevant Data
4-10	BCLR, BSET	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	1	Original Operand Data
			4	Opcode Address + 2	1	Mask Byte
			5	$FFFF	1	Irrelevant Data
			6	Operand Address	0	Result Operand Data
4-11	BRCLR, BRSET	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	1	Original Operand Data
			4	Opcode Address + 2	1	Mask Byte
			5	Opcode Address + 3	1	Branch Offset
			6	$FFFF	1	Irrelevant Data

* The reference number is given to provide a cross-reference to Table 10.1.

Table 10.5 Cycle-by-Cycle Operation — Extended Mode (Sheet 1 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
5-1	JMP	3	1	Opcode Address	1	Opcode ($7E)
			2	Opcode Address + 1	1	Jump Address (High Byte)
			3	Opcode Address + 2	1	Jump Address (Low Byte)
5-2	ADCA, ADCB, ADDA, ADDB, ANDA, ANDB, BITA, BITB, CMPA, CMPB, EORA, EORB, LDAA, LDAB, ORAA, ORAB, SBCA, SBCB, SUBA, SUBB	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	1	Operand Data
5-3	STAA, STAB	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	0	Accumulator Data
5-4	LDD, LDS, LDX	5	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	1	Operand Data (High Byte)
			5	Operand Address + 1	1	Operand Data (Low Byte)
5-5	STD, STS, STX	5	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	0	Register Data (High Byte)
			5	Operand Address + 1	0	Register Data (Low Byte)
5-6	LDY	6	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($FF)
			3	Opcode Address + 2	1	Operand Address (High Byte)
			4	Opcode Address + 3	1	Operand Address (Low Byte)
			5	Operand Address	1	Operand Data (High Byte)
			6	Operand Address + 1	1	Operand Data (Low Byte)
5-7	STY	6	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($FF)
			3	Opcode Address + 2	1	Operand Address (High Byte)
			4	Opcode Address + 3	1	Operand Address (Low Byte)
			5	Operand Address	0	Register Data (High Byte)
			6	Operand Address + 1	0	Register Data (Low Byte)
5-8	ASL, ASR, CLR, COM, DEC, INC, LSL, LSR, NEG, ROL, ROR	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	1	Original Operand Data

*The reference number is given to provide a cross-reference to Table 10.1.

Table 10.5 Cycle-by-Cycle Operation — Extended Mode (Sheet 2 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
5-8 (Continued)	ASL, ASR, CLR, COM, DEC, INC, LSL, LSR, NEG, ROL, ROR	6	5	$FFFF	1	Irrelevant Data
			6	Operand Address	0	Result Operand Data
5-9	TST	6	1	Opcode Address	1	Opcode ($7D)
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	1	Original Operand Data
			5	$FFFF	1	Irrelevant Data
			6	$FFFF	1	Irrelevant Data
5-10	ADDD, CPX, SUBD	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	1	Operand Data (High Byte)
			5	Operand Address + 1	1	Operand Data (Low Byte)
			6	$FFFF	1	Irrelevant Data
5-11	CPD, CPY	7	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Operand Address (High Byte)
			4	Opcode Address + 3	1	Operand Address (Low Byte)
			5	Operand Address	1	Operand Data (High Byte)
			6	Operand Address + 1	1	Operand Data (Low Byte)
			7	$FFFF	1	Irrelevant Data
5-12	JSR	6	1	Opcode Address	1	Opcode ($BD)
			2	Opcode Address + 1	1	Subroutine Address (High Byte)
			3	Opcode Address + 2	1	Subroutine Address (Low Byte)
			4	Subroutine Address	1	First Opcode in Subroutine
			5	Stack Pointer	0	Return Address (Low Byte)
			6	Stack Pointer – 1	0	Return Address (High Byte)

* The reference number is given to provide a cross-reference to Table 10.1.

Table 10.6 Cycle-by-Cycle Operation — Indexed X Mode (Sheet 1 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
6-1	JMP	3	1	Opcode Address	1	Opcode ($6E)
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
6-2	ADCA, ADCB, ADDA, ADDB, ANDA, ANDB, BITA, BITB, CMPA, CMPB, EORA, EORB, LDAA, LDAB, ORAA, ORAB, SBCA, SBCB, SUBA, SUBB	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Operand Data
6-3	ASL, ASR, CLR, COM, DEC, INC, LSL, LSR, NEG, ROL, ROR	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Original Operand Data
			5	$FFFF	1	Irrelevant Data
			6	(IX) + Offset	0	Result Operand Data
6-4	TST	6	1	Opcode Address	1	Opcode ($6D)
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Original Operand Data
			5	$FFFF	1	Irrelevant Data
			6	$FFFF	1	Irrelevant Data
6-5	STAA, STAB	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	0	Accumulator Data
6-6	LDD, LDS, LDX	5	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Operand Data (High Byte)
			5	(IX) + Offset + 1	1	Operand Data (Low Byte)
6-7	LDY	6	1	Opcode Address	1	Opcode (Page Select Byte) ($1A)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($EE)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IX) + Offset	1	Operand Data (High Byte)
			6	(IX) + Offset + 1	1	Operand Data (Low Byte)
6-8	STD, STS, STX	5	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data

*The reference number is given to provide a cross-reference to Table 10.1.

Table 10.6 Cycle-by-Cycle Operation — Indexed X Mode (Sheet 2 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
6-8 (Continued)	STD, STS, STX	5	4	(IX) + Offset	0	Register Data (High Byte)
			5	(IX) + Offset + 1	0	Register Data (Low Byte)
6-9	STY	6	1	Opcode Address	1	Opcode (Page Select Byte) ($1A)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($EF)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IX) + Offset	0	Register Data (High Byte)
			6	(IX) + Offset + 1	0	Register Data (Low Byte)
6-10	ADDD, CPX, SUBD	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Register Data (High Byte)
			5	(IX) + Offset + 1	1	Register Data (Low Byte)
			6	$FFFF	1	Irrelevant Data
6-11	CPD, CPY	7	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IX) + Offset	1	Register Data (High Byte)
			6	(IX) + Offset + 1	1	Register Data (Low Byte)
			7	$FFFF	1	Irrelevant Data
6-12	JSR	6	1	Opcode Address	1	Opcode ($AD)
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	First Opcode in Subroutine
			5	Stack Pointer	0	Return Address (Low Byte)
			6	Stack Pointer – 1	0	Return Address (High Byte)
6-13	BCLR, BSET	7	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Original Operand Data
			5	Opcode Address + 2	1	Mask Byte
			6	$FFFF	1	Irrelevant Data
			7	(IX) + Offset	0	Result Operand Data
6-14	BRCLR, BRSET	7	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Original Operand Data
			5	Opcode Address + 2	1	Mask Byte
			6	Opcode Address + 3	1	Branch Offset
			7	$FFFF	1	Irrelevant Data

* The reference number is given to provide a cross-reference to Table 10.1.

Table 10.7 Cycle-by-Cycle Operation — Indexed Y Mode (Sheet 1 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
7-1	JMP	4	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($6E)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
7-2	ADCA, ADCB, ADDA, ADDB, ANDA, ANDB, BITA, BITB, CMPA, CMPB, EORA, EORB, LDAA, LDAB, ORAA, ORAB, SBCA, SBCB, SUBA, SUBB	5	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Operand Data
7-3	ASL, ASR, CLR, COM, DEC, INC, LSL, LSR, NEG, ROL, ROR	7	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Original Operand Data
			6	$FFFF	1	Irrelevant Data
			7	(IY) + Offset	0	Result Operand Data
7-4	TST	7	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($6D)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Original Operand Data
			6	$FFFF	1	Irrelevant Data
			7	$FFFF	1	Irrelevant Data
7-5	STAA, STAB	5	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	0	Accumulator Data
7-6	LDD, LDS, LDX, LDY	6	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Operand Data (High Byte)
			6	(IY) + Offset + 1	1	Operand Data (Low Byte)
7-7	STD, STS, STX, STY	6	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data

* The reference number is given to provide a cross-reference to Table 10.1.

Table 10.7 Cycle-by-Cycle Operation — Indexed Y Mode (Sheet 2 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
7-7 (Continued)	STD, STS, STX, STY	6	5	(IY) + Offset	0	Register Data (High Byte)
			6	(IY) + Offset + 1	0	Register Data (Low Byte)
7-8	ADDD, CPD, CPX, CPY, SUBD	7	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Operand Data (High Byte)
			6	(IY) + Offset + 1	1	Operand Data (Low Byte)
			7	$FFFF	1	Irrelevant Data
7-9	JSR	7	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($AD)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	First Opcode in Subroutine
			6	Stack Pointer	0	Return Address (Low Byte)
			7	Stack Pointer – 1	0	Return Address (High Byte)
7-10	BCLR, BSET	8	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Original Operand Data
			6	Opcode Address + 3	1	Mask Byte
			7	$FFFF	1	Irrevant Data
			8	(IY) + Offset	0	Result Operand Data
7-11	BRCLR, BRSET	8	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Original Operand Data
			6	Opcode Address + 3	1	Mask Byte
			7	Opcode Address + 4	1	Branch Offset
			8	$FFFF	1	Irrelevant Data

* The reference number is given to provide a cross-reference to Table 10.1.

Table 10.8 Cycle-by-Cycle Operation – Relative Mode

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
8-1	BCC, BCS, BEQ, BGE, BGT, BHI, BHS, BLE, BLO, BLS, BLT, BMI, BNE, BPL, BRA, BRN, BVC, BVS	3	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Branch Offset
			3	$FFFF	1	Irrelevant Data
8-2	BSR	6	1	Opcode Address	1	Opcode ($8D)
			2	Opcode Address + 1	1	Branch Offset
			3	$FFFF	1	Irrelevant Data
			4	Subroutine Address	1	Opcode of Next Instruction
			5	Stack Pointer	0	Return Address (Low Byte)
			6	Stack Pointer – 1	0	Return Address (High Byte)

* The reference number is given to provide a cross-reference to Table 10.1.

11. ELECTRICAL SPECIFICATIONS

11.1 MAXIMUM RATINGS

Raing	Symbol	Value	Unit
Supply Voltage	V_{DD}	− 0.3 to + 7.0	V
Input Voltage	Vin	− 0.3 to + 7.0	V
Operating Temperature Range MC68HC11E9	T_A	T_L to T_H − 40 to 85	℃
Storage Temperature Range	T_{stg}	− 55 to 150	℃
Current Drain per Pin* Excluding VDD,VSS,VRH, and VRL	I_D	25	mA

* One pin at a time observing maximum power dissipation limits.

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. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (e.g., either GND or V_{DD}).

11.2 THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic 52-Pin Quad Pack (PLCC)	θ_{JA}	TBD	℃/W

11.3 POWER CONSIDERATIONS

The avarage chip-junction temperature, T_J, in ℃ can be obtained from:

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

where:

T_A = Ambient Temperature, ℃

θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, ℃/W

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

P_{INT} = $I_{DD} \times V_{DD}$, Watts - Chip Internal Power

$P_{I/O}$ = Power Dissipation on Input and Output Pins, Watts - User Determined

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

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

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

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

$$K = PD \cdot (TA + 273℃) + \theta_{JA} \cdot P_D^2 \quad (3)$$

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.

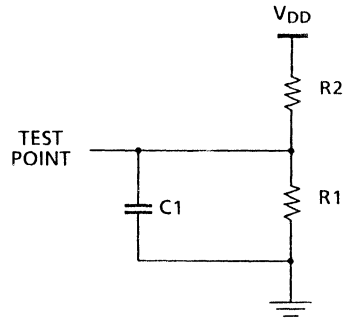
11.4 DC ELECTRICAL CHARACTERISTICS (V_{DD}=5.0 Vdc±10%,V_{SS}=0 Vdc,T_A=T_L to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Max	Unit
Output Voltage (see Note 1) All Outputs ILoad = ± 10.0μA All Outputs Except RESET and MODA	V_{OL} V_{OH}	– V_{DD} – 0.1	0.1 –	V
Output High Voltage All Outputs Except $\overline{RESET}$, I_{Load} = -0.8mA, V_{DD} = 4.5V (see Note 1) XTAL, and MODA	V_{OH}	V_{DD} – 0.8	–	V
Output Low Voltage All Outputs Except XTAL I_{Load} = 1.6mA	V_{OL}	–	0.4	V
Input High Voltage All Inputs Except $\overline{RESET}$ $\overline{RESET}$	V_{IH}	0.7 × V_{DD} 0.8 × V_{DD}	V_{DD} V_{DD}	V
Input Low Voltage All Inputs	V_{IL}	V_{SS}	0.2 × V_{DD}	V
I/O Ports, Three-State Leakage PA3, PA7, PC0-PC7, PD0-PD5, V_{in} = V_{IH} or V_{IL} AS/STRA, MODA $\overline{LIR}$, $\overline{RESET}$	I_{OZ}	–	± 10	μA
Input Current (see Note 2) V_{in} = V_{DD} or V_{SS} PA0-PA2, $\overline{IRQ}$ $\overline{XIRQ}$ V_{in} = V_{DD} or V_{SS} MODB/V_{STBY}	I_{in}	 – –	 ± 1 ± 10	μA
RAM Standby Voltage Powerdown	V_{SB}	4.0	V_{DD}	V
RAM Standby Current Powerdown	I_{SB}	–	20	μA
Total Supply Current (see Note 3) RUN: Single Chip Expanded Multiplexed WAIT: All Peripheral Functions Shut Down Single-Chip Mode Expanded Multiplexed Mode STOP: No Clocks, Single-Chip Mode	I_{DD} W_{IDD} S_{IDD}	 – – – – –	 15 27 6 10 100	 mA mA mA mA μA
Input Capacitance PA0-PA3, PE0-PE7, $\overline{IRQ}$, $\overline{XIRQ}$, EXTAL PA7, PC0-PC7, PD0-PD5, AS/STRA, MODA/$\overline{LIR}$, $\overline{RESET}$	C_{in}	– –	8 12	pF
Power Dissipation Single Chip Mode Expanded Multiplexed Mode	P_D	– –	85 150	mW

Notes :

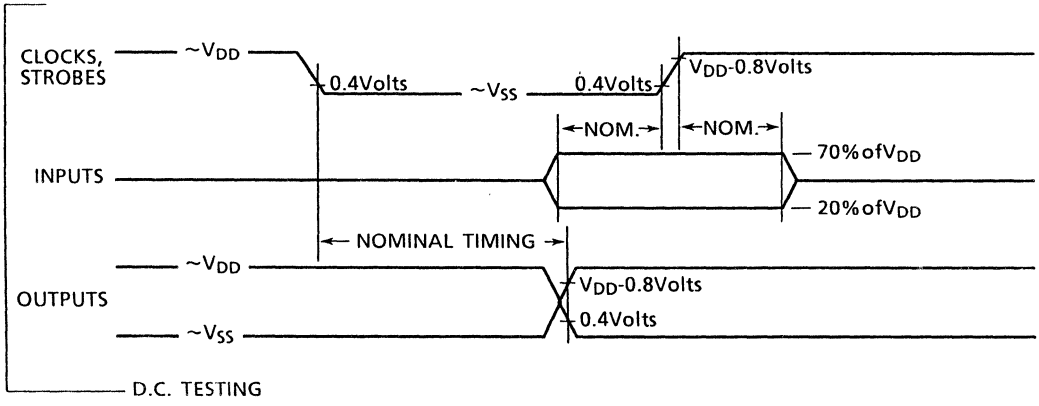
1. V_{OH} specification for $\overline{RESET}$ and MODA is not applicable because they are open-drain pins. V_{OH} specification not applicable to ports C and D in wire-OR mode.
2. See A/D specification for leakage current for port E.

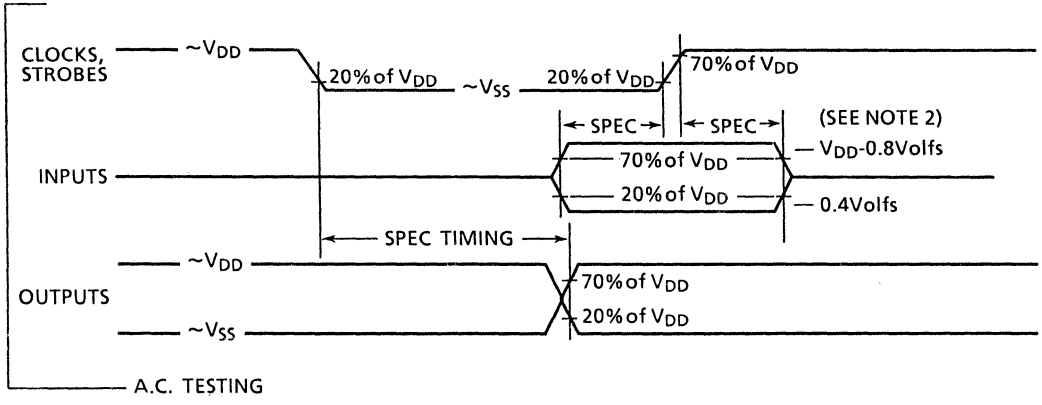
3. All ports configured as inputs,
   $V_{IL} \leqq 0.2$ V,
   $V_{IH} \geqq V_{DD} - 0.2$ V,
   No dc loads,
   EXTAL is driven with a square wave, and
   $t_{cyc} = 476.5$ ns.



Equivalent Test Load[1]

Pins	R1	R2	C1
PA3-PA7 PB0-PB7 PC0-PC7 PD0, PD5 E, AS, R/W	3.26KΩ	2.38KΩ	90pF
PD1-PD4	3.26KΩ	2.38KΩ	200pF





Notes:
1. Full test loads are applied during all dc electrical and ac timing measurements.
2. During ac timing measurements inputs are driven to 0.4 volts and V_{DD}-0.8 volts while timing measurements are taken at the 20% and 70% of V_{DD} points.

Figure 11.1 Test Methods

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

Characteristic	Symbol	1.0MHz		2.0MHz		2.1MHz		Unit
		Min	Max	Min	Max	Min	Max	
Frequency of Operation	fo	dc	1.0	dc	2.0	dc	2.1	MHz
E Clock Period	t_{cyc}	1000	–	500	–	476	–	ns
Crystal Frequency	f_{XTAL}	–	4.0	–	8.0	–	8.4	MHz
External Oscillator Frequency	$4 f_o$	dc	4.0	dc	8.0	dc	8.4	MHz
Processor Control Setup Time (see Figures 11.3, 11.5, and 11.6) $t_{PCS} = 1/4\ t_{cyc} - 50$ ns	t_{PCS}	200	–	75	–	69	–	ns
Reset Input Pulse Width (see Note 1 and Figure 11.3) (To Guarantee External Reset Vector)	PW_{RSTL}	8	–	8	–	8	–	t_{cyc}
(Minimum Input Time; May be Preempted by Internal Reset)		1	–	1	–	1	–	
Mode Programming Setup Time (see Figure 11.3)	t_{MPS}	2	–	2	–	2	–	ns
Mode Programming Hold Time (see Figure 11.3)	t_{MPH}	0	–	0	–	0	–	ns
Interrupt Pulse Width, $\overline{IRQ}$ Edge Sensitive Mode (see Figures 11.4 and 11.6) $PW_{IRQ} = t_{cyc} + 20$ ns	PW_{IRQ}	1020	–	520	–	496	–	ns
Wait Recovery Startup Time (See Figure 11.5)	t_{WRS}	–	4	–	4	–	4	t_{cyc}
Timer Pulse Width Input Capture, Pulse Accumulator Input (see Figure 11.2) $PW_{TIM} = t_{cyc} + 20$ ns	PW_{TIM}	1020	–	520	–	496	–	ns

Notes :

1. $\overline{RESET}$ will be recognized during the first clock cycle it is held low. Internal circuity then drives the pin low for four clock cycles, releases the pin, and samples the pin level two cycles later to detemine the source of the interrupt. See SECTION 9 RESETS, INTERRUPT, AND LOW POWER MODES for details.
2. All timing is shown with respect to 20% V_{DD} and 70% V_{DD} unless otherwise noted.

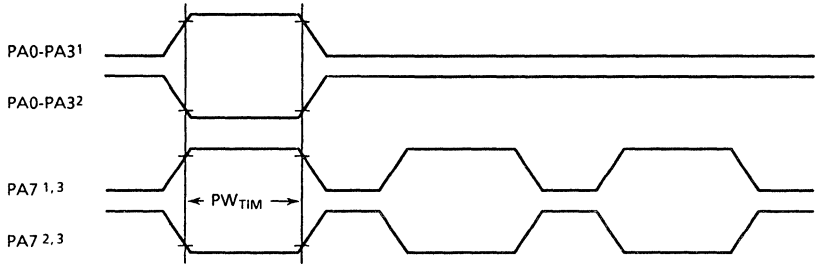
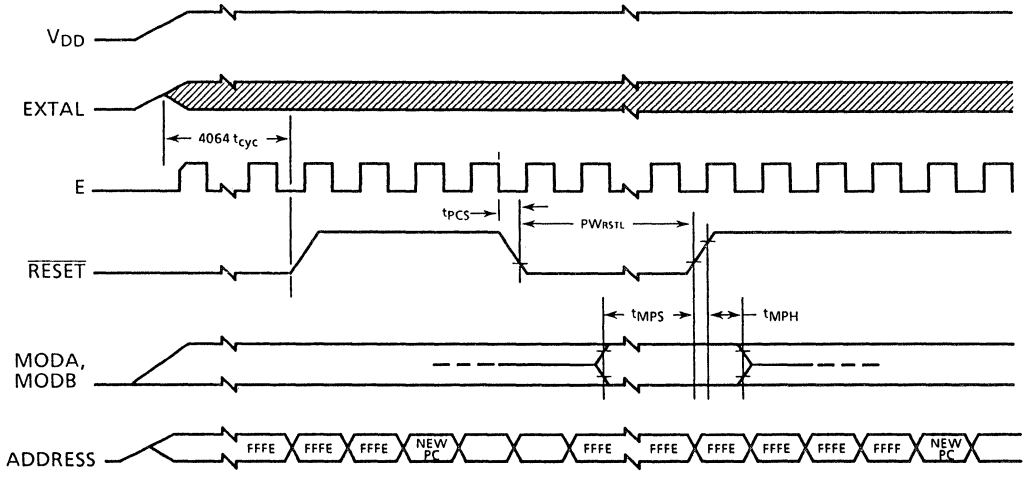


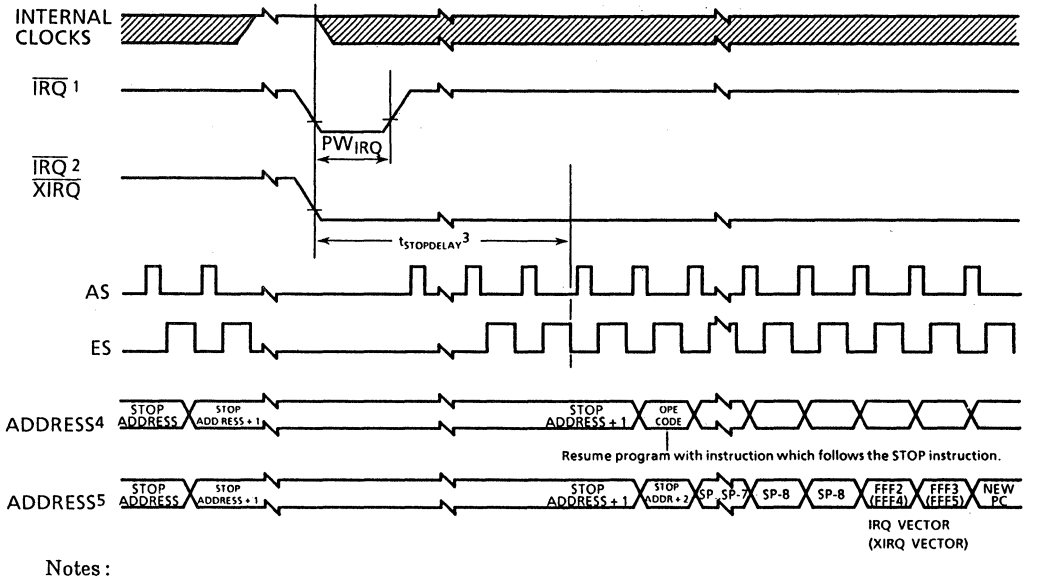
Figure 11.2 Timer Inputs Timing Diagram

Notes :
1. Rising edge sensitive input
2. Falling edge sensitive input
3. Maximum pulse accumulator clocking rate is E frequency divided by 2.



Note : Refer to Table 9.7 for pin states during $\overline{\text{RESET}}$

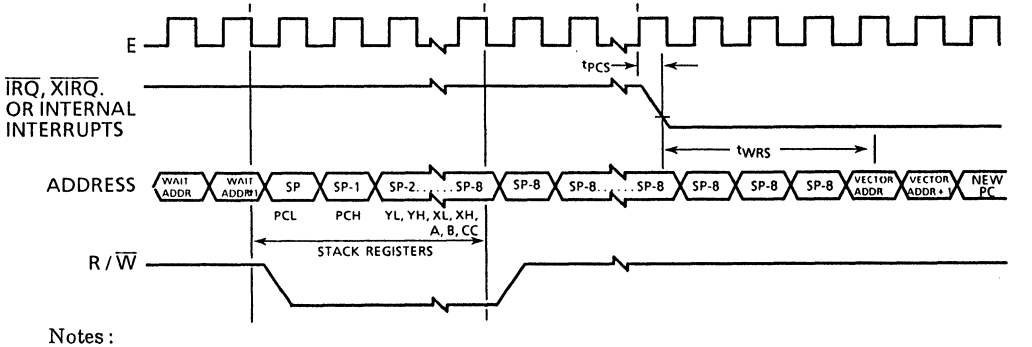
Figure 11.3 POR and External Reset Timing Diagram



Notes :

1. Edge sensitive $\overline{IRQ}$ pin (IRQE bit = 1)
2. Level sensitive $\overline{IRQ}$ pin (IRQE bit = 0)
3. tSTOPDELAY = 4064 tcyc if DLY bit = 1 or 4 tcyc if DLY = 0.
4. $\overline{XIRQ}$ WITH X bit CCR = 1 .
5. $\overline{IRQ}$, or ($\overline{XIRQ}$ with X bit in CCR = 0) .
6. Refer to Table 9.7 for pin states during STOP.

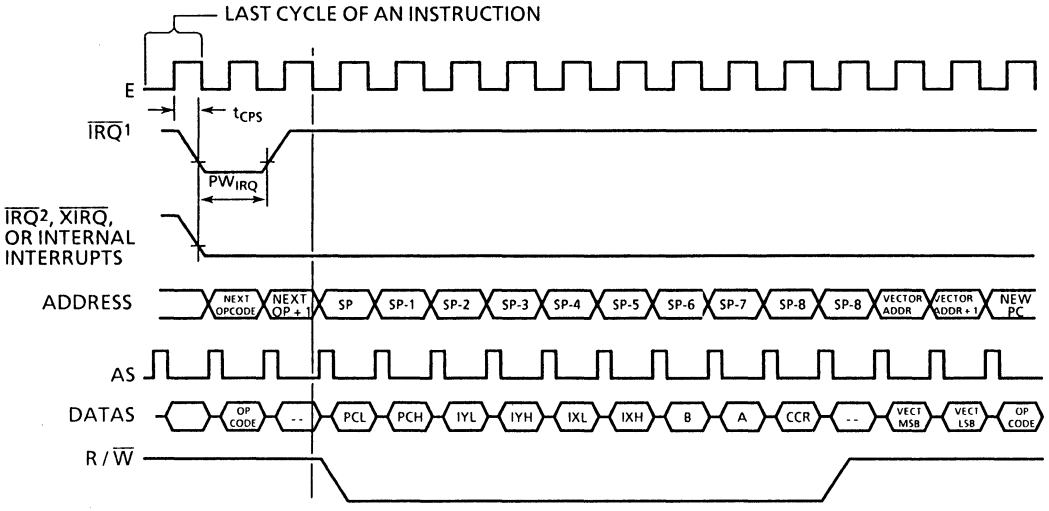
Figure 11.4 STOP Recovery Diagram



Notes :

1 Refer to Table 9.7 for pin states during WAIT.
2 $\overline{RESET}$ will also cause recovery from WAIT.

Figure 11.5 WAIT Recovery from Interrupt Timing Diagram



Notes: 1. Edge sensitive $\overline{IRQ}$ pin (IRQE bit = 1).
2. Level sensitive $\overline{IRQ}$ pin (IRQE bit = 0).

Figure 11.6 Interrupt Timing Diagram

11.6 PERIPHERAL PORT TIMING (V_{DD}=5.0 Vdc±10%, V_{SS}=0 Vdc, T_A=T_L to T_H)

Characteristic	Symbol	1.0MHz		2.0MHz		2.1MHz		Unit
		Min	Max	Min	Max	Min	Max	
Frequency of Operation(E Clock Frequency)	f_o	1.0	1.0	2.0	2.0	2.1	2.1	MHz
E Clock Period	t_{cyc}	1000	–	500	–	476	–	ns
Peripheral Data Setup Time (MCU Read of Ports A, C, D, and E) (see Figure 11.8)	t_{PDSU}	100	–	100	–	100	–	ns
Peripheral Data Hole Time (MCU Read of Ports A, C, D, and E) (see Figure 11.8)	t_{PDH}	50	–	50	–	50	–	ns
Delay Time, Peripheral Data Write (see Figures 11.7, 11.9, 11.12, and 11.13)	t_{PWD}							ns
MCU Write to Port A		–	150	–	150	–	150	
MCU Writes to Ports B,C, and D t_{PWD} = 1/4 t_{cyc} + 90 ns		–	340	–	215	–	209	
Input Data Setup Time(Port C) (see Figure 11.10 and 11.11)	t_{IS}	60	–	60	–	60	–	ns
Input Data Hold Time(Port C) (see Figure 11.10 and 11.11)	t_{IH}	100	–	100	–	100	–	ns
Delay Time, E Fall to STRB t_{DEB} = 1/4 t_{cyc} + 100 ns (see Figures 11.9, 11.11,11.12, and 11.13)	t_{DEB}	–	350	–	225	–	219	ns
Setup Time, STRA Asserted to E Fall(see Note 1) (see Figures 11.11, 11.12, and 11.13)	t_{AES}	0	–	0	–	0	–	ns
Delay Time, STRA Asserted to Port C Data Output Valid (see Figure 11.13)	t_{PCD}	–	100	–	100	–	100	ns
Hold Time, STRA Negated to Port C Data (see Figure 11.13)	t_{PCH}	10	–	10	–	10	–	ns
Three-State Hold Time (see Figure 11.13)	t_{PCZ}	–	150	–	150	–	150	ns

Notes :

1. If this setup time is met, STRB will acknowledge in the next cycle. If it is not met, the response may be delayed one more cycle.
2. Port and D timing is valed for active (CWOM and DWOM bits not set in PIOC and SPCR registers respectively).
3. All timing is shown with respect to 20% V_{DD} and 70% V_{DD} unless otherwise noted.

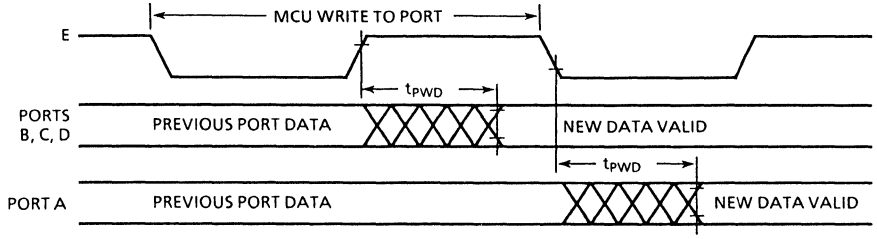
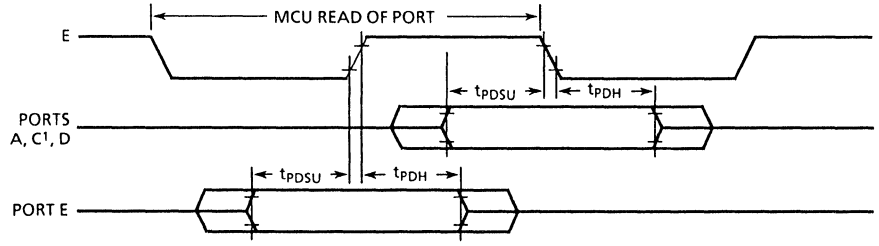


Figure 11.7 Port Write Timing Diagram



Note1 : For non-latched operation of Port C.

Figure 11.8 Port Read Timing Diagram

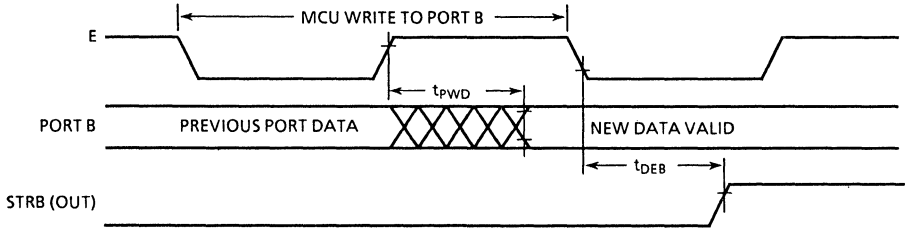


Figure 11.9 Simple Output Strobe Timing Diagram

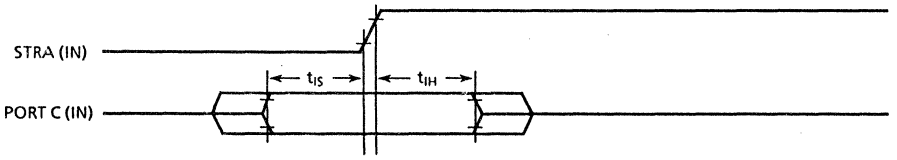
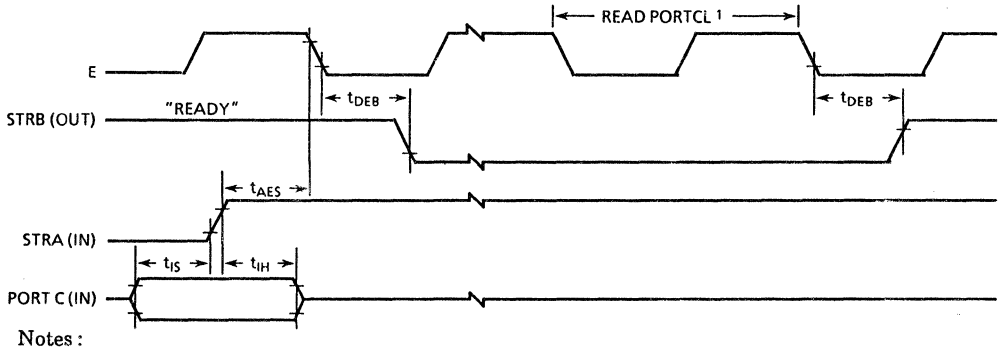
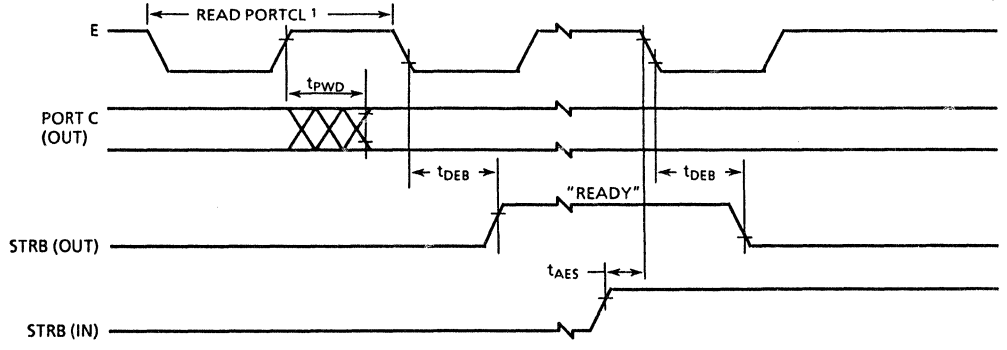


Figure 11.10 Simple Input Strobe Timing Diagram



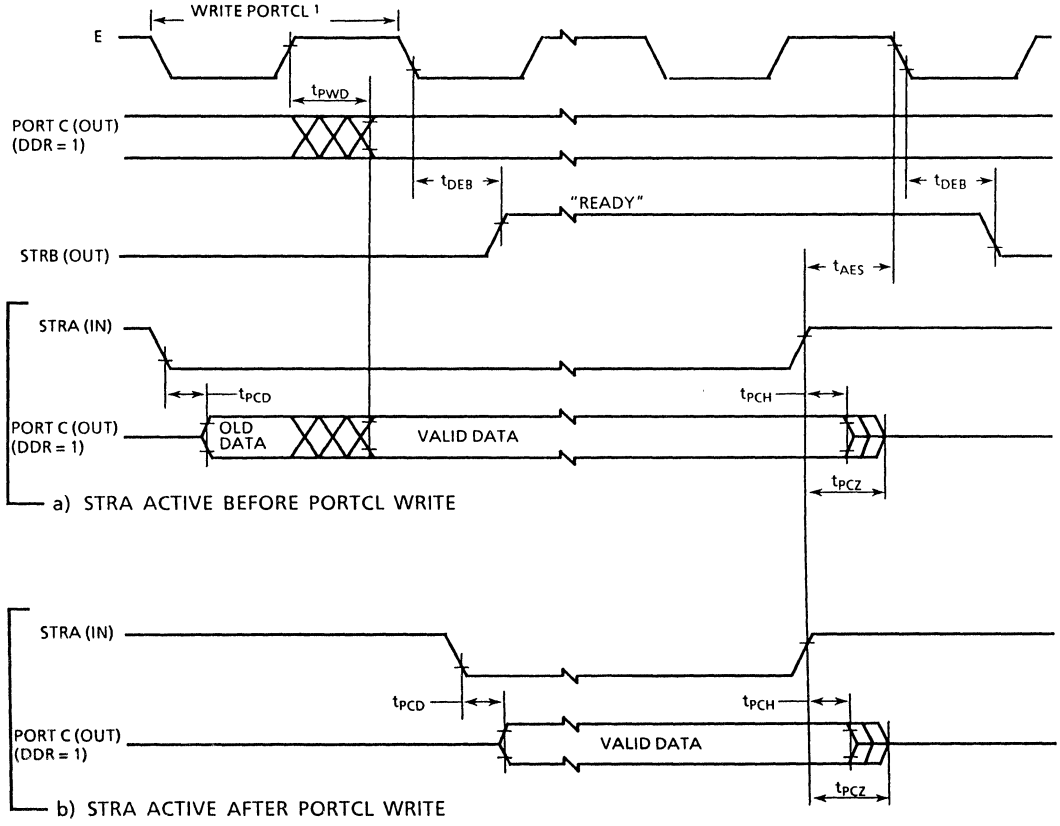
Notes :
1. After reading PIOC with STAF set.
2. Figure shows rising edge STRA (EGA = 1) and high true STRB (INVB = 1).

Figure 11.11 Port C Input Handshake Timing Diagram



Notes :
1. After reading PIOC with STAF set.
2. Figure shows rising edge STRA (EGA = 1) and high true STRB (INVB = 1).

Figure 11.12 Port C Output Handshake Timing Diagram



Notes :
1. After PIOC with STAF set.
2. Figure shows edge STRA (EGA = 1) and high true STRB (INVB = 1).

Figure 11.13 Three-State Variation of Output Handshake Timing Diagram (STRA Enables Output Buffer)

11.7 A/D CONVERTER CHARACTERISTICS (VDD = 5.0 Vdc ± 10%, VSS = 0Vdc, $T_A = T_L$ to T_H, 750kHz ≦ E ≦ 2.1MHz, unless otherwise noted)

Characteristic	Parameter	Min	Absolute	Max	Unit
Resolution	Number of Bits Resolved by the A/D	8	–	–	Bits
Non-Linearity	Maximum Deviation from the Ideal and an Actual A/D Transfer Characteristics	–	–	± 1/2	LSB
Zero Error	Difference Between the Output of an Ideal and an Actual A/D for Zero Input Voltage	–	–	± 1/2	LSB
Full Scale Error	Difference Between the Output of an Ideal A/D for Full-Scale Input Voltage	–	–	± 1/2	LSB
Total Unadjusted Error	Maximum Sum of Non-Linearity, Zero Error, and Full-Scale Error	–	–	± 1/2	LSB
Quantization Error	Uncertainty Due to Converter Resolution	–	–	± 1/2	LSB
Absolute Accuracy	Difference Between the Actual Input Voltage and the Full-Scale Weighted Equivalent of the Binary Output Code, All Error Sources Included	–	–	± 1	LSB
Conversion Range	Analog Input Voltage Range	V_{RL}	–	V_{RH}	V
V_{RH}	Maximum Analog Reference Voltage (see Note 2)	V_{RL}	–	V_{DD} + 0.1	V
V_{RL}	Maximum Analog Reference Voltage (see Note 2)	V_{SS} – 0.1	–	V_{RH}	V
ΔV_R	Maximum Difference between V_{RH} and V_{RL} (see Note 2)	3	–	–	V
Conversion Time	Total Time to Perform a Single Analog-to-Digital Conversion: a. E Clock b. Internal RC Oscillator	 – –	 32 –	 – tcyc + 32	 tcyc µs
Monotonicity	Conversion Result Never Decreases with an Increase in Input Voltage and has no Missing Codes		Guaranteed		
Zero Input Reading	Conversion Result when $V_{in} = V_{RL}$	00	–	–	Hex
Full Scale Reading	Conversion Result when $V_{in} = V_{RH}$	–	–	FF	Hex
Sample Acquisition Time	Analog Input Acquisition Sampling Time: a. E Clock b. Internal RC Oscillator	 – –	 12 –	 – 12	 tcyc µs
Sample/Hold Capacitance	Input Capacitance during Sample PE0-PE7	–	20 (Typ)	–	pF
Input Leakage	Input Leakage on A/D Pins PE0-PE7 V_{RL}, V_{RH}	– –	– –	400 1.0	nA µA

Notes:

1. Source impedances greater than 10KΩ will adversely affect accuracy, due mainly to input leakage.
2. Performance verified down to 2.5V ΔV_R, but accuracy is tested and guaranteed at $\Delta V_R = 5V \pm 10\%$.

11.8 EXPANSION BUS TIMING (V_{DD}=5.0 Vdc±10%, V_{SS}=0 Vdc, T_A=T_L to T_H, see Figure 11.14)

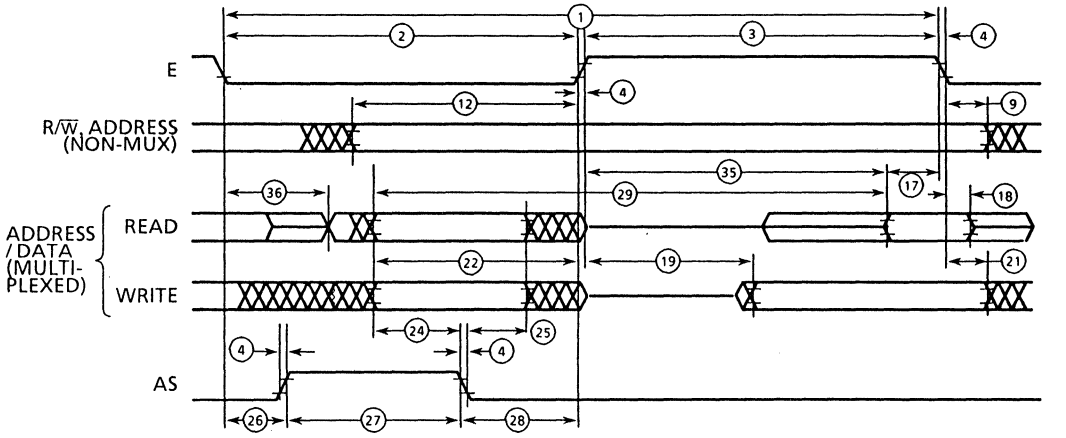
Num	Characteristic	Symbol	1.0MHz		2.0MHz		2.1MHz		Unit
			Min	Max	Min	Max	Min	Max	
	Frequency of Operation(E Clock Frequency)	f_o	1.0	1.0	2.0	2.0	2.1	2.1	MHz
1	Cycle Time	t_{cyc}	1000	–	500	–	476	–	ns
2	PUlse Width, E Low $PW_{EL} = 12t_{cyc} - 23$ ns	PW_{EL}	477	–	227	–	215	–	ns
3	PUlse Width, E High $PW_{EH} = 1/2t_{cyc} - 28$ ns	PW_{EH}	472	–	222	–	210	–	ns
4	E and AS Rise and fall Time	t_r, t_f	–	20	–	20	–	20	ns
9	Address Hold Time $t_{AV} = 1/8t_{cyc} - 29.5$ns see Note 1(a)	t_{AH}	95.5	–	33	–	30	–	ns
12	Non-Muxed Address Valid Time to E rise $t_{AV} = PW_{EL} - (t_{ASD} + 80$ ns) see Note 1(b)	t_{AV}	281.5	–	94	–	85	–	ns
17	Read Data Setup Time	t_{DSR}	30	–	30	–	30	–	ns
18	Read Data Hold Time(Max = t_{MAD})	t_{DHR}	10	145.5	10	83	10	80	ns
19	Write Data Delay Time $t_{DDW} = 1/8t_{cyc} + 65.5$ ns see Note 1(a)	t_{DDW}	–	190.5	–	128	–	125	ns
21	Write Data Hold Time $t_{DHW} = 1/8t_{cyc} - 29.5$ ns see Note 1(b)	t_{DHW}	95.5	–	33	–	30	–	ns
22	Muxed Address Valid Time to E Rise $t_{AVM} = PW_{EL} - (t_{ASD} + 90$ ns) see Note 1(b)	t_{AVM}	271.5	–	84	–	75	–	ns
24	Muxed Address Valid Time to As Fall $t_{AVM} = PW_{ASH} - 70$ ns	t_{ASL}	151	–	26	–	20	–	ns
25	Muxed Address Hold Time $t_{AHL} = 1/8\ t_{cyc} - 29.5$ ns see Note 1(b)	t_{AHL}	95.5	–	33	–	30	–	ns
26	Delay Time, E to AS Rise $t_{ASD} = 1/8\ t_{cyc} - 9.5$ ns see Note 1(a)	t_{ASD}	115.5	–	53	–	50	–	ns
27	Pulse Width, AS High $PW_{ASH} = 1/4t_{cyc} - 29$ ns	PW_{ASH}	221	–	96	–	90	–	ns
28	Delay Time, AS to E Rise $t_{ASED} = 1/8t_{cyc} - 9.5$ ns see Note 1(b)	t_{ASED}	115.5	–	53	–	50	–	ns
29	MPU Address Access Time $t_{ACCA} = t_{AVM} + t_r + PW_{EH} - t_{DSR}$ see note 1(b)	t_{ACCA}	733.5	–	296	–	275	–	ns
35	MPU Access Time $t_{ACCE} = PW_{EH} - t_{DSR}$	t_{ACCE}	–	442	–	192	–	180	ns
36	Muxed Address Delay (Previous Cycle MPU Read) $t_{MAD} = t_{ASD} + 30$ ns see Note 1(a)	t_{MAD}	145.5	–	83	–	80	–	ns

Notes:

1. Input clocks with duty cycles other than 50% will affect bus performance. Timing parameters affected by input clock duty cycle are identified by (a) and (b). To recalculate the approximate bus timing values, substitute the follwing expressions in place of 1/8 t_{cyc} in the formulas where applicable:
   (a) (1 - DC)×1/4tcyc
   (b) DC×1/4tcyc
   Where:

DC is the decimal value of duty cycle percentage (high time)

2. All timing is shown with respect to 20% V_{DD} and 70% V_{DD} unless othewise noted.



Note : Measurement point shown are 20% and 70% V_{DD}.

Figure 11.14 Expansion Bus Timing Diagram

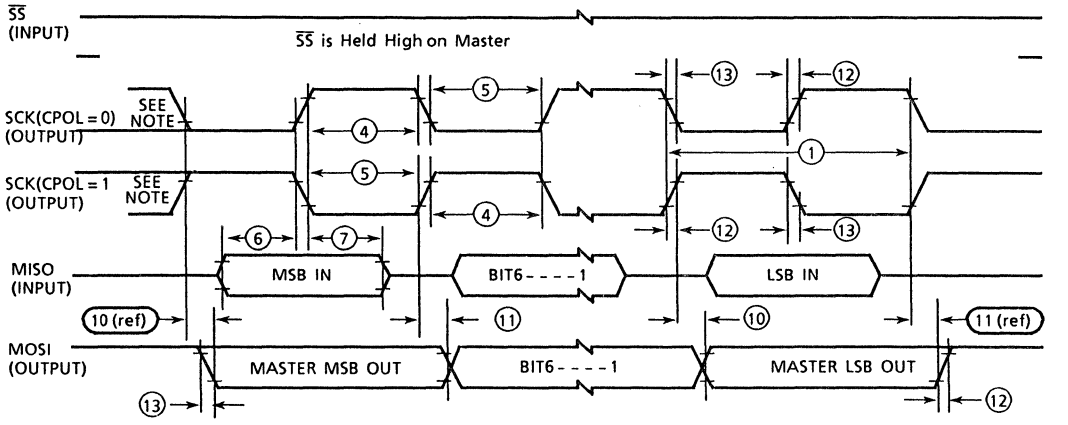
11.9 SERIAL PERIPHERAL INTERFACE (SPI) TIMING (V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, see Figure 11.15)

Num.	Characteristic	Symbol	Min	Max	Unit
	Operating Frequency Master Slave	 f_{OP} (m) f_{OP} (S)	 dc dc	 0.5 2.1	 fop MHz
1	Cycle Time Master Slave	 f_{cyc} (m) f_{cyc} (S)	 2.0 480	 – –	 t_{cyc} ns
2	Enable Lead Time Master Slave	 t_{lead} (m) t_{lead} (S)	 * 240	 – –	 ns ns
3	Enable Lag Time Master Slave	 t_{lag} (m) t_{lag} (S)	 * 240	 – –	 ns ns
4	Clock (SCK) High Time Master Slave	 t_w (SCKH) $_m$ t_w (SCKH) $_s$	 340 190	 – –	 ns ns
5	Clock (SCK) Low Time Master Slave	 t_w (SCKL) $_m$ t_w (SCKL) $_s$	 340 190	 – –	 ns ns
6	Data Setup Time (Inputs) Master Slave	 t_{su} (m) t_{su} (s)	 100 100	 – –	 ns ns
7	Data Hold Time (Inputs) Master Slave	 t_h (m) t_h (s)	 100 100	 – –	 ns ns
8	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 (After Enable Edge)**	t_v (S)	–	240	ns
11	Data Hold Time (Outputs) (After Enable Edge)	t_{ho}	0	–	ns
12	Rise Time (20% V_{DD} to 70% V_{DD}, C_L = 200pF) SPI Outputs(SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and $\overline{SS}$)	 t_{rm} t_{rs}	 – –	 100 2.0	 ns µs
13	Fall Time (70% V_{DD} to 20% V_{DD}, C_L = 200pF) SPI outputs(SCK, MOSI, and MISO) SPI Inputs(SCK, MOSI, MISO, and $\overline{SS}$)	 t_{fm} t_{fs}	 – –	 100 2.0	 ns µs

* Signal production depends on software.

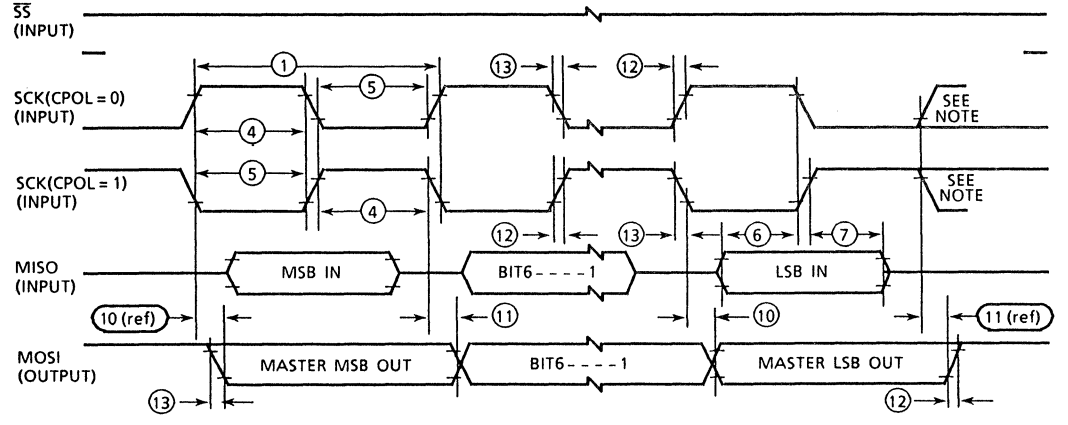
** Assumes 200 pF load on all SPI pins.

Note : All timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted.



Note : This first clock edge is generated intemally but is not seen at the SCK pin.

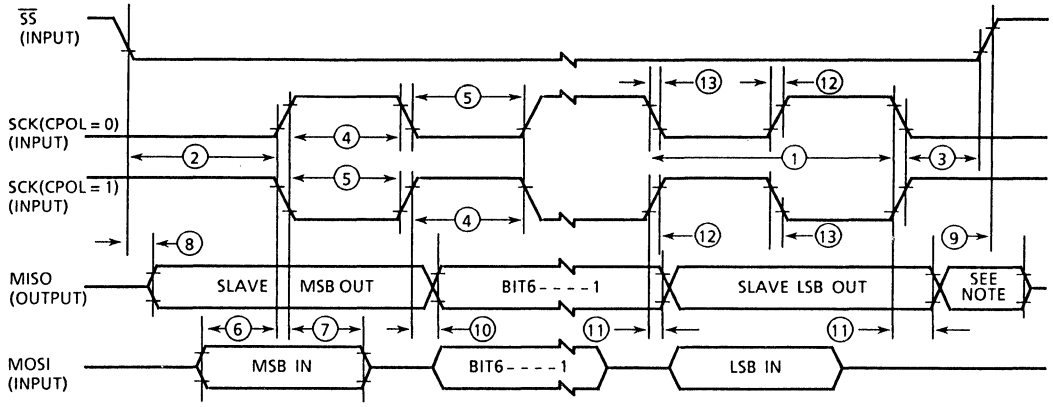
a) SPI MASTER TIMING (CPHA = 0)



Note : This last clock edge is generated intemally but is not seen at the SCK pin.

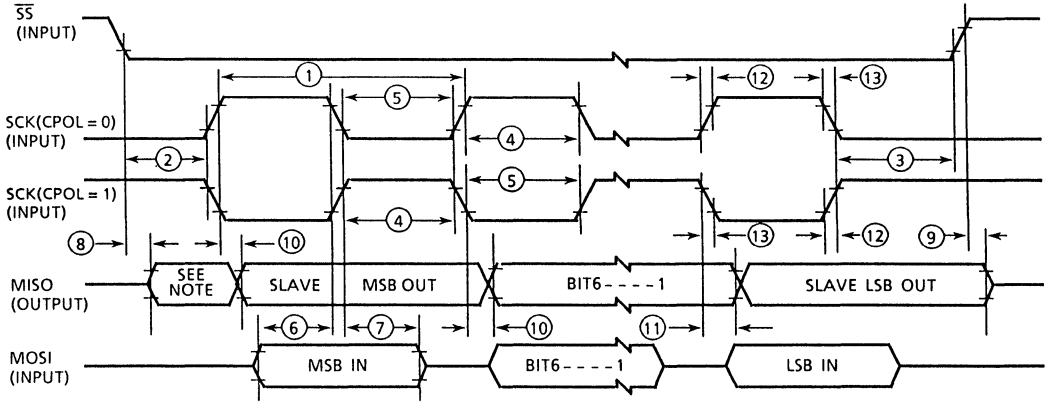
b) SPI MASTER TIMING (CPHA = 1)

Figure 11.15 SPI Timing Diagrams (Sheet 1 of 2)



Note : Not defined but normally MSB of character just received.

c) SPI SLAVE TIMING (CPHA = 0)



Note : Not defined but normally LSB of character previously transmitted.

d) SPI SLAVE TIMING (CPHA = 1)

Figure 11.15 SPI Timing Diagrams (Sheet 2 of 2)

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

Characteristic	Temperature Range			Unit
	– 40 to 85°C	– 40 to 105°C	– 40 to 125°C	
Programming time (See Note 1) Under 1.0 MHz with RC Oscillator Enable 1.0 to 2.0 MHz with RC Oscillator Disabled 2.0 MHz (or Anytime RC Oscillator Enabled)	 10 20 10	 15 Must Use RC 15	 20 Must Use RC 20	ms
Erase Time (see Note 1) Byte, Row, and Bulk	10	10	10	ms
Write Erase Endurance (See Note 2)	10,000	10,000	10,000	Cycles
Data Retention (See Note 2)	10	10	10	Years

Notes:

1. The RC oscillator must be enabled (by setting the CSEL bit in the OPTION register) for EEPROM programming and erasure when the E-clock frequency is below 1.0 MHz.
2. See current quarterly Reliability Monitor Report for current failure rate information.

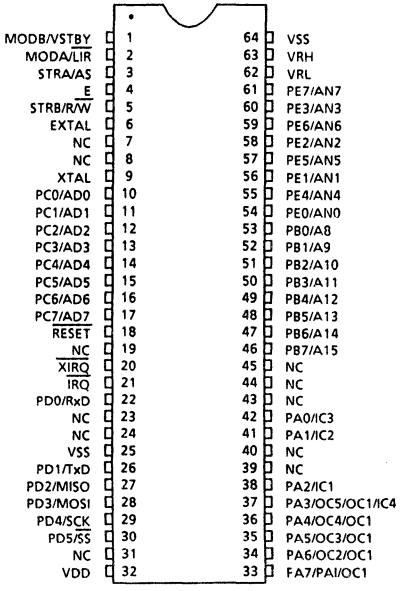
12. MECHANICAL DATA

This section contains the pin assignment and package dimension diagrams for the TMP68HC11E9.

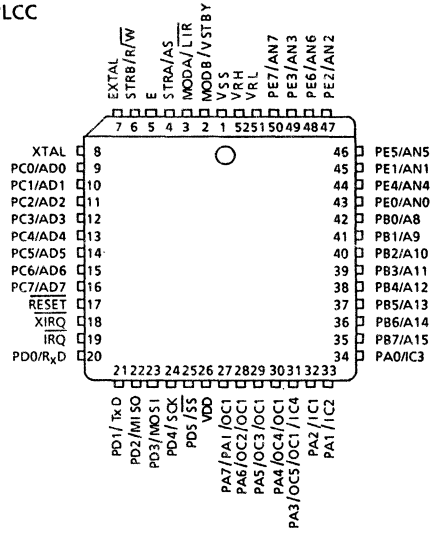
12.1 PIN ASSIGNMENTS

The TMP68HC11E9 is available in a 64-pin plastic shrink dual-in-line package (S-DIP) a 52-pin plastic lead chip carrier (PLCC) package. The following paragraph provide pin assignments.

N SUFFIX
64 PIN S-DIP



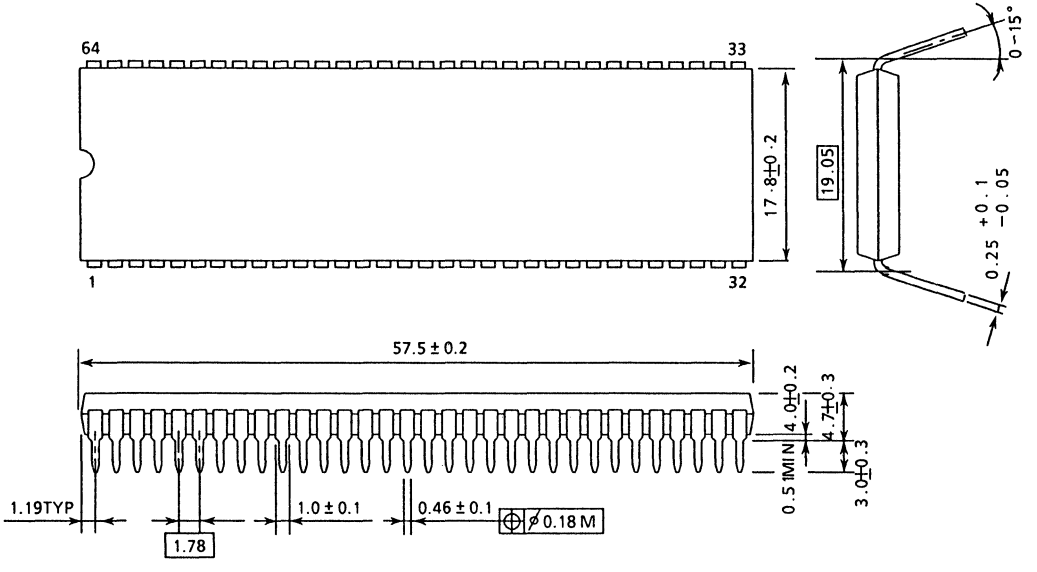
T SUFFIX
52 PIN PLCC



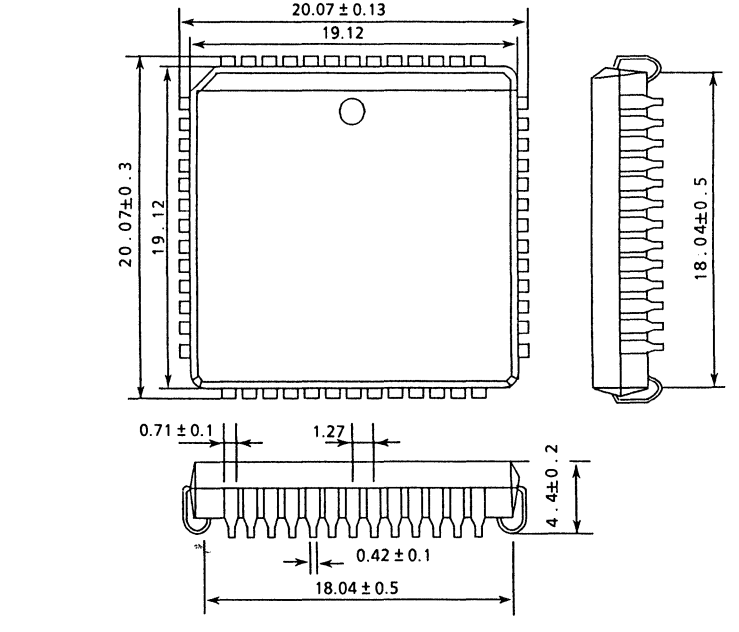
12.2 PACKAGE DIMENSIONS

N SUFFIX 64PIN S-DIP

(UNIT : mm)



T SUFFIX 52PIN PLCC



SECTION 4
TMP68C711J6

1. This technical data may be controlled under U.S. Export Administration Regulations and may be subject to the approval of the U.S. Department of Commerce prior to export. Any export or re-export directly or indirectly, in contravention of the U.S. Export Administration Regulations is strictly prohibited.

2. LIFE SUPPORT POLICY

   Toshiba products described in this databook are not authorized for use as critical components in life support systems without the written consent of the appropriate officer of Toshiba America, Inc. Life support systems are either systems intended for surgical implant in the body or systems which sustain life.

   A critical component is any component of a life support system whose failure to perform may cause a malfunction or failure of the life support system, or may affect its safety or effectiveness.

3. The information in this databook has been carefully checked and is believed to be reliable, however, no responsibility can be assumed for inaccuracies that may not have been caught. All information in this databook is subject to change without prior notice. Furthermore, Toshiba cannot assume responsibility for the use of any license under the patent rights of Toshiba or any third parties.

4. No part of this manual may be transferred or reproduced without prior permission of Toshiba Corporation.

CONTENTS

1. INTRODUCTION

This section depicts the general characteristics and special features of the TMP68C711J6 high-density complementary metal oxide semiconductor (HCMOS) microcontroller unit (MCU).

1.1 THE TOSHIBA TMP68C711J6 MCU

The TMP68C711J6 MCU contains highly sophisticated on-chip peripheral functions. This high-speed, low-power programmable read only memory (PROM) MCU has a nominal bus speed of 3.15 MHz. The fully static design allows operations at frequencies down to dc.

1.2 SPECIAL FEATURES

Refer to Figure 1-1 and the following list for hardware and software features of the TMP68C711J6:

* Expanded 16-Bit Timer System with Four-Stage Programmable Prescaler
* Enhanced Nonreturn-to-Zero (NRZ) Serial Communications Interface (SCI)
* Power Saving STOP and WAIT Modes Boundary
* 64K Memory Addressability
* Nonmultiplexed Address/Data Bus
* Serial Peripheral Interface (SPI)
* 16K Bytes of Erasable Programmable Read Only Memory (EPROM) or One Time Programmable Read Only Memory (OTPROM)
* 256 Byte Bootstrap ROM
* 8-Bit Pulse Accumulator Circuit
* 512 Bytes of Static RAM (All Saved During Standby)
* Real Time Interrupt Circuit
* Computer Operating Properly (COP) Watchdog System
* 68-Pin packaging with Seven I/O Ports for I/O Intensive Applications.

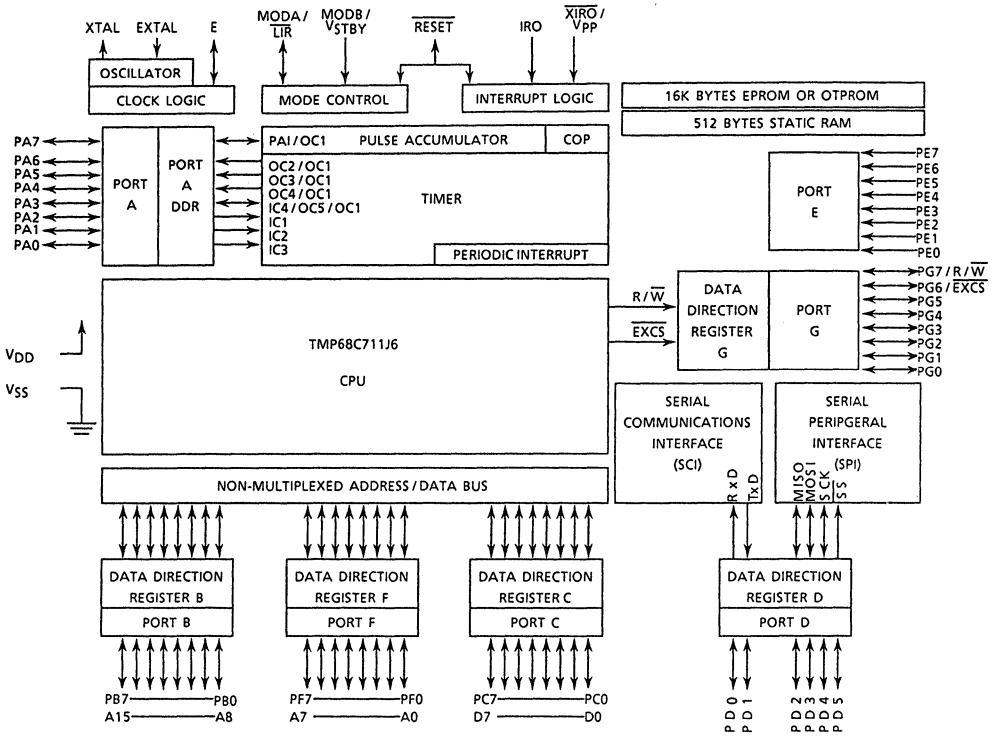


Figure 1.1 TMP68C711J6 Block Diagram

2. OPERATING MODES AND SIGNAL DESCRIPTIONS

This section describes the operating modes and signals of the TMP68C711J6.

2.1 OPERATING MODES

The TMP68C711J6 uses two dedicated pins, MODA and MODB, to select one of two normal operating modes or one of two special operating modes. A value reflecting the MCU status or mode selected is latched on bits SMOD and MDA of the HPRIO register on the rising edge of reset. The normal operating modes are the single-chip and expanded-nonmultiplexed modes. The special operating modes are the bootstrap and test modes. Mode selection according to the values encoded on the MODA and MODB pins, and the value latched in the SMOD and MDA bits are shown in the following table:

RESET	MODA	MODB	MODE SELECTED	SMOD	MDA
1	0	1	(Normal) Single-Chip	0	0
1	1	1	(Normal) Expanded-Nonmultiplexed	0	1
1	0	0	(Special) Bootstrap (BOOT / PROG)	1	0
1	1	0	(Special) Test	1	1
0	0	0	(Special) EPROM Emulation Mode	X	X

2.1.1 Single-Chip Mode

In the single-chip mode, the MCU functions as a self-contained microcontroller and has no external address or data bus. The 16K-byte PROM would contain all program code and is located at $C000-FFFF. This mode provides maximum use of the pins for on-chip peripheral functions, and all the address and data activity occurs within the MCU.

2.1.2 Expanded-Nonmultiplexed Mode

In the expanded-nonmultiplexed mode, the MCU can address up to 64K bytes of address space. High-order address bits are output on the port B pins, low-order address bits on the Port F pins, and the data bus is on port C. The R/$\overline{W}$ pin is used to control the direction of data transfer on the port C bus. The $\overline{EXCS}$ pin is an external chip elect which can be used directly or as a qualifier for decoding any external devices.

If this mode is entered out of reset, the PROM is located at $4000-7FFF and vector accesses are from external memory.

Note: R/$\overline{W}$, $\overline{EXCS}$, and address bus (port B, port F), are inputs in single-chip mode and may need to be pulled up so that off-chip accesses cannot occur while the MCU is in single-chip mode.

2.1.3 Bootstrap Mode (BOOT/PROG)

This special mode is similar to single-chip mode. The resident bootloader program contains a 256 byte program in a special on-chip ROM. The user downloads a small program into on-board RAM using the SCI port. Program control is passed to RAM when an idle line of at least four characters occurs. In this mode, all interrupt vectors are mapped to RAM see Table 2-1, so that the user can set up a jump table, if desired.

Bootstrap mode (BOOT) is entered out of reset if the voltage level on both MODA and MODB is low. The programming aspect of bootstrap mode, used to program the PROM (EPROM or OTPROM) through the MCU, is entered automatically if $\overline{IRQ}$ is low and programming voltage is available on the V_{PP} pin. $\overline{IRQ}$ should be pulled up while in reset with MODA and MODB configured for bootstrap mode to prevent unintentional programming of the PROM. The PROG mode, used for programming the MCU as though it were a standard 27256 type EPROM is entered by holding a low signal on the MODA, MODB, and $\overline{RESET}$ pins. See Section 7 for details on the PROG mode.

Table 2.1 Bootstrap Mode Jump Vectors

Address	Vector
00C4	SCI
00C7	SPI
00CA	Pulse Accumulator Input Edge
00CD	Pulse Accumulator Overflow
00D0	Timer Overflow
00D3	Timer Output Compare 5/Input Capture 4
00D6	Timer Output Compare 4
00D9	Timer Output Compare 3
00DC	Timer Output Compare 2
00DF	Timer Output Compare 1
00E2	Timer Input Capture 3
00E5	Timer Input Capture 2
00E8	Timer Input Capture 1
00EB	Real Time Interrupt
00EE	$\overline{IRQ}$
00F1	$\overline{XIRQ}$
00F4	SWI
00F7	Illegal Opcode
00FA	COP Fail
00FD	Clock Monitor
Start of Boot Code	Reset

This versatile mode (BOOT/PROG) can be used for test and diagnostic functions on completed modules, and for programming the onboard PROM. The serial receive logic is initialized by software in the bootloader ROM, which provides program control for the SCI baud rate and word format. Mode switching to other modes can occur under program control by writing to the SMOD and MDA bits of the HPRIO register. Two special bootloader functions allow either an immediate jump to RAM at memory address $0000 or an immediate jump to PROM at $C000.

2.1.4 Test Mode

This special expanded mode is primarily intended for production testing. The user can access a number of special test control bits in this mode. Reset and interrupt vectors are fetched externally from locations $BFC0-BFFF. A switch can be made from this mode to other modes under program control.

2.1.5 PROM Emulation Mode (PROG)

PROM emulation mode, the PROG aspect of bootstrap mode, is used for programming the MCU as though it were a standard 27256 type EPROM. This mode is entered by holding a low signal on the MODA, MODB, and $\overline{\text{RESET}}$ pins. A socket adapter is required for OTPROM or EPROM programming in this mode.

2.2 SIGNAL DESCRIPTION

The following paragraphs describe the signals necessary to the various functions of the MCU.

2.2.1 V_{DD} and V_{SS}

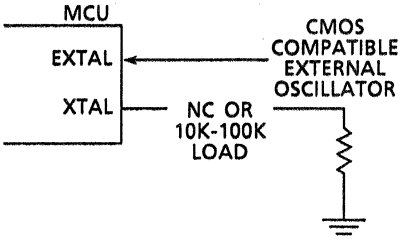
Power is supplied to the MCU using these pins. VDD is power (+5V ± 10%), VSS is ground (0V).

2.2.2 $\overline{\text{RESET}}$

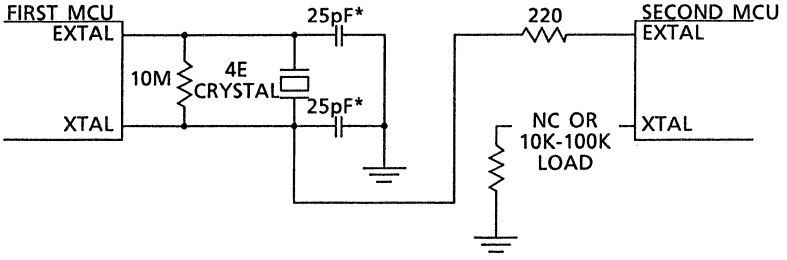
This active-low bidirectional control pin is used as an input to initialize the MCU to a known startup state. It is also used as an open-drain output to indicate that an internal failure has been detected in either the clock monitor or in the computer operating properly (COP) circuit. In addition, the state of this pin is one of the factors governing the selection of the BOOT/PROG mode.

2.2.3 XTAL and EXTAL

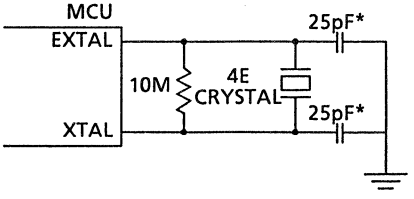
These pins provide the interface for either a crystal or a CMOS-compatible clock to control the internal clock generator circuitry. The frequency applied must be four times higher than the desired clock rate. Refer to Figure 2.1 for crystal and clock connections.



(a) External Oscillator Connections



(b) One Crystal Driving Two MCUs



(c) Common Crystal Connections

*Values include all stray capacitances.

Figure 2.1 Oscillator Connections

2.2.4 E Clock

This pin provides an output for the internally generated E clock, which can be used for timing reference. The frequency of the E clock output is one-fourth that of the input frequency at the XTAL and EXTAL pins.

2.2.5 $\overline{\text{IRQ}}$/CE

The $\overline{\text{IRQ}}$ pin provides the capability for asynchronously applying interrupts to the MCU. Either negative edge-sensitive triggering or level-sensitive triggering is program selectable by using the IRQE bit of the OPTION register. This pin is

configured as level sensitive during reset. While the PROM is being programmed and verified in PROG mode, this pin provides the Chip Enable ($\overline{CE}$) signal. An external resistor is required on $\overline{IRQ}$ to pull the pin to VDD to prevent accidental programming of the PROM during reset.

2.2.6 $\overline{XIRQ}/V_{PP}$

The $\overline{XIRQ}$ pin provides the capability for asynchronously applying nonmaskable interrupts to the MCU after a poweron reset (POR). During reset, the X bit in the condition code register is set, masking any interrupt until enabled by software. This level-sensitive input requires an external pullup resistor to VDD.

In the programming configuration of the bootstrap mode (PROG), this pin is used to supply EPROM or OTPROM programming voltage, VPP, to the MCU. To avoid programming accidents during reset, this pin should be kept in the VSS to VDD range during normal operation.

2.2.7 MODA/$\overline{LIR}$ and MODB/V_{STBY}

As reset transitions, these pins are used to latch the part into one of the four CPU controlled modes of operation. The $\overline{LIR}$ output can be used as an aid to debugging once reset is completed. The open-drain $\overline{LIR}$ pin goes to an active low during the first E clock cycle of each instruction and remains low for the duration of that cycle. The V_{STBY} input is used to retain RAM contents during power down.

2.2.8 R/$\overline{W}$

This pin provides two different functions, depending on the operating mode. In single-chip and bootstrap mode, the pin functions as input/output port G pin 7. In the expanded-nonmultiplexed and test modes, the pin provides the read-write (R/$\overline{W}$) function. R/$\overline{W}$ is used to control the direction of transfers on the external data bus.

2.2.9 $\overline{EXCS}$

This active low signal indicates that the address being accessed is external. It can be used to drive chip selects of off-chip devices in expanded and test modes. In the expanded-nonmultiplexed and test modes, it decodes addresses which are off-chip accesses.

2.2.10 Input/Output Lines (PA7-PA0, PB7-PB0, PC7-PC0, PD5-PD0, PE7-PE0, PF7-PF0, PG7-PG0)

In the 68-pin PLCC package, there are 54 input/output (I/O) lines which are arranged into six 8-bit ports, ports A, B, C, E, F, and G, and a single 6-bit port D. The lines of ports A, B, C, D, F and G are fully bidirectional under the

software control of their respective data direction registers. Port E is an input only port.

Many of these seven ports can serve a purpose other than simple I/O, depending on the operating mode or peripheral functions selected. Note that ports B, F, C, and two bits of port G are available for I/O only in single-chip and boot modes. Table 2-2 is a summary of pin functions to operating modes, by line and by port.

Table 2.2 Port Signal Functions

Port-Bit	Single-Chip and Bootstrap Mode	Expanded-Nonmultiplexed and Special Test Mode
A-0	PA0 / IC3	PA0 / IC3
A-1	PA1 / IC2	PA1 / IC2
A-2	PA2 / IC1	PA2 / IC1
A-3	PA3 / OC5 / IC4 (and / or OC1)	PA3 / OC5 / IC4 (and / or OC1)
A-4	PA4 / OC4 (and / or OC1)	PA4 / OC4 (and / or OC1)
A-5	PA5 / OC3 (and / or OC1)	PA5 / OC3 (and / or OC1)
A-6	PA6 / OC2 (and / or OC1)	PA6 / OC2 (and / or OC1)
A-7	PA7 / PAI (and / or OC1)	PA7 / PAI (and / or OC1)
B-0	PB0	A8
B-1	PB1	A9
B-2	PB2	A10
B-3	PB3	A11
B-4	PB4	A12
B-5	PB5	A13
B-6	PB6	A14
B-7	PB7	A15
C-0	PC0	D0
C-1	PC1	D1
C-2	PC2	D2
C-3	PC3	D3
C-4	PC4	D4
C-5	PC5	D5
C-6	PC6	D6
C-7	PC7	D7
D-0	PD0 / RxD	PD0 / RxD
D-1	PD1 / TxD	PD1 / TxD
D-2	PD2 / MISO	PD2 / MISO
D-3	PD3 / MOSI	PD3 / MOSI
D-4	PD4 / SCK	PD4 / SCK
D-5	PD5 / $\overline{SS}$	PD5 / $\overline{SS}$

Table 2.2 Port Signal Functions

Port-Bit	Single-Chip and Bootstrap Mode	Expanded-Nonmultiplexed and Special Test Mode
E-0	PE0	PE0
E-1	PE1	PE1
E-2	PE2	PE2
E-3	PE3	PE3
E-4*	PE4	PE4
E-5*	PE5	PE5
E-6*	PE6	PE6
E-7*	PE7	PE7
F-0	PF0	A0
F-1	PF1	A1
F-2	PF2	A2
F-3	PF3	A3
F-4	PF4	A4
F-5	PF5	A5
F-6	PF6	A6
F-7	PF7	A7
G-0	PG0	PG0
G-1	PG1	PG1
G-2	PG2	PG2
G-3	PG3	PG3
G-4	PG4	PG4
G-5	PG5	PG5
G-6	PG6	$\overline{EXCS}$
G-7	PG7	R / $\overline{W}$

* Not Bounded in 64-Pin Version

3. MEMORY AND CONTROL AND STATUS REGISTERS

This section describes the memory and the mapping of the control and status registers of the TMP68C711J6 MCU.

3.1 MEMORY

Figure 3.1 illustrates the memory map for both normal modes of operation (single-chip and expanded-nonmultiplexed), as well as for both special modes of operation (bootstrap and test modes). In the single-chip mode, the MCU does not generate external addresses. The internal memory locations are shown in the shaded areas, and the contents of these shaded areas are explained on the right side of the diagram. In the expanded-nonmultiplexed mode, the memory locations between $4000-$7FFF (shaded areas) are for externally addressed memory and I/O. The special bootstrap mode is similar to the single-chip mode, except that the bootstrap program ROM is located at memory locations $BF00 through $BFFF, vectors included. The special test mode is similar to the expanded-nonmultiplexed, except the interrupt vectors are at external memory locations.

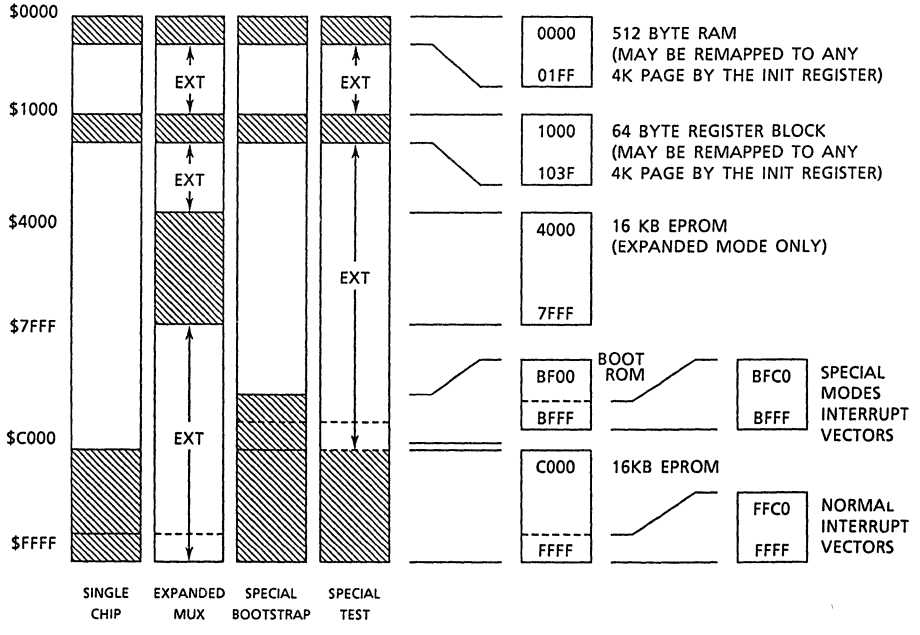


Figure 3.1 Memory Maps

Table 3.1 Register and Control Bit Assignments (Sheet 1 of 2)

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0		
$1000	Bit7	–	–	–	–	–	–	Bit0	PORTA	I/O Port A
$1001	Bit7	–	–	–	–	–	–	Bit0	DDRA	Data Direction for Port A
$1002	Bit7	–	–	–	–	–	–	Bit0	PORTF	I/O Port F
$1003	Bit7	–	–	–	–	–	–	Bit0	DDRF	Data Direction for Port F
$1004	Bit7	–	–	–	–	–	–	Bit0	PORTB	I/O Port B
$1005	Bit7	–	–	–	–	–	–	Bit0	DDRB	Data Direction for Port B
$1006	Bit7	–	–	–	–	–	–	Bit0	PORTC	I/O Port C
$1007	Bit7	–	–	–	–	–	–	Bit0	DDRC	Data Direction for Port C
$1008	0	0	Bit5	–	–	–	–	Bit0	PORTD	I/O Port D
$1009	0	0	Bit5	–	–	–	–	Bit0	DDRD	Data Direction for Port D
$100A	Bit7	–	–	–	–	–	–	Bit0	PORTE	Input Port E
$100B	FOC1	FOC2	FOC3	FOC4	FOC5	0	0	0	CFORC	Compare Force Register
$100C	OC1M7	OC1M6	OC1M5	OC1M4	OC1M3	0	0	0	OC1M	OC1 Action Mask Register
$100D	OC1D7	OC1D6	OC1D5	OC1D4	OC1D3	0	0	0	OC1D	OC1 Action Data Register
$100E	Bit15	–	–	–	–	–	–	Bit8	TCNT	Timer Counter Register
$100F	Bit7	–	–	–	–	–	–	Bit0		
$1010	Bit15	–	–	–	–	–	–	Bit8	TIC1	Input Capture 1 Register
$1011	Bit7	–	–	–	–	–	–	Bit0		
$1012	Bit15	–	–	–	–	–	–	Bit8	TIC2	Input Capture 2 Register
$1013	Bit7	–	–	–	–	–	–	Bit0		
$1014	Bit15	–	–	–	–	–	–	Bit8	TIC3	Input Capture 3 Register
$1015	Bit7	–	–	–	–	–	–	Bit0		
$1016	Bit15	–	–	–	–	–	–	Bit8	TOC1	Output Compare 1 Register
$1017	Bit7	–	–	–	–	–	–	Bit0		
$1018	Bit15	–	–	–	–	–	–	Bit8	TOC2	Output Compare 2 Register
$1019	Bit7	–	–	–	–	–	–	Bit0		
$101A	Bit15	–	–	–	–	–	–	Bit8	TOC3	Output Compare 3 Register
$101B	Bit7	–	–	–	–	–	–	Bit0		
$101C	Bit15	–	–	–	–	–	–	Bit8	TOC4	Output Compare 4 Register
$101D	Bit7	–	–	–	–	–	–	Bit0		
$101E	Bit15	–	–	–	–	–	–	Bit8	TI4O5	Output Compare 5 Register/
$101F	Bit7	–	–	–	–	–	–	Bit0		Input Capture 4 Register

Table 3.1 Register and Control Bit Assignments (Sheet 2 of 2)

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0		
$1020	OM2	OL2	OM3	OL3	OM4	OL4	OM5	OL5	TCTL1	Timer Control Register 1
$1021	EDG4B	EDG4A	EDG1B	EDG1A	EDG2B	EDG2A	EDG3B	EDG3A	TCTL2	Timer Control Register 2
$1022	OC1I	OC2I	OC3I	OC4I	I4O5I	IC1I	IC2I	IC3I	TMSK1	Timer Interrupt Mask Reg.1
$1023	OC1F	OC2F	OC3F	OC4F	I4O5F	IC1F	IC2F	IC3F	TFLG1	Timer Interrupt Flag Reg.1
$1024	TOI	RTII	PAOVI	PAII	0	0	PR1	PR0	TMSK2	Timer Interrupt Mask Reg.2
$1025	TOF	RTIF	PAOVF	PAIF	0	0	0	0	TFLG2	Timer Interrupt Flag Reg. 2
$1026	0	PAEN	PAMOD	PEDGE	0	I4/O5	RTR1	RTR0	PACTL	Pulse Accum.Control Reg.
$1027	Bit7	–	–	–	–	–	–	Bit0	PACNT	Pulse Accum.Count Reg.
$1028	SPIE	SPE	DWOM	MSTR	CPOL	CPHA	SPR1	SPR0	SPCR	SPI Control Register
$1029	SPIF	WCOL	0	MODF	0	0	0	0	SPSR	SPI Status Register
$102A	Bit7	–	–	–	–	–	–	Bit0	SPDR	SPI Data Register
$102B	TCLR	0	SCP1	SCP0	RCKB	SCR2	SCR1	SCR0	BAUD	SCI Baud Rate Control
$102C	R8	T8	0	M	WAKE	0	0	0	SCCR1	SCI Control Register 1
$102D	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK	SCCR2	SCI Control Register 2
$102E	TDRE	TC	RDRF	IDLE	OR	NF	FE	0	SCSR	SCI Status Register
$102F	Bit7	–	–	–	–	–	–	Bit0	SCDR	SDI Data (Read RDR, Write TDR)
$1030									Rsvd.	
to										
$1035									Rsvd.	
$1036	Bit7	–	–	–	–	–	–	Bit0	PORTG	I/O Port G
$1037	Bit7	–	–	–	–	–	–	Bit0	DDRG	Data Direction Port G
$1038	GWOM	GWOM	0	0	0	XCSEN	0	0	OPT2.	System configuration Options 2
$1039	0	0	IRQE	DLY	CME	0	CR1	CR0	OPTION	System configuration Options
$103A	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	COPRST	Arm/Reset COP Timer Circuity
$103B	0	0	ELAT	0	0	0	0	PGM	PPROG	EPROM Programming Control Register
$103C	RBOOT	SMOD	MDA	IRVNE	PSEL3	PSEL2	PSEL1	PSEL0	HPRIO	Highest Priority I Interrupt and Misc.
$103D	RAM3	RAM2	RAM1	RAM0	REG3	REG2	REG1	REG0	INIT	RAM asnd I/O Mapping Register
$103E	TILOP	EPTST	OCCR	CBYP	DISR	FCM	FCOP	0	TEST1	Factory Test Control Register
$103F	0	0	0	0	0	NOCOP	EPON	0	CONFIG	Configuration Control Register

3.2 RAM AND I/O MAPPING REGISTER (INIT)

The INIT register is a special-purpose 8-bit register that is used during initialization to change the default locations of RAM and control registers within the MCU memory map. It can be written to only once within the first 64 E clock cycles after a reset in normal modes. Thereafter it becomes a read-only register.

	7	6	5	4	3	2	1	0	
$103D	RAM3	RAM2	RAM1	RAM0	REG3	REG2	REG1	REG0	INIT
RESET:	0	0	0	0	0	0	0	1	

RAM3-RAM0 (INIT bits 7-4), specify the starting address for the 512 bytes of static RAM. REG3-REG0 (INIT bits 3-0) specify the starting address for the control and status register block. In each case, the four RAM or REG bits become the four upper bits of the 16-bit address of the RAM or registers. Since the INIT register is set to $01 by reset, the internal registers begin at $1000 and RAM begins at $0000.

Throughout this document, control and status register addresses are displayed with the high-order digit shown as a bold '1'. This convention indicates that the register block may be relocated to any 4K memory page, but that its default location is $1000.

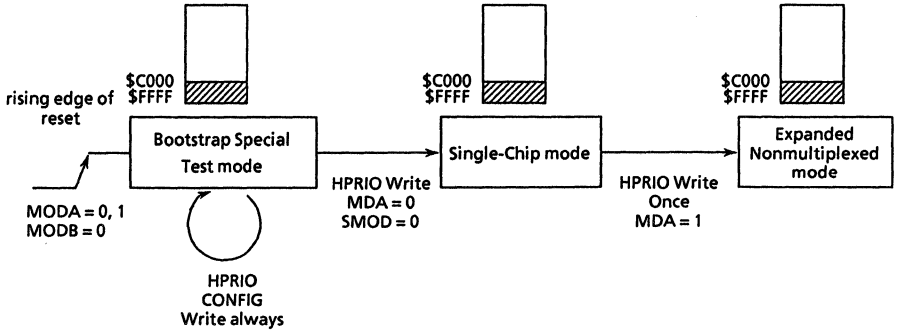
RAM and the control and status registers can be relocated independently. If the control and status registers are relocated in such a way as to conflict with PROM, then the register block takes priority, and the EPROM or OTPROM at those locations becomes inaccessible. No harmful conflicts result. Lower priority resources simply become inaccessible. Similarly, if an internal resource conflicts with an external device, no harmful conflict results, since data from the external device is not applied to the internal data bus. Thus, it cannot interfere with the internal read.

Note : There are unused register locations in the 64 byte control and status register block. Reads of these unused registers return data from the undriven internal data bus, not from another source that happens to be located at the same address.

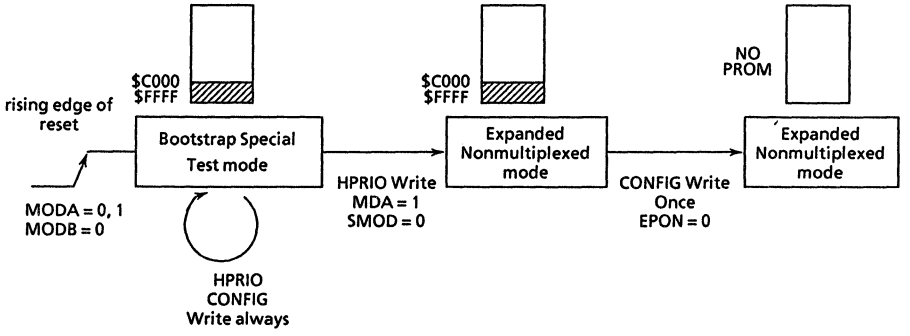
3.3 OPERATING MODES CHANGE

The TMP68C711J6 can change operating modes which construct various memory map, to change HPRIO and CONFIG registers. HPRIO and CONFIG registers describe Section 5 and 7. Cases of operating mode change are as follow.

- CASE-1



- CASE-2



- CASE-3

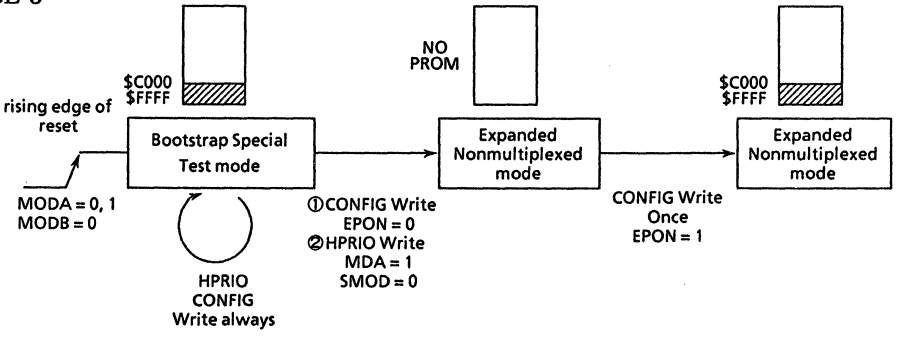
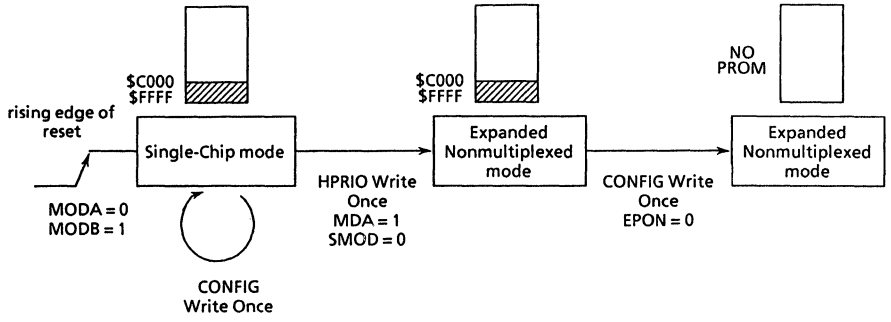


Figure 3.2 Operating Modes Change (Sheet 1 of 2)

- CASE-4



- CASE-5

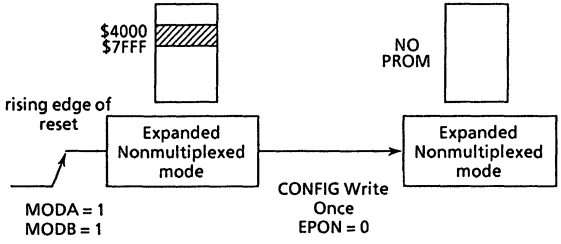


Figure 3.2 Operating Modes Change (Sheet 2 of 2)

4. INPUT/OUTPUT PORTS

The TMP68C711J6 is equipped with six 8-bit I/O ports (A, B, C, F, E and G), and a single 6-bit I/O port D. Port functions are controlled by the particular mode of operation selected, as shown in Table 2-2. In the single-chip and bootstrap modes, all the ports are configured as parallel I/O data ports. In expanded-nonmultiplexed and test modes, ports B, F, C, and lines G6 ($\overline{EXCS}$) and G7 (R/$\overline{W}$) are configured as a memory expansion bus. Port B is the high order address bus. Port F is the low order address bus. Port C is the data bus, and R/$\overline{W}$ and $\overline{EXCS}$ serve as bus control.

The remaining ports are unaffected by mode changes. Ports A and D can be used as general-purpose I/O ports, though each has an alternate function. Port A bits handle the timer functions. Port D handles the SPI and SCI functions in addition to its bus direction control functions. Port E is always an input only port.

4.1 PORT A

In both the normal operating modes, the port A data register (PORTA) can be configured for four timer input capture (IC) functions and three timer output compare (OC) functions, or for four OC and three IC functions, and either a pulse accumulator input (PAI) or a fifth OC output function.

4.1.1 Port A Data Register (PORTA)

	7	6	5	4	3	2	1	0	
$1000	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0	PORTA
RESET:	0	0	0	0	0	0	0	0	

Alternate Pin Functions:

PA1	OC2	OC3	OC4	OC5 / IC4	IC1	IC2	IC3
OC1	OC1	OC1	OC1	OC1			

PORTA can be read any time. Inputs return the pin level, whereas outputs return the pin driver input level. If written, PORTA stores the data in an internal latch. It drives the pins only if they are configured as outputs. Writes to PORTA do not change the pin state when the pins are configured for timer output compares.

Out of reset, all of the port A bits are general high impedance inputs. On bidirectional lines PA7 through PA3, the timer forces the I/O state to be an output if the associated output compare is enabled. In this case, data direction register A (DDRA) bits DDA7 through DDA3 will not be changed or have any effect on bits PA7-PA3. When the output compare functions associated with these pins are disabled, then DDRA again governs the PORTA I/O state.

4.1.2 PORT A Data Direction Register (DDRA)

	7	6	5	4	3	2	1	0	
$1001	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	DDRA
RESET	0	0	0	0	0	0	0	0	

1 = Corresponding Port A pin is configured as output.
0 = Corresponding Port A pin is configured for input only.

4.2 PORT B

Port B is an 8-bit, general purpose I/O port with a data register (PORTB), and a data direction register (DDRB). In the single-chip mode, port B pins are general-purpose I/O pins (PB7-PB0). In the expanded-nonmultiplexed mode, all of the port B pins act as the high-order address bits (A15-A8) of the address bus.

4.2.1 Port B Data Register (PORTB)

	7	6	5	4	3	2	1	0	
$1004	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0	PORTB
RESET:	0	0	0	0	0	0	0	0	
Alternate Pin Function:									
	A15	A14	A13	A12	A11	A10	A9	A8	

PORTB can be read at any time. Inputs return the sensed levels at the pin, while outputs return the input level of the port B pin drivers. If PORTB is written, the data is stored in an internal latch and can be driven only if port B is configured for general purpose outputs in single chip or bootstrap mode.

Port B pins are general purpose inputs out of reset in single chip and bootstrap modes. These pins are outputs (the high order address bits) out of reset in expanded nonmultiplexed and test modes.

In PROG mode, PORTB is high order address inputs.

4.2.2 Port B Data Direction Register (DDRB)

	7	6	5	4	3	2	1	0	
$1005	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	DDRB
RESET	0	0	0	0	0	0	0	0	

1 = Corresponding Port B pin is configured as output.
0 = Corresponding Port B pin is configured for input only.

4.3 PORT C

Port C is an 8-bit, general purpose I/O port with a data register (PORTC), and a data direction register (DDRC). In the single-chip mode, port C pins are general-purpose I/O pins (PC7-PC0). In the expanded-nonmultiplexed mode, port C pins are configured as bidirectional data pins controlled by the R / $\overline{W}$ signal.

4.3.1 Port C Data Register (PORTC)

	7	6	5	4	3	2	1	0	
$1006	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0	PORTC
RESET	0	0	0	0	0	0	0	0	
(PROG mode)	O7	O6	O5	O4	O3	O2	O1	O0	

PORTC can be read at any time. Inputs return the sensed levels at the pin, while outputs return the input level of the port C pin drivers. If PORTC is written, the data is stored in an internal latch and can be driven only if port C is configured for general-purpose outputs in single-chip or bootstrap mode.

Port C pins are general purpose inputs out of reset in single chip and bootstrap modes. These pins are data bus lines out of reset in expanded nonmultiplexed and test modes.

In PROM programming (PROG) mode this port is the data bus (O7-O0).

4.3.2 Port C Data Direction Register (DDRC)

	7	6	5	4	3	2	1	0	
$1007	DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	DDRC
RESET	0	0	0	0	0	0	0	0	

1 = Corresponding Port C pin is configured as output.
0 = Corresponding Port C pin is configured for input only.

4.4 PORT D

Port D is an 8-bit, general-purpose I/O port with a data register (PORTD) and a data direction register (DDRD). The eight port D bits (D7-D0) can be used for general purpose I/O, for the SCI and SPI subsystems, or for bus data direction control.

4.4.1 Port D Data Register (PORTD)

	7	6	5	4	3	2	1	0	
$1008	0	0	PD5	PD4	PD3	PD2	PD1	PD0	PORTD
RESET	0	0	0	0	0	0	0	0	
Alternate Pin Function:									
			$\overline{SS}$	SCK	MOSI	MISO	TxD	RxD	

PORTD can be read at any time and inputs return sensed levels at the pin, whereas outputs return the input level of the port D pin drivers. If PORTD is written, the data is stored in an internal latch, and can be driven only if port D is configured for general-purpose output. This port shares functions with the on-chip SCI and SPI subsystems, while bits 6 and 7 control the direction of data flow on the bus in expanded and special test modes.

4.4.2 Port D Data Direction Register (DDRD)

	7	6	5	4	3	2	1	0	
$1009	0	0	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	DDRD
RESET	0	0	0	0	0	0	0	0	

When port D is a general-purpose I/O port, the DDRD register controls the direction of the I/O pins as follows:

1 = Configures the corresponding port D pin for output.

0 = Configures the corresponding port D pin for input only

When port D is functioning with the SPI system enabled, bit 5 is dedicated as the slave select (SS8) input. In SPI slave mode, DDD5 has no meaning or effect. In SPI master mode, DDD5 affects port D bit 5 as follows:

1 = Port D bit 5 is configured as a general-purpose output line.

0 = Port D bit 5 is an error-detect input to the SPI.

If the SPI is enabled and expects port D bits 2, 3, and 4 (MISO, MOSI, and SCK) to be inputs, then they will be inputs, regardless of the state of DDRD bits 2, 3, and 4. If the SPI expects port D bits 2, 3, and 4 to be outputs, they can be outputs only if DDRD bits 2, 3, and 4 are set.

4.5 PORT E

Port E is an 8-bit, input only port with a data register (PORTE) input buffers are enabled only during an actual read cycle. The eight port E bits (PE7-PE0) can therefore be used for general purpose input only during an actual read cycle. Reads return the sensed levels at the pins, and writes have no meaning or effect.

4.5.1 Port E Data Register (PORTE)

	7	6	5	4	3	2	1	0	
$100A	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0	PORTE
RESET	*	*	*	*	*	*	*	*	

* Reset does not affect this address.

4.6 PORT F

Port F is an 8-bit, general purpose I/O port with a data register (PORTF), and a data direction register (DDRF). In the single-chip mode, port F pins are general-purpose I/O pins (PF7-PF0). In the expanded-nonmultiplexed mode, all of the port F pins act as the low-order address bits (A7-A0) of the address bus.

4.6.1 Port F Data Register (PORTF)

	7	6	5	4	3	2	1	0	
$1002	PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0	PORTF
RESET:	0	0	0	0	0	0	0	0	
Alternate Pin Function:									
	A7	A6	A5	A4	A3	A2	A1	A0	

PORTF can be read at any time. inputs return the sensed levels at the pin, while outputs return the input level of the port F pin drivers. If PORTF is written, the data is stored in an internal latch and can be driven only if port F is configured for general purpose outputs in single chip or bootstrap mode.

Port F pins are general purpose inputs out of reset in single chip and bootstrap modes. These pins are outputs (the low order address bits) out of reset in expanded nonmultiplexed and test modes. In PROG mode, port F is low order address inputs.

4.6.2 Port F Data Direction Register (DDRF)

	7	6	5	4	3	2	1	0	
$1003	DDF7	DDF6	DDF5	DDF4	DDF3	DDF2	DDF1	DDF0	DDRF
RESET	0	0	0	0	0	0	0	0	

1 = Corresponding Port F pin is configured as output.
0 = Corresponding Port F pin is configured for input only.

4.7 PORT G

Port G is an 8-bit, general purpose I/O port with a data register (PORTG), and a data direction register (DDRG). In the single-chip mode, port G pins are general-purpose I/O pins (PG7-PG0). In the expanded-nonmultiplexed mode, port G pins 7 and 6 are configured as nonmultiplexed address/data bus control lines R/$\overline{W}$ and $\overline{EXCS}$, while pins PG5-PG0 are still general purpose high impedance inputs.

4.7.1 Port G Data Register (PORTG)

	7	6	5	4	3	2	1	0	
$1036	PG7	PG6	PG5	PG4	PG3	PG2	PG1	PG0	PORTG
RESET	0	0	0	0	0	0	0	0	
Alternate Pin Function:									
	R/$\overline{W}$	$\overline{EXCS}$	PG5	PG4	PG3	PG2	PG1	PG0	

PORTG can be read at any time. Inputs return the sensed levels at the pin, while outputs return the input level of the port G pin drivers. If PORTG is written, the data is stored in an internal latch and can be driven only if port G is configured for general-purpose outputs in single-chip or bootstrap mode.

Port G pins are general purpose inputs out of reset in single chip and bootstrap modes. Pins PG7 and PG6 are bus control lines, while PG5-PG0 are inputs in expanded nonmultiplexed and test modes.

4.7.2 Port G Data Direction Register (DDRG)

	7	6	5	4	3	2	1	0	
$1037	DDG7	DDG6	DDG5	DDG4	DDG3	DDG2	DDG1	DDG0	DDRG
RESET	0	0	0	0	0	0	0	0	

1 = Corresponding Port G pin is configured as output.
0 = Corresponding Port G pin is configured for input only.

In expanded and test modes, R/$\overline{W}$ and $\overline{EXCS}$ functions override the state of bits DDG7 and DDG6.

4.8 SYSTEM CONFIGURATION OPTIONS 2 REGISTER (OPT2)

	7	6	5	4	3	2	1	0	
$1038	GWOM	CWOM	0	0	0	XCSEN	0	0	OPT2
RESET	0	0	0	0	0	*	0	0	

* Reset sets this bit in expanded and test modes, clears it in single-chip and bootstrap modes.

GWOM - Port G Wired-OR Mode

This bit affects all port G pins together.

1 = Port G outputs act as open-drain outputs.

0 = Port G outputs are normal CMOS outputs.

CWOM - Port C Wire-OR Mode

This bit affects all port C pins together.

1 = Port C outputs act as open-drain outputs.

0 = Port C outputs are normal CMOS outputs.

Bits 5, 4, 3, 1 and 0 - Not implemented.

XCSEN - External Chip Select Enable

1 = $\overline{EXCS}$ signal is enabled. In expanded and test modes only, the $\overline{EXCS}$ signal can be used to drive the chip selects of off-chip devices.

0 = $\overline{EXCS}$ is disabled. PG6 is general purpose I/O.

5. RESETS, INTERRUPTS, AND LOW POWER MODES

This section provides a description of the resets, interrupts, and low power modes. The computer operating properly (COP) watchdog system and clock monitor are described as part of the reset system. The interrupt description includes a flowchart to illustrate how interrupts are executed.

5.1 RESETS

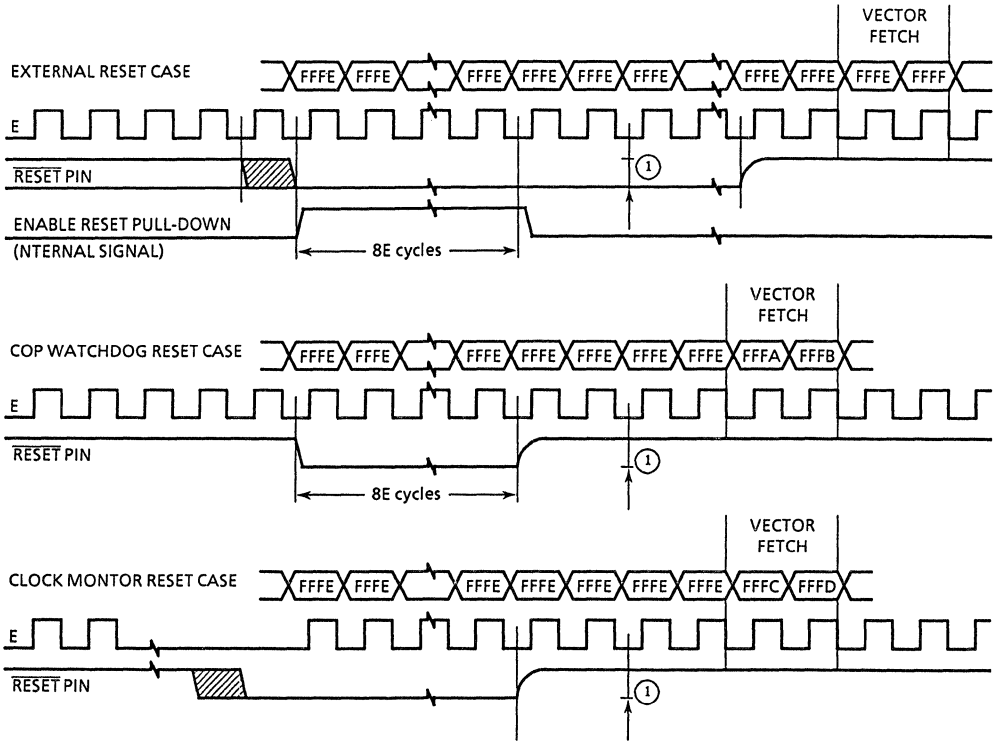
The MCU has four possible types of reset:an active low external reset pin ($\overline{\text{RESET}}$), a power-on reset, a computer operating properly (COP) watchdog timer reset, and a clock monitor reset.

5.1.1 External $\overline{\text{RESET}}$ Pin

The $\overline{\text{RESET}}$ pin is used to reset the MCU and allow an orderly software startup procedure. When a reset condition is sensed, this pin is driven low by an internal device for four E clock cycles, then released, and two E clock, cycles later it is sampled. If the pin is still low, it means that an external reset has occurred. If the pin is high, it implies that the reset was initiated internally by either the watchdog timer (COP) or the clock monitor (refer to Figure 5.1). This method of differentiation between internal and external reset conditions assumes that the reset pin will rise to a logic one in less than two E clock cycles once it is released and that an externally generated reset should stay active for at least twelve E clock cycles.

5.1.2 Power-On Reset

The power-on reset occurs when a positive transition is detected on V_{DD}. The power-on reset is used strictly for power turn-on conditions and should not be used to detect any drops in power supply voltage. The power-on circuitry provides a 4064 cycle time delay from the time of the first oscillator operation. In a system where E = 2 MHz, power on reset lasts about 2 milliseconds. If the external $\overline{\text{RESET}}$ pin is low at the end of the power-on delay time, the MCU remains in the reset condition until the $\overline{\text{RESET}}$ pin goes high.



Note (1) $\overline{\text{RESET}}$ pin is sampled at this time. Low implies an extemal reset. High implies clock monitor or COP watchdog caused reset.

Figure 5.1 Reset Timing

5.1.2.1 CPU.

After reset, the CPU fetches the restart vector from locations $FFFE and $FFFF($BFFE and $BFFF if in special bootstrap or special test operating mode) during the first three cycles, and begins executing instructions. The stack pointer and other CPU registers are indeterminate immediately after reset; however, the X and I interrupt mask bits in the condition code register are set to mask any interrupt requests. Also, the S bit in the condition code register is set to disable the STOP mode.

5.1.2.2 Memory Map.

After reset, the INIT register is initialized to $01, putting the 512 bytes of RAM at locations $0000 through $01FF and the control registers at locations $1000 through $103F.

5.1.2.3 Timer.

During reset, the timer system is initialized to a count of $0000. The prescaler bits are cleared, and all output compare registers are initialized to $FFFF. All input capture registers are indeterminate after reset. The output compare 1 mask (OC1M) register is cleared so that successful OC1 compares do not affect any I/O pins. The other four output compares are configured to not affect any I/O pins on successful compares. All three input capture edge-detector circuits are configured for "capture disabled" operation. The timer overflow interrupt flag and all eight timer function interrupt flags are cleared. All nine timer interrupts are disabled since their mask bits are cleared.

5.1.2.4 Real Time Interrupt.

The real time interrupt flag is cleared and automatic handware interrupts are masked. The rate control bits are cleared after reset and may be initialized by software before the real time interrupt system is used.

5.1.2.5 Pulse Accumulator.

The pulse accumulator system is disabled at reset so that the PAI input pin defaults to being a general purpose input pin.

5.1.2.6 COP.

The COP watchdog system is enabled if the NOCOP control bit in the system configuration control register is clear, and disabled if NOCOP is set. The COP rate is set for the shortest duration timeout.

- Single-Chip mode NOCOP=0
- Expanded-Nonmulti plexed mode NOCOP=0
- Special Boot Strap mode NOCOP=1
- Special Test mode NOCOP=1

5.1.2.7 SCI Serial I/O.

The reset condition of the SCI system is independent of the operating mode. At reset, the SCI baud rate is indeterminate and must be established by a software write to the BAUD register. All transmit and receive interrupts area masked and both the transmitter and receiver are disabled so the port pins default to being general purpose I/O lines. The SCI frame format is initialized to an 8-bit character size. The send break and receiver wake up functions are disabled. The TDRE and TC status bits in the SCI status register are both set, indicating that there is no transmit data in either the transmit data register of the transmit serial shift register. The RDRF, IDLE, OR NF, and FE receive-related status bits are all cleared.

Note that upon reset in special bootstrap mode execution begins in the 256 byte boot ROM. This firmware sets port D to wire-OR mode, establishes a baud rate, and enables the SCI receiver and transmitter.

5.1.2.8 SPI Serial I/O.

The SPI system is desabled by reset. The port pins associated with this function default to being general purpose I/O lines.

5.1.2.9 System.

The highest priority I interrupt defaults to being the external $\overline{IRQ}$ pin by PSEL3-PSEL0 equal to 0: 1: 0: 1. The $\overline{IRQ}$ interrupt pin is configured for level sensitive operation (for wire-OR systems). The RBOOT, SMOD, and MDA bits in the HPRIO register reflect the status of the MODB and MODA inputs at the rising edge of reset. The DLY control bit is set to specify that an oscillator start-up delay is imposed upon recovery from STOP mode. The clock monitor system is disabled by CME equal zero.

5.1.3 Computer Operating Properly (COP) Reset

The MCU includes a computer operating properly watchdog system to help protect against software failures. To use a COP watchdog timer, a watchdog timer reset sequence must be executed on a regular periodic basis so that the watchdog timer is never allowed to time out.

The internal COP function includes special control bits which permit specification of one of four time out periods and even allows the function to be disabled completely. The COP system has a separate reset vector.

Two other control bits in the OPTION register select one of four timeout durations for the COP timer. The actual timeout period is dependent on the system E clock frequency, but for reference purposes, Table 5.1 shows the relationship between the CR1 and CR0 control bits and the COP timeout period for various system clock frequencies.

Table 5.1 COP Timeout Period versus CR1 and CR0

CR1	CR0	E / 2^{15} Divided By	XTAL = 2^{23} Timeout - 0 / + 15.6ms	XTAL = 8.0 MHz Timeout - 0 / + 16.4ms	XTAL = 4.9152 MHz Timeout - 0 / + 26.7ms	XTAL = 4.0 MHz Timeout - 0 / + 32.8ms	XTAL = 3.6864 MHz Timeout - 0 / + 35.6ms
0	0	1	15.625 ms	16.384 ms	26.667 ms	32.768 ms	35.556 ms
0	1	4	62.5 ms	65.536 ms	106.67 ms	131.07 ms	142.22 ms
1	0	16	250 ms	262.14 ms	426.67 ms	524.29 ms	568.89 ms
1	1	64	1 s	1.049 s	1.707 s	2.1 s	2.276 s
		E clock	2.1 MHz	2.0 MHz	1.2288 MHz	1.0 MHz	921.6 kHz

The default reset condition of the CR1 and CR0 bits is cleared which corresponds to the shortest timeout period.

The sequence required to reset the watchdog timer is:

1) Write $55 to the COP reset register (COPRST) as $103A, followed by
2) Write $AA to the same address.

Both writes must occur in correct order prior to timeout but, any number of instructions may be executed between the writes. The elapsed time between adjacent software reset sequence must never be greater than the COP time out period. Reading the COPRST register does not return meaningful data and does not affect the watchdog timer.

5.1.4 Clock Monitor Reset

The clock monitor function is enabled by the CME control bit in the OPTION register. When CME bit is clear, the monitor function is disabled. When the CME bit is set, the clock monitor function detects the absence of an E clock for more than a certain period of time. The timeout period is dependent on processing parameters and will be between 5 and 100 microseconds. This means that an E-clock rate of 200 kHz or more will never cause a clock monitor failure and an E-clock rate of 10 kHz or less will definitely cause a clock monitor failure. This implies that systems operating near or below an E-clock rate of 200 kHz should not use the clock monitor function.

Upon detection of a slow or absent clock, the clock monitor circuit will cause a system reset. This reset is issued to the external system via the bidirectional $\overline{\text{RESET}}$ pin. The clock monitor system has a separate reset vector.

Special considerations are needed when using a STOP function and clock monitor in the same system. Since the STOP function causes the clocks to be halted, the clock monitor function will generate a reset sequence if it is enabled at the time the STOP mode is entered.

The clock monitor is useful as a backup for the COP watchdog timer. Since the watchdog timer requires a clock to function, it will not indicate any failure if the system clocks fail. The clock monitor would detect such a failure and force the MCU to its reset state. Note that clocks are not required for the MCU to reach its reset configuration, although clocks are required to sequence through reset back to the run condition.

5.1.5 Configuration Options Register (OPTION)

This is a special purpose 8-bit register that is used (optionally) during initialization to configure internal system configuration options. With the exception of bits 7, 6, and 3 (ADPU, CSEL, and CME) which may be read or written at any time, this register may be written to only once after a reset and thereafter is a read-only register. If no write is performed to this location within 64 E-clock cycles after reset, then bits 5, 4, 1, and 0 (IRQE, DLY, CR1, and CR0) will become read-only to minimize the possibility of any accidential changes to the system configuration (writes will be ignored). While in special test modes, the protection mechanism on this register is preempted and all bits in the OPTION register may be written repeatedly.

	7	6	5	4	3	2	1	0	
$1039	0	0	IRQE	DLY	CME	0	CR1	CR0	OPTION
RESET	0	0	0	1	0	0	0	0	

BITS 7, 6, 2

Not used in this register, these bits always read zero.

IRQE - $\overline{\text{IRQ}}$ Edge/Level Sensitive

This bit may only be written under special circumstances as described above. When this bit is clear, the $\overline{\text{IRQ}}$ pin is configuredfor level sensitive wired-OR operation (low level) and when it is set, the $\overline{\text{IRQ}}$ pin is configured for edge-only sensitivity (falling edges).

DLY - STOP Exit Turn-On Delay

This bit may only be written under special circumstances as described above. This bit is set during reset and controls whether or not a relatively long turn-on delay will be imposed before processing can resume after a STOP period. If an external clock source is supplied this delay can be inhibited so that processing can resume within a few cycles of a wake up from STOP mode. When DLY bit is set, a 4064 E clock cycle delay is imposed to allow oscillator stabilization and when DLY bit is clear, this delay is bypassed.

CME - Clock Monitor Enable

This control bit may be read or written at any time and controls whether or not the internal clock monitor circuit will trigger a reset sequence when a slow or absent system clock is detected. When it is clear, the clock monitor circuit is disabled and when it is set, the clock monitor circuit is enabled. Systems operating at or below 200 kHz should not use the clock monitor function. Reset clears the CME bit.

Bit 2 - Not Implemented

This bit always reads zero.

CR1 and CR0 - COP Timer Rate Selects

These bits may only be written under special circumstances as described above. Refer to Table 5.1 for the relationship between CR1: CR0 and the COP timeout period.

CR1	CR0	E / 2^{15} Divided By
0	0	1
0	1	4
1	0	16
1	1	64

5.2 INTERRUPTS

When an external or internal (hardware) interrupt occurs, the interrupt is not serviced until the current instruction being executed is completed. Until the current instruction is complete, the interrupt is considered pending. After completion of current instruction execution, unmasked interrupts may be serviced in accordance with an established fixed hardware priority circuit;however, one I-bit related interrupt source may be dynamically elevated to the highest I bit priority position in the heirarchy (see 5.2.5 Highest Priority I Interrupt Register (HPRIO)).

Seventeen hardware interrupts and one software interrupt (excluding reset type interrupts) can be generated from all of the possible sources. The interrupts can be divided into two basic categories, maskable and non-maskable. In the TMP68C711J6 fifteen of the interrupts can be masked using the condition code register I bit. In addition to being maskable by the I bit in the condition code register, all of the on-chip interrupt sources are individually maskable by local control bits.

The software interrupt (SWI instruction) is a non-maskable instruction rather than a maskable interrupt source. The illegal opcode interrupt is a non-maskable interrupt. The last interrupt source, external input to the $\overline{\text{XIRQ}}$ pin, is considered a non-maskable interrupt because once enabled, it cannot be masked by software;however, it is masked during reset and upon receipt of an interrupt at the $\overline{\text{XIRQ}}$ pin. Tables 5.2, 5.3 provide a list of each interrupt, its vector location in memory, and the actual condition code and control bits that mask it. A discussion of the various interrupts is provided below. Figure 5.4 shows the interrupt stacking order.

Table 5.2 Interrupt Vector Masks and Assignments

Vector Address	Interrupt Source	Condition Code Register Mask	Local Mask
FFC0, FFC1	Reserved	–	–
*	*		
*	*		
FFD4, FFD5	Reserved	–	–
FFD6, FFD7	SCI Serial System	I Bit	–
	Receive Data Register Full	–	RIE
	Receive Overrun	–	RIE
	Idle Line Detect	–	ILIE
	Transmit Data Register Empty	–	TIE
	Transmit Complete	–	TCIE
FFD8, FFD9	SPI Serial Transfer Complete	I Bit	SPIE
FFDA, FFDB	Pulse Accumulator Input Edge	I Bit	PAII
FFDC, FFDD	Pulse Accumulator Overflow	I Bit	PAOVI
FFDE, FFDF	Timer Overflow	I Bit	TOI
FFE0, FFE1	Timer IC4 / OC5	I Bit	I4O5I
FFE2, FFE3	Timer Output Compare 4	I Bit	OC4I
FFE4, FFE5	Timer Output Compare 3	I Bit	OC3I
FFE6, FFE7	Timer Output Compare 2	I Bit	OC2I
FFE8, FFE9	Timer Output Compare 1	I Bit	OC1I
FFEA, FFEB	Timer Input Capture 3	I Bit	IC3I
FFEC, FFED	Timer Input Capture 2	I Bit	IC2I
FFEE, FFEF	Timer Input Capture 1	I Bit	IC1I
FFF0, FFF1	Real-Time Interrupt	I Bit	RTII
FFF2, FFF3	$\overline{\text{IRQ}}$-External Pin	I Bit	None
FFF4, FFF5	$\overline{\text{XIRQ}}$ Pin (Pesudo-Nonmaskable)	X Bit	None
FFF6, FFF7	SWI	None	None
FFF8, FFF9	Illegal Opcode Trap	None	None
FFFA, FFFB	COP Failure (Reset)	None	NOCOP
FFFC, FFFD	Clock Monitor Fail (Reset)	None	CME
FFFE, FFFF	$\overline{\text{RESET}}$	None	None

Table 5.3 SCI Serial System Interrupts

Interrupt Cause	Local Mask
Receive Data Register Full	RIE
Receiver Overrun	RIE
Idle Detect	ILIE
Transmit Data Register Empty	TIE
Transmit Complete	TCIE

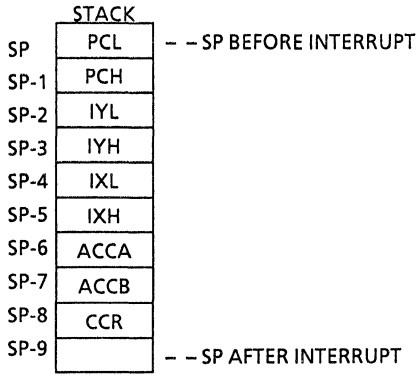


Figure 5.4 Interrupt Stacking Order

5.2.1 Software Interrupt (SWI)

The software interrupt is executed in the same manner as any other instruction and will take precedence over interrupts only if the orher interrupts are masked (I and X bits in the condition code register set). The SWI instruction is executed in a manner simslar to other maskable interrupts in that it sets the 1bit, CPU registers are stacked, etc.

Note : The SWI instruction will not be fetched if another interrupt os pending. However, once an SWI instruction has begun, no other interrupt can be honored until the SWI vector has been fetched.

5.2.2 Illegal Opcode Trap

Since not all possible opcodes or opcode sequences are defined, an illegal opcode detection circuit has been included. When an illegalopcode is detected, an interrupt is requested to the illegal opcode vector. The illegal opcode vector should never be left unitialized. It is a good idea to reinitialize the stack pointer as a result of an illegal opcode interrupt so repeated execution of illegal opcodes does not cause stack overruns.

5.2.3 Interrupt Mask Bits in Condition Code Register

Upon reset, both the X bit and the I bit are set to inhibit all maskable interrupts and $\overline{\text{XIRQ}}$. After minimum system initialization, software may clear the X bit by a TAP instruction, thus enabling $\overline{\text{XIRQ}}$ interrupts. Thereafter software cannot set the X bit so an $\overline{\text{XIRQ}}$ interrupt is effectively a nonmaskable interrupt. Since the operation of the I bit related interrupt structure has no effect on the X bit, the external $\overline{\text{XIRQ}}$ pin remains effectively non-masked. In the interrupt priority logic, the $\overline{\text{XIRQ}}$ interrupt is a higher priority than any source that is maskable by the 1bit. All I bit reated interrupts operate normally with their own priority relationship. When an I bit related interrupt occurs, the I bit is automatically set by hardware after stacking the condition code register byte, but the X bit is not affected.

When an X bit related interrupt occurs, both the X bit and the I bit are automatically set by hardware after stacking the condition code register. An RTI (return from interrupt) instruction restores the X and I bits to their pre-interrupt request state.

5.2.4 Priority Structure

Interrupts obey a fixed hardware priority circuit to resolve simultaneous requests; however, one 1bit related interrupt source may be elevated to the highest 1bit priority position in the resolution circuit. The first six interrupt sources are not masked by the 1bit in the condition code register and have the fixed priority interrupt relationship of: reset, clock monitor fail, COP fail, illegal opcode, and $\overline{\text{XIRQ}}$. (SWI is actually an instruction and has highest priority other than reset in the sense that once the SWI opcode is fetched, no other interrupt can be honored until the SWI vector has been fetched). Each of these sources is an input to the priority resolution circuit. The highest 1bit masked priority input to the resolution circuit is assigned under software control (of the HPRIO register) to be connected to any one of the remaining I bit related interrupt sources. In order to avoid timing races, the HPRIO register may only be written while the 1bit related interrupts are inhibited (1bit in condition code register is a logic one). An interrupt that is assigned to this high priority position is still subject to masking by any associated control bits or the 1bit in the condition code register. The interrupt vector address is not affected by assigning a source to this higher priority position.

Figures 5.5, 5.6, and 5.7 illustrate the interrupt process as it relates to normal processing. Figure 5.5 shows how the CPU begins from a reset and how interrupt detection relates to normal opcode fetches. Figure 5.6 is an expansion of a block in Figure 5.5 and shows how interrupt priority is resolved. Figure 5.7 is an expansion of the SCI interrupt block in Figure 5.7 shows the resolution of interrupt sources within the SCI subsystem.

5.2.5 Highest Priority I Interrupt and Miscellaneous Register (HPRIO)

Four bits of this register (PSEL3-PSEL0) are used to select one of the I-bit-related interrupt sources and to elevate it to the highest I-bit-masked position of the priority resolution circuit. In addition, four miscellaneous system control bits are included in this register.

	7	6	5	4	3	2	1	0	
$103C	RBOOT	SMOD	MDA	IRVNE	PSEL3	PSEL2	PSEL1	PSEL0	HPRIO
RESET:	0	0	0	0	0	1	0	1	Single-Chip mode
	0	0	1	0	0	1	0	1	Expanded-Nonmultiplexed mode
	1	1	0	0	0	1	0	1	Special Bootstrap mode
	0	1	1	1	0	1	0	1	Special Test mode

RBOOT - Read Bootstrap ROM

This bit can be read at any time. It can be written in special modes (SMOD = 1) at any time.

In normal modes (SMOD = 0), it can be written 1 on first write, it can be written 0 at any time.

In special bootstrap mode, it is set during reset. Reset clears it in all other modes.

1 = Bootloader ROM is enabled in the memory map at $BF00-BFFF.

0 = Bootloader ROM is disabled, and is not in the memory map.

SMOD and MDA - Special Mode Select, and Mode Select A

These two bits can be read at any time. They reflect the status of the MODA and MODB input pins at the rising edge of reset. SMOD may only be written to in special modes. It cannot be written to a one after being cleared without an interim reset. MDA may be written at any time in special modes, but only once in Single-chip mode. In Expanded-Nonmultiplexed mode, it cannot be written, it is read only bit.

An interpretation of the values of these two bits is shown in the following table:

Input MODB	Pins MODA	Mode Description	Latched SMOD	at Reset MDA
1	0	Single Chip	0	0
1	1	Expanded Nonmultiplexed	0	1
0	0	Special Bootstrap	1	0
0	1	Special Test	1	1

IRVNE - Internal Read Visibility Enable/Not E

This bit may be read at any time. It may only be written once in any mode. IRVNE is set during reset in special test mode only, and cleared by reset in the other modes.

In expanded and test modes, this bit determines whether the internal read visibility is on or off:

1 = Data from internal reads is driven out on the external data bus in expanded modes.

0 = Data from internal reads is not visible on the external data bus.

In single chip and bootstrap modes, IRVNE determines whether the E clock is driven out or forced low:

1 = E pin is driven low.

0 = E clock is driven out of the chip.

Mode	IRVNE Out of Reset	E Clock Out of Reset	IRV Out of Reset	IRVNE Affects Only	IRVNE May Be Written
Single-Chip	0	On	Off	E	Once
Expanded-Non multiplexed	0	On	Off	E	Once
Bootstrap	0	On	Off	IRV	Once
Special Test	1	On	On	IRV	Once

Note: To prevent bus conflicts when using internal read visibility, the user must disable all external devices from driving the data bus during any internal access.

PSEL3 - PSEL0 - Priority Selects

These four bits are used to specify one I bit related interrupt source, which then becomes the highest priority I bit related interrupt source. These bits may only be written while the I bit in the CCR is set, inhibiting I bit related interrupts. An interpretation of the value of these bits is shown in the following table:

PSEL3	PSEL2	PSEL1	PSEL0	Interrupt Source Promoted
0	0	0	0	Timer Overflow
0	0	0	1	Pulse Accumulator Overflow
0	0	1	0	Pulse Accumulator Input Edge
0	0	1	1	SPI Serial Transfer Complete
0	1	0	0	SCI Serial System
0	1	0	1	Reserved (Default to $\overline{IRQ}$)
0	1	1	0	$\overline{IRQ}$ (External Pin)
0	1	1	1	Real Time Interrupt
1	0	0	0	Timer Input Capture 1
1	0	0	1	Timer Input Capture 2
1	0	1	0	Timer Input Capture 3
1	0	1	1	Timer Output Compare 1
1	1	0	0	Timer Output Compare 2
1	1	0	1	Timer Output Compare 3
1	1	1	0	Timer Output Compare 4
1	1	1	1	Timer IC4 / OC5

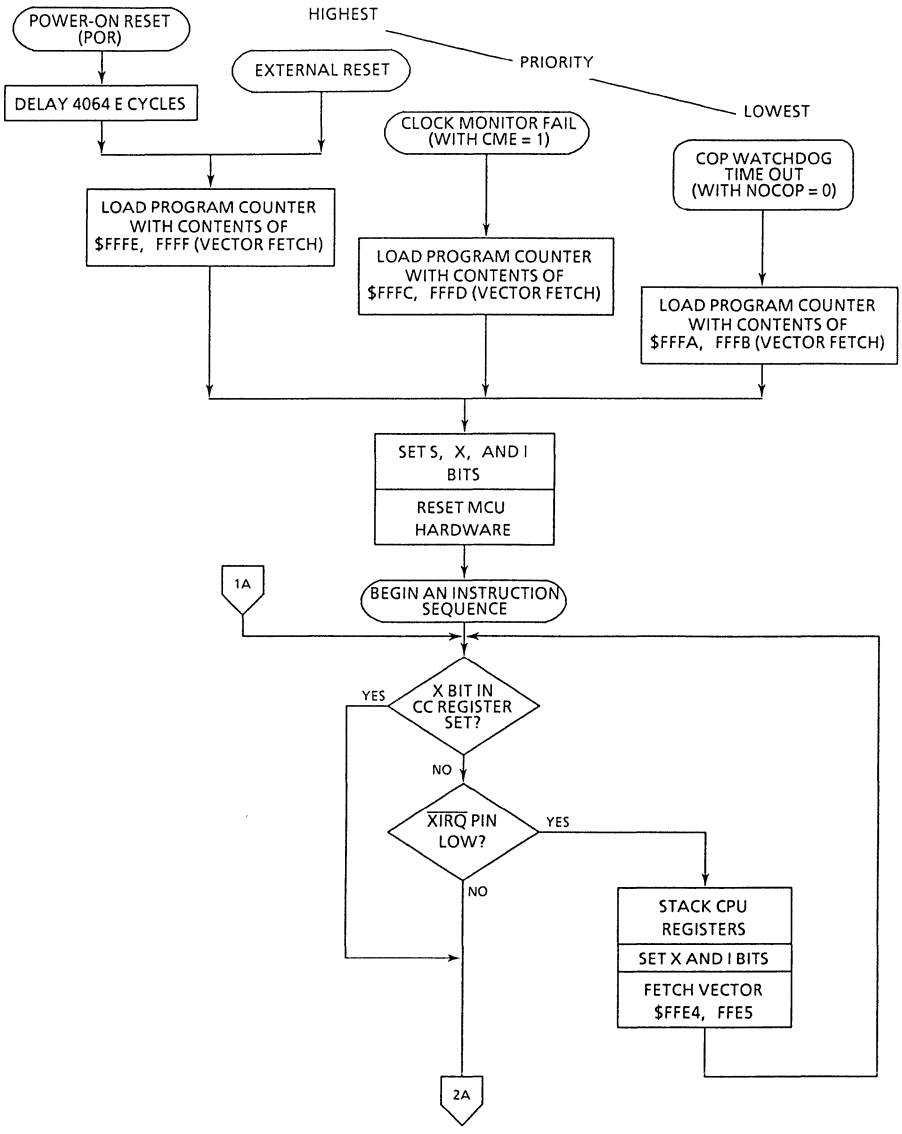


Figure 5.5 Processing Flow Out of Resets (Sheet 1of 2)

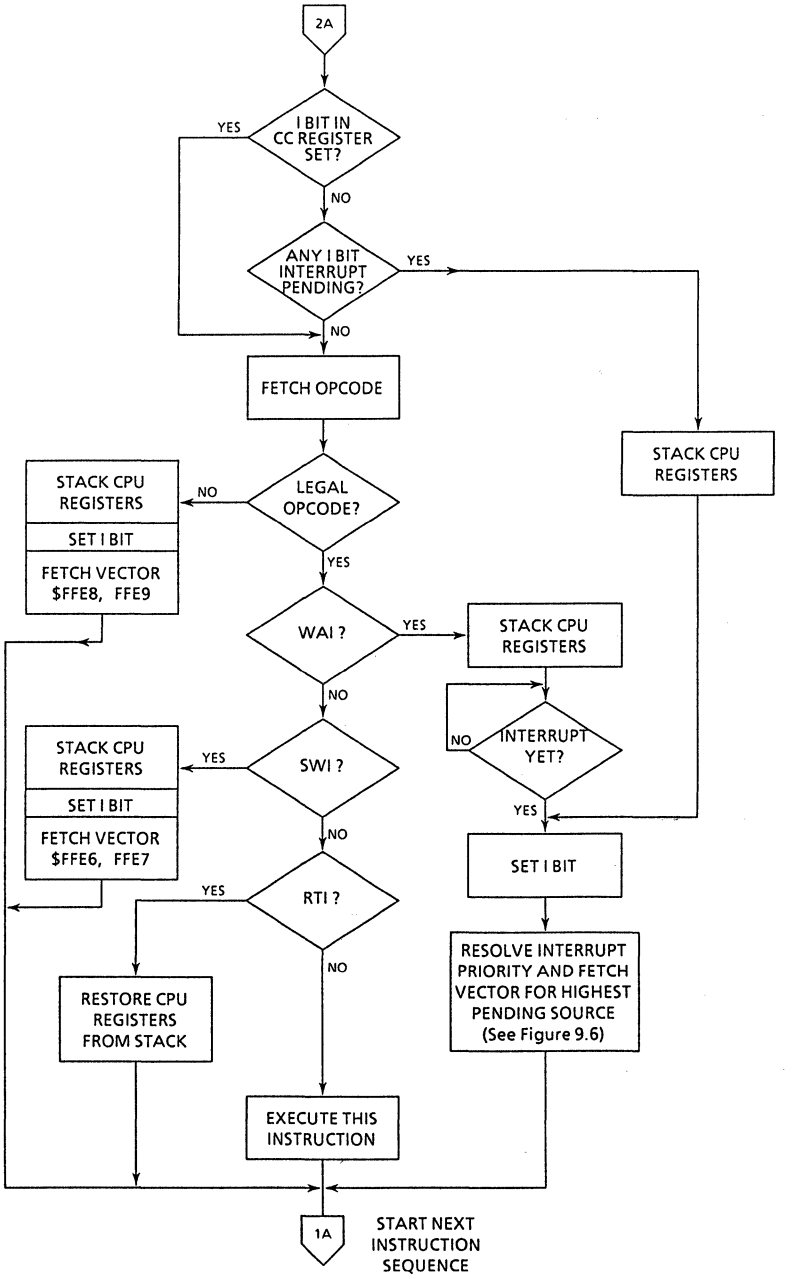


Figure 5.5 Processing Flow Out of Resets (Sheet 2 of 2)

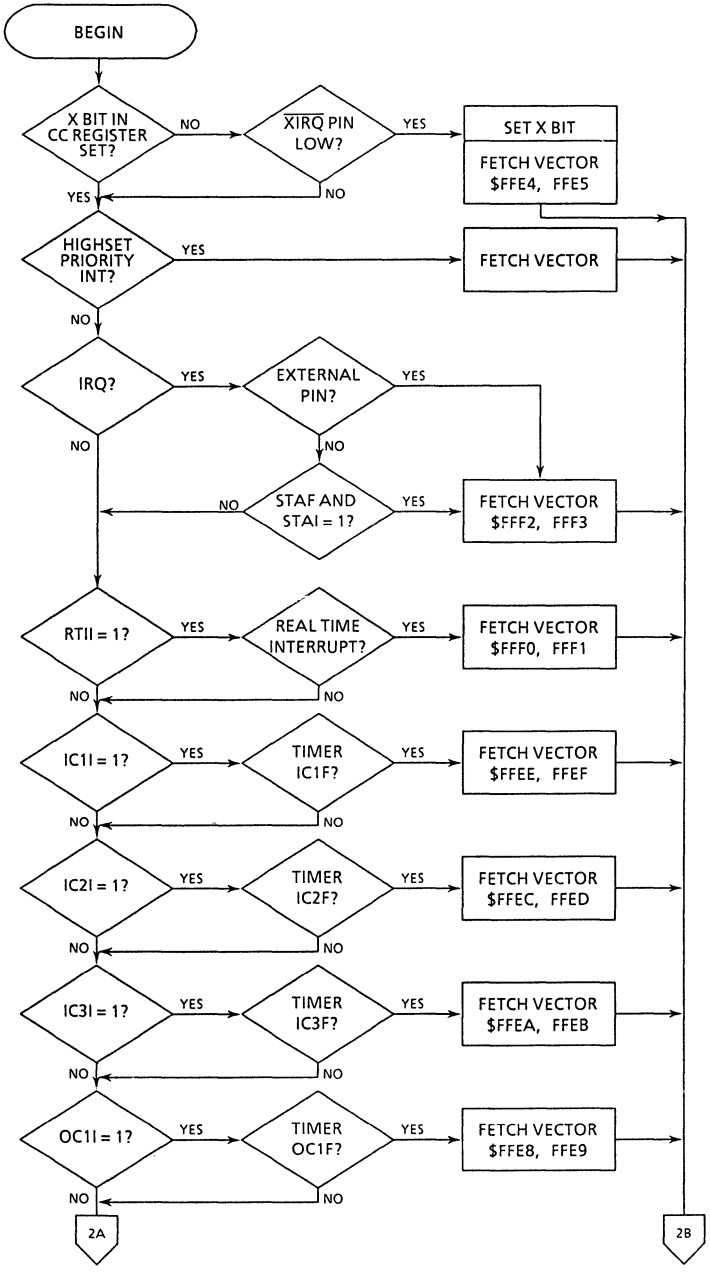
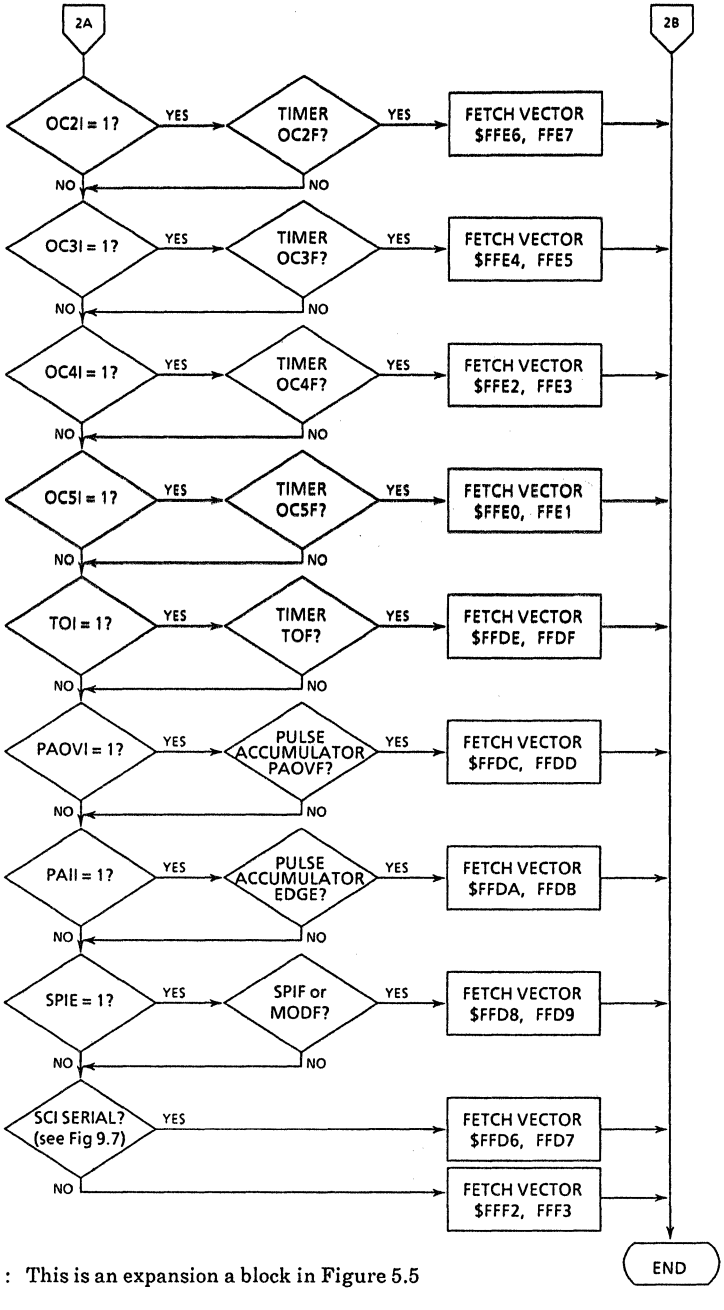
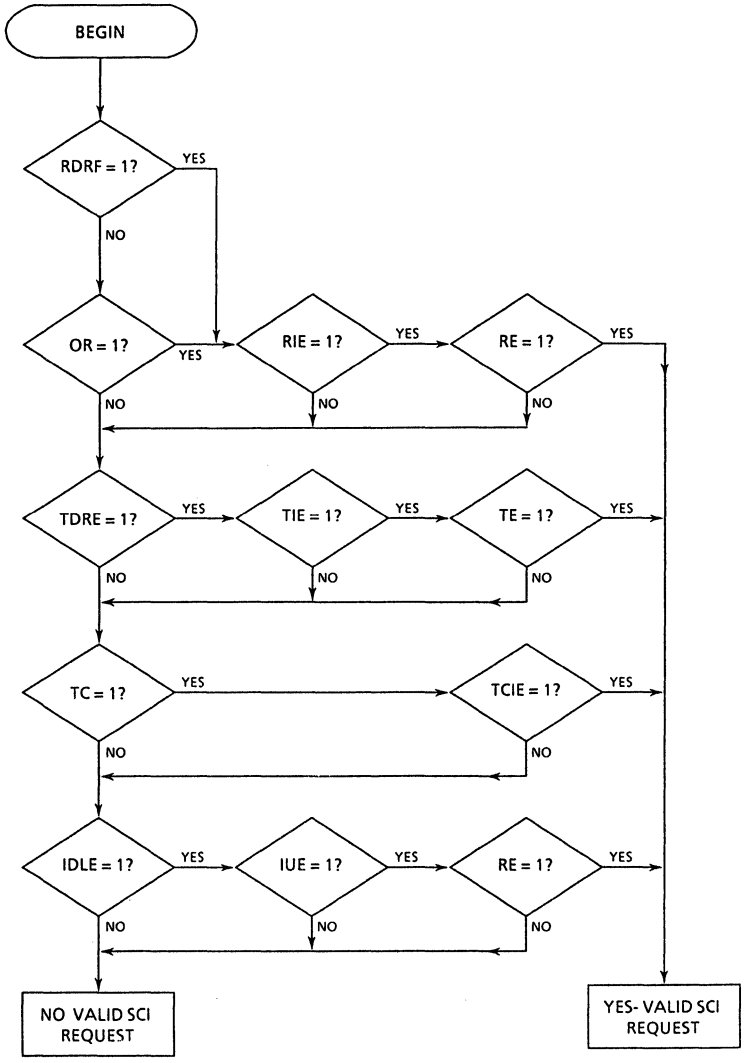


Figure 5.6 Interrupt Priority Resolution (Sheet 1 of 2)



Note : This is an expansion a block in Figure 5.5

Figure 5.6 Interrupt Priority Resolution (Sheet 2 of 2)



Note : This is an expansion a block in Figure 5.6

Figure 5.7 Interrupt Source Resolution Within SCI

5.3 LOW POWER MODES

The MCU contains two programmable low power consumption modes;WAIT and STOP. These two instructions are discussed below. Table 5.7 summarizes the activity on all pins of the MCU for all operating conditions.

5.3.1 WAIT Instruction

The WAI instruction puts the MCU in a low power consumption mode, keeping the oscillator running. Upon execution of a WAI instruction, the machine state is stacked and program execution stops. The wait state can be exited only by an unmasked interrupt or $\overline{\text{RESET}}$. If the I bit is set (interrupts masked) and the COP is disabled, the timer system will be turned off to additionally reduce power consumption. The amount of power savings is application dependent and depends upon circuitry connected to the MCU pins as well as which subsystems (i.e., timer, SPI, SCI) are active when the WAIT mode is entered.

5.3.2 STOP Instruction

The STOP instruction places the MCU in its lowest power consumption mode provided the S bit in the condition code register is clear. If the S bit is set, the STOP mode is disabled and STOP instructions are treated as NOPs (no operation). In the STOP mode, all clocks including the internal oscillator are stopped causing all internal processing to be halted. Recovery from the STOP mode may be accomplished by $\overline{\text{RESET}}$,$\overline{\text{XIRQ}}$, or an unmasked $\overline{\text{IRQ}}$. When the $\overline{\text{XIRQ}}$ is used, the MCU exits from the STOP mode regardless of the state of the X bit in the condition code register;however, the actual recovery sequence differs depending on the state of the X bit. If the X bit is clear, the MCU starts up with the stacking sequence leading to normal service of the $\overline{\text{XIRQ}}$ request. if the X bit is set, then processing will continue with the instruction immediately following the STOP instruction and no $\overline{\text{XIRQ}}$ interrupt service routine is requested. A $\overline{\text{RESET}}$ will always result in an exit from the STOP mode, and the start of MCU operation is determined by the reset vector.

Since the oscillator is stopped in the STOP mode, a restart delay of 4064 clock cycle times may be required to allow oscillator stabilization. If the internal oscillator is being used, this delay is required; however, if a stable external oscillator is being used, a control bit in the OPTION register may be used (DLY = 0) to give a delay of four cycle.

Table 5.7 Pin State Summary for RESET, STOP, and WAIT

Pins	Single-Chip mode Special Bootstrap mode			Expanded Nonmultiplexed mode Special Test mode		
	RESET	WAIT	STOP	RESET	WAIT	STOP
Output Only						
E	Active E	Active E	0	Active E	Active E	0
XTAL !!!	Active	Active	1	Active	Active E	1
Input/Output						
RESET	I (0)	I	I	I (0)	I	I
MODA/LIR	I (0)	OD (1)	OD (1)	I (1)	OD (1)	OD (1)
MODB/V_{STBY}	I (MODB)	I (V_{STBY})	I (V_{STBY})	I (MODES)	I (V_{STBY})	I (V_{STBY})
PA0-PA7	I	I/O	I/O	I	I/O	I/O
PB0-PB7	I	I/O	I/O	HI ADD	HI ADD	HI ADD
PC0-PC7	I	I/O	I/O	DATA	DATA	
PD0-PD5	I	I/O	I/O	I	I/O	I/O
PF0-PF7	I	I/O	I/O	LO ADD	LO ADD	LO ADD
PG0-PG5	I	I/O	I/O	I	I/O	I/O
PG6	I	I/O	I/O	O	O or I/O	O or I/O
PG7	I	I/O	I/O	O	O	O
Input Only						
EXTAL	Input Clock or Connect to Crystal with XTAL					
$\overline{IRQ}$	Terminate Unused Inputs to V_{DD}					
$\overline{XIRQ}$	Terminate Unused Inputs to V_{DD}					
PE0-PE7	If Not Used, External Drive Not Required					

SYMBOLS:

- DATA = Current data present.
- I = Input pin, if ( ) associated then this is required input state.
- I/O = Input output pin, state determined by data direction register.
- LO ADD = Low byte of address.
- HI ADD = High byte of the address.
- OD = Open drain output, ( ) current output state.
- SS = Steady state, output pin stays in current state.
- !!! = XTAL is output but not normally usable for any output function beyond crystal drive.

6. PROGRAMMABLE TIMER, REAL TIME INTERRUPT, AND PULSE ACCUMULATOR

This section describes the 16-bit programmable timer, the real time interrupt, and the pulse accumulator system.

6.1 PROGRAMMABLE TIMER

The timer has a single 16-bit free-running counter which is clocked by the output of a four-stage prescaler (divide by 1, 4, 8, or 16), which is in turn driven by the MCU E clock. Input functions are called input captures. These input captures record the count form the free-running counter in response to a detected edge on an input line. Output functions, called output compares, cause an output action when there is a match between a 16-bit output-compare register and the free-running counter. This timer system has three input capture registers and five output compare registers.

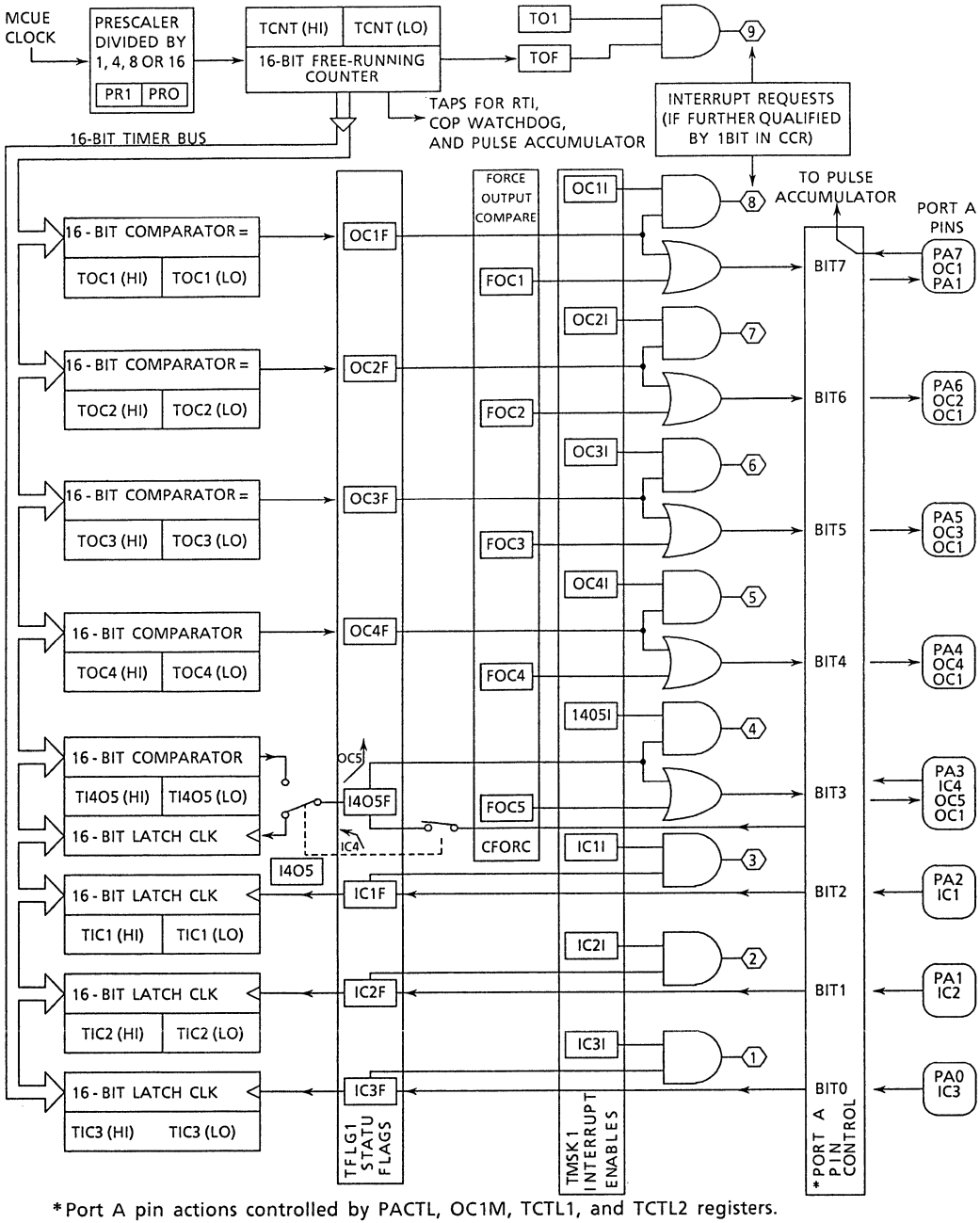
6.1.1 Counter

The key element in the timer system is a 16-bit free-running counter, or timer counter register. After reset, the MCU is configured to use the E clock as the input to the free-running counter. Initialization software may optionally reconfigure the system to use one of the three prescaler values. The prescaler control bits can only be written once during the first 64 cycles after a reset. Software can read the counter at any time without affecting its value because it is clocked and read during opposite phases of the E clock.

A counter read should first address the most significant byte. An MPU read of this address causes the least significant byte to be transferred to a buffer. This buffer is not affected by reset and is accessed when reading the least significant byte of the counter. For double byte read instructions, the two accesses occur on consecutive bus cycles.

The counter is cleared to $0000 during reset and is a read-only register with one exception. In test modes only, any MPU write to the most significant byte presets the counter to $FFF8 regardless of the value involved in the write.

When the count changes from $FFFF to $0000, the timer overflow flag (TOF) bit is set in timer interrupt flag register 2 (TFLG2). An interrupt can be enabled by setting the interrupt enable bit (TOI) in timer interrupt mask register 2 (TMSK2).



*Port A pin actions controlled by PACTL, OC1M, TCTL1, and TCTL2 registers.

Figure 6.1 Timer Block Diagram

6.1.2 Input Capture

The input capture registers are 16-bit read-only registers which are not affected by reset and are used to latch the value of the counter when a defined transition is sensed by the corresponding input capture edge detector. The level transition which triggers counter transfer is defined by the corresponding input edge bits (EDGxB, EDGxA) in TCTL2.

The result obtained by an input capture corresponds to the value of the counter one E clock cycle after the transition which triggered the edge-detection logic. The selected edge transition sets the ICxF bit in timer interrupt flag register 1 (TFLG1) and can cause an interrupt if the corresponding ICxI bit(s) is (are) set in the timer interrupt mask register 1 (TMSK1). A read of the input capture register's most significant byte inhibits captures for one E cycle to allow a double-byte read of the full 16-bit register.

6.1.2.1 Input Capture4.

Port A pin 3 serves multiple functions. After reset, data direction bit 3 (DDRA3), in the PACTL register is cleared to zero configuring Port A pin 3 as an input. Port A pin 3 can then be used as an input capture4 (IC4), by setting I4/O5 to "one" in the PACTL register. The I4/O5 bit is configured to OC5 (cleared to zero) on reset. If DDRA3 is configured as an output (set to one) and IC4 is enabled, writes to Port A bit 3 causes edges on the PA3 pin to result in input captures. All other aspects of using IC4 remain the same as the other input captures, weith the exception that the 16-bit timer output compare 5 register now also server as the 16-bit timer input capture 4 register. When the TI4O5 register is acting as the IC4 capture register it cannot be written to. Upon reset, I4/O5 is configured as . The OC5 function overrides DDRA3 to force the Port A pin 3 to be an output whenever OM5:OL5 bit are not 0:0. In all other aspects, OC5 works the same as the other output compares.

6.1.3 Output Compare

All output compare registers are 16-bit read/write registers which are initialized to $FFFF by reset. They can be used as output waveform controls or as elapsed time indicators. If an output compare register is not used, it may be used as a storage location.

All output compare registers have a separate dedicated comparator for comparing against the free-running counter. If a match is found, the corresponding output compare flag (OCxF) bit in TFLG1 is set and a specified action is automatically taken. For output compare functions two through five the automatic action is controlled by pairs of bits (OMx and OLx) in the timer control register 1 (TCTL1). Each pair of control bits are encoded to specify the output action to be taken as a result of a successful OCx compare. The output action is taken on each succesful compare regardless of wheather or not the OCxF flag was previously clear.

An interrupt can also accompany a successful output compare, provided that the corresponding interrupt enable bit (OCx1) is set in TMSK1.

After an MPU write cycle to the most significant byte, output compares are inhibited for one E cycle in order to allow writing two consecutive bytes before making the next comparison. If both bytes of the register are to be changed, a double-byte write instruction should be used in order to take advantage of the compare inhibit feature.

MPU write can be made to either byte of the output compare register without affecting the other byte.

A write-only register, timer compare force (CFORC), allows forced compares. Five of the bit positions in the CFORC register correspond to the five output compares. To force a compare, or compares, a write is done to CFORC register withe the associated bits set for each output compare that is to be forced. The action taken as a result of a forced compare is the same as if there was a match between the OCx register and the free-running counter, except that the corresponding interrupt status flag bits are not set. Output actions are synchronized to the prescaled timer clock so there could be as much as 16 E clock cycles of delay between the write to CFORC and the output action.

6.1.4 Output Compare 1 I/O Pin Control

Unlike the other four output compares, output compare 1 can automatically affect any or all of the five output pins (bits 3-7) in port A as a result of a successful compare between the OC1 register and the 16-bit free-running counter. The two 5-bit registers used in conjunction with this function are the output compare 1 mask register (OC1M) and the output compare 1 data register (OC1D).

Register OC1M is used to specify the bits of port A (I/O and timer port) which are to be affected as a result of a successful OC1 compare. Register OC1D is used to specify the data which is to be stored to the affected bits of port A as the result of a successful OC1 compare. If an OC1 compare and another output compare occur during the same E cycle and both attempt to alter the same port A line, the OC1 compare prevails.

This function allows control of multiple I/O pins automatically with a single output compare.

Another intended use for the special I/O pin control on output compare 1 is to allow more than one output compare to control a single I/O pin. This allows pulses as short as one E clock cycle to be generated.

6.1.5 Timer Compare Force Register (CFORC)

The timer compare force register is used to force early output compare actions. The CFORC register is an 8-bit write-only register. Reads of this location have no meaning and always return logic zeros. Note that the compare force function is not generally

recommended for use with the output toggle function because a normal compare occuring immediately before or after the force may result in undesirable operation.

	7	6	5	4	3	2	1	0	
$100B	FOC1	FOC2	FOC3	FOC4	FOC5	0	0	0	CFORC
RESET	0	0	0	0	0	0	0	0	

FOC1-FOC5 - Force Output Compare x Action

0= Has no meaning

1= Causes action programmed for output compare x, except the OCxF flag bit is not set.

Bits 2-0 - Not Implemented

These bits always read zero.

6.1.6 Output Compare 1 Mask Register (OC1M)

This register is used in conjunction with output compare 1 to specify the bits of port A which are affected as a result of a successful OC1 compare.

	7	6	5	4	3	2	1	0	
$100C	OC1M7	OC1M6	OC1M5	OC1M4	OC1M3	0	0	0	OC1M
RESET	0	0	0	0	0	0	0	0	

The bit of the OC1M register correspond bit-for-bit with the lines of port A (lines 7 through 3 only). For each bit that is affected by the successful compare, the corresponding bit in OC1M should be set to one.

Note that the pulse accumulator function shares line 7 of port A. If the DDRA7 bit in the pulse accumulator control register (PACTL) is set, then port A line 7 is configured as an output and OC1 can obtain access by setting OC1M bit 7. In this condition if the PAEN bit in the PACTL register is set, enabling the pulse accumulator input, then OC1 compares cause a write of OC1D bit 7 to an internal latch, and the output of that latch drives the pin and the pulse accumulator input. This action can then cause the pulse accumulator to take the appropriate action (pulse count or gate modes).

6.1.7 Output Compare 1 Data Register (OC1D)

This register used in conjunction with output compare 1 to specify the data which is to be stored to the affected bits of port A as the result of a successful OC1 compare.

	7	6	5	4	3	2	1	0	
$100D	OC1D7	OC1D6	OC1D5	OC1D4	OC1D3	0	0	0	OC1D
RESET	0	0	0	0	0	0	0	0	

The bits of the OC1D register correspond bit-for-bit with the lines of port A (lines 7 thru 3 only) . When a successful OC1 compare occurs, for each bit that is set in OC1M, the corresponding data bit in OC1D is stored in the corresponding bit of port A. If there is a conflicting situation where an OC1 compare and another output compare function occur during the same E cycle with both attempting to alter the same port A line, the OC1 action prevails.

6.1.8 Timer Control Register 1 (TCTL1)

	7	6	5	4	3	2	1	0	
$1020	OM2	OL2	OM3	OL3	OM4	OL4	OM5	OL5	TCTL1
RESET	0	0	0	0	0	0	0	0	

OM2, OM3, OM4, and OM5 - Output Mode

OL2, OL3, OL4, and OL5 - Output Level

These two control bits (OMx and OLx) are encoded to specify the output action taken as a result of a successful OCx compare.

OMx	OLx	Action Taken Upon Successful Compare
0	0	Timer disconnected from output pin logic
0	1	Toggle OCx output line
1	0	Clear OCx output line to zero
1	1	Set OCx output line to one

6.1.9 Timer Control Register 2 (TCTL2)

	7	6	5	4	3	2	1	0	
$1021	EDG4B	EDG4B	EDG1B	EDG1A	EDG2B	EDG2A	EDG3B	EDG3A	TCTL2
RESET	0	0	0	0	0	0	0	0	

EDGxB and EDGxA - Input Caputre x Edge Control.

These two bits (EDGxB and EDGxA) are cleared to zero by reset and are encoded to configure the input sensing logic for input capture x as follow:

EDGxB	EDBxA	Configuration
0	0	Capture disabled
0	1	Capture on rising edges only
1	0	Capture on falling edges only
1	1	Capture on any (rising or falling)edge

6.1.10 Timer Interrupt Mask Register 1 (TMSK1)

	7	6	5	4	3	2	1	0	
$1022	OC1I	OC2I	OC3I	OC4I	I4O5I	IC1I	IC2I	IC3I	TMSK1
RESET	0	0	0	0	0	0	0	0	

OCxI - Output compare x Interrupt

If the OCxI enable bit is set when the OCx flag bit is set, a hardware interrupt sequence is requested.

ICxI - Input Capture x Interrupt

If the ICxI enable bit is set when the ICxF flag bit is set, a hardware interrupt sequence is requested.

Note : When the I4/O5 bit in the PACTL register is one, the I4O5I bit behaves as the Input Capture 4 Interrupt bit, and when I4/O5 is zero, the I4O5I bit acts as the Output Compare 5 Interrupt control bit.

6.1.11 Timer Interrupt Flag Register 1 (TFLG1)

Timer interrupt flag register 1 is used to indicate the occurrence of timer system events, and together with the TMSK1 register allows the timer subsystem to oprate in a polled or interrupt driven system. For each bit in TFLG1, there is a corresponding bit in TMSK1 in the same bit position. If the mask bit is set, each time the condition for the corresponding flag are met, a hardware interrupt sequence is requested as well as the flg bit being set.

These timer system status flags are cleared by writing a one to the bit positions corresponding to the flag(s) which are to be cleared. Bit manipulation instructions would be inappropriate for flag clearing because they are read-modify-write instruction. Even though the instruction mask implies that the programmer is only interested in

some of the bits in the manipulated location, the entire location is actually read and rewritten which may clear other bits in the register.

	7	6	5	4	3	2	1	0	
$1023	OC1F	OC2F	OC3F	OC4F	I4O5F	IC1F	IC2F	IC3F	TFLG1
RESET	0	0	0	0	0	0	0	0	

OCxF - Output Compare xFlag

This flag bit is set each time the timer counter matches the output compare register x value. A write of a zero does not affect this bit. A write of a one causes this bit to be cleared.

ICxF - Input Caputure x Flag

This flag is set each time a selected active edge is detected on the ICx input line. A write of a zero does not affect this bit. A write of a one causes this bit to be cleared.

Note : When the I4/O5 bit in the PACTL register is one, the I4O5F bit behaves as the Input Capyure 4 Flag bit, and when I4/O5 is zero, the I4O5I bit acts as the Output Compare 5 Flag.

6.1.12 Timer Interrupt Mask Register 2 (TMSK2)

Timer interrupt mask register 2 is used to control whether or not a hardware interrupt sequence is requested as a result of a status bit being set in timer interrupt flag register 2. In addition, two timer prescaler bits are included in this register. For each of the four most significant bits in timer flag register 2, (TFLG2), there is a correponding bit in the timer mask register 2 (TMSK2) in the same bit position.

	7	6	5	4	3	2	1	0	
$1024	TOI	RTII	PAOVI	PAII	0	0	PR1	PR0	TMSK2
RESET	0	0	0	0	0	0	0	0	

TOI - Timer Overflow Interrupt Enable

0 = TOF interrupts disabled

1 = Interrupt requested when TOF = 1

RTII - RTI Interrupt Enable

0 = RTIF interrupts disabled

1 = Interrupt requested when RTIF = 1

PAOVI - Pulse Accumulator Overflow Interrupt Enable

0 = PAOVF interrupts disabled

1 = Interrupt requested when PAOVF = 1

PAII - Pulse Accumulator Input Interrupt Enable

0 = PAIF interrupts disabled

1 = Interrupt requested when PAIF = 1

Bits 3 and 2 - Not Implemented

These bits always read zero.

PR1 and PRO - Timer Prescaler Selects

These two bits may be read at any time but may only be written during initalization. Writes are disabled after the first write or after 64 E cycles out of reset.

If the MCU is in special test or special bootstrap mode, then these two bits may be written any time.

Bits3 and 2 - Not Implemented

These bits always read zero.

PR1 and PRO - Timer Prescaler Selects

These two bits may be read at any time but may only be written during initalization. Writes are disabled after the first write or after 64 E cycles out of reset. If the MCU is in special test or special bootstrap mode, then these two bits may be written any time.

These two bits specify the timer prescaler divide factor.

PR1	PR0	Divide-by-Factor
0	0	1
0	1	4
1	0	8
1	1	16

6.1.13 Timer Interrupt Flag Register 2 (TFLG2)

Timer interrupt flag register 2 is used to indicate the occurrence of timer system events, and together with the TMSK2 register, allows the timer subsystems to operate in a polled or interrupt driven system. For each bit in timer flag register 2 (TFLG2), there is a corresponding bit in timer mask register 2 (TMSK2) in the same bit position. If the enable bit is set each time the conditions for the corresponding flag are met, a hardware interrupt sequence is requested as well as the flag bit being set.

The timer system status flags are cleared by writing a one to the bit positions corresponding to the flag(s) which are to be cleared. Bit manipulation instructions would be inappropriate for flag clearing because they are read-modify-write instructions. Even though the instruction mask implies that the programmer is only interested in some of the bits in the manipulated location, the entire location is actually read and rewritten which may clear other bits in the register.

	7	6	5	4	3	2	1	0	
$1025	TOF	RTIF	PAOVF	PAIF	0	0	0	0	TFLG2
RESET	0	0	0	0	0	0	0	0	

TOF - Timer Overflow

This bit is cleared by reset. It is set to one each time the 16-bit free-running counter advances from a value of $FFFF to $0000. This bit is cleared by a write to the TFLG2 register with bit 7 set.

RTIF - Real Time Interrupt Flag

This bit is set at each rising edge of the selected tap point. This bit is cleared by a write to the TFLG2 register with bit 6 set.

PAOVF - Pulse Accumulator Overflow Interrupt Flag

This bit is set when the count in the pulse accumulator rolls over from $FF to $00. This bit is cleared by a write to the TFLG2 register with bit 5 set.

PAIF - Pulse Accumulator Input Edge Interrupt Flag

This bit is set when an active edge is detected on the PAI input pin. This bit is cleared by a write to the TFLG2 register with bit 4 set.

Bits 3-0 - Not Implemented

These bits always read zero.

6.2 REAL TIME INTERRUPT

The real time interrupt feature on the MCU is configured and controlled by using two bits (RTR1 and RTR0) in the PACTL register to select one of four interrupt rates. The RTII bit in the TMSK2 register enables the interrupt capability. Every timeout causes the RTIF bit in TFLG2 to be set, and if RTII is set, and interrupt request is generated. After reset, one entire real time interrupt period elapses before the RTIF flag is set for the first time.

6.3 PULSE ACCUMULATOR

The pulse accumulator is an 8-bit read/write counter which can operate in either of two modes (external event counting or gated time accumulation) depending on the state of the PAMOD control bit in the PACTL register. In the event counting mode, the 8-bit counter is clocked to increasing values by an external pin. The maximum clocking rate for the external event counting mode is E clock divided by two. In the gated time accumulation mode, a free-running E clock/64 signal drives the 8-bit counter, but only while the external PAI input pin is activated.

The pulse accumulator uses port A bit 7 as its PAI input, but this pin also shares function as a general purpose I/O pin and as a timer output compare pin. Normally port A bit 7 would be configure as a input when being used for the pulse accumulator. Note that even when port A bit 7 is configured for output, this pin still drives the input to the pulse accumulator.

6.4 PULSE ACCUMULATOR CONTROL REGISTER (PACTL)

Four bits in this register are used to control an 8-bit pulse accumulator system and two other bits are used to select the rate for the real time interrupt system.

	7	6	5	4	3	2	1	0	
$1026	0	PAEN	PAMOD	PEDGE	0	I4/O5	RTR1	RTR0	PACTL
RESET	0	0	0	0	0	0	0	0	

Bit 7.3 Not used in this register, these bits always read zero.

PAEN - Pulse Accumulator System Enable

0 = Pulse accumulator off

1 = Pulse accumulator on

PAMOD - Pulse Accumulator Mode

0 = External event counting

1 = Gated time accumulation

PEDGE - Pulse Accumulator Edge Control

This bit has different meanings depending on the state of the PAMOD bit.

PAMOD	PEDGE	Action on Clock
0	0	PAI Falling Edge Increments the Counter
0	1	PAI Rising Edge Increments the Counter
1	0	A '0' on PAI Inhibits Counting
1	1	A '1' on PAI Inhibits Counting

I4/O5 - Input Capture 4/Output Compare5

0 = Output compare 5 function enable (No IC4)

1 = Input capture4 function enable (No OC5)

These bits always read zero.

RTR1 and RTR0 - RTI Interrupt Rate Selects

These two bits select one of four rates for the real rime periodic interrupt circuit (see Table 6.1). Reset clears these two bits and after reset, a full RTI period elapses before the first RTI interrupt.

Table 6.1 Real Time Interrupt Rate versus RTR1 and RTR0

RTR1	RTR0	Divide E By	XTAL = 2^{23}	XTAL = 8.0 MHz	XTAL = 4.9152 MHz	XTAL = 4.0 MHz	XTAL = 3.6864 MHz
0	0	2^{13}	3.91 ms	4.10 ms	6.67 ms	8.19 ms	8.89 ms
0	1	2^{14}	7.81 ms	8.19 ms	13.33 ms	16.38 ms	17.78 ms
1	0	2^{15}	15.62 ms	16.38 ms	26.67 ms	32.77 ms	35.56 ms
1	1	2^{16}	31.25 ms	32.77 ms	53.33 ms	65.54 ms	71.11 ms
		E clock	2.1 MHz	2.0 MHz	1.2288 MHz	1.0 MHz	921.6 MHz

7. PROGRAMMABLE READ ONLY MEMORY (PROM)

The TMP68C711J6 has 16K bytes of programmable read only memory (PROM), either erasable programmable read only memory (EPROM) or one-time programmable read only memory (OTPROM). The PROM address is $C000-FFFF. In expanded-nonmultiplexed mode, where the PROM is located at $4000-$7FFF, the PROM is turned off after reset.

7.1 PROM PROGRAMMING

There are two separate methods of programming the on-chip PROM of an TMP68HC711J6. In the first method (PROG Mode), certain pins of the MCU are made to emulate the pins of a 27256 type EPROM, and may be programmed as a standard EPROM. In the second method (MCU Mode), the PROM is programmed through the MCU in the bootstrap or test modes.

7.2 PROGRAMMING PROM USING PROG MODE

In order to make the TMP68C711J6 emulate a standard 27256 type EPROM, the MCU must be shifted into a new programming mode called PROG, and a footprint conversion must be made using an adapter. The MCU enters PROG when pins MODA, MODB, and $\overline{\text{RESET}}$ are held low. No clocks are necessary. Table 7-1 shows which MCU pins emulate EPROM pins in this mode:

Table 7.1. 27256 Emulation

MCU Pins	EPROM Pins
PF7-PF0	A7-A0
PB6-PB0	A14-A8
PB7	$\overline{\text{OE}}$
PC7-PC0	O7-O0
$\overline{\text{XIRQ}}$	V_{PP}
$\overline{\text{IRQ}}$	$\overline{\text{CE}}$
V_{DD}	V_{CC}
V_{SS}	GND
MODA	GND
MODB	GND
$\overline{\text{RESET}}$	GND
PA7-PA0	GND
PD7-PD0	GND
PG7-PG0	GND
PE7-PE0	GND
EXTAL	GND
XTAL	NC
E	NC

Note in table 7-1 that MCU address A14 is connected, but treated as a "don't care" to allow a potential upgrade to 32K PROM. The 16K PROM repeats 2 times in the 32K programming space of the 27256.

7.2.1 External Read and Verify (PROG Mode)

To read and verify the OTPROM or EPROM, the MCU is placed in PROG mode. $\overline{OE}$ (MCU pin PB7) is held high to de-select the PROM. $\overline{CE}$ may be high or low at this point. An address is presented. After a setup time, $\overline{OE}$ and $\overline{CE}$ (if not already low) are brought low to allow the address to decode the EPROM location. After access time is completed, data becomes valid on the data bus and is read. $\overline{OE}$ is brought high before presenting the next address.

When verifying only, it is acceptable to leave both $\overline{OE}$ and $\overline{CE}$ low and present addresses. The data is presented out of the data bus after a setup time.

7.2.2 External Programming (PROG Mode)

To program the OTPROM or EPROM, the MCU is placed in PROG mode. $\overline{CE}$ is held high while the V_{PP} ($\overline{XIRQ}$) pin is brought to programming voltage level. $\overline{OE}$ is held high throughout. The address and data to be programmed are presented. After a setup time, $\overline{CE}$ is brought low. This state is held for the duration of programming time. $\overline{CE}$ must be brought high before the address and data are changed to program the next location. The correlation between MCU and PROG Mode pins is shown in Figure 7-1.

The erased state of each byte of PROM is $FF. Each bit of the EPROM or OTPROM is programmable to 0 on a per byte basis. Once an EPROM bit is programmed to 0, it may not be reprogrammed to 1 without UV erasure. OTPROM cannot be erased or reprogrammed. Bytes may be programmed in any order.

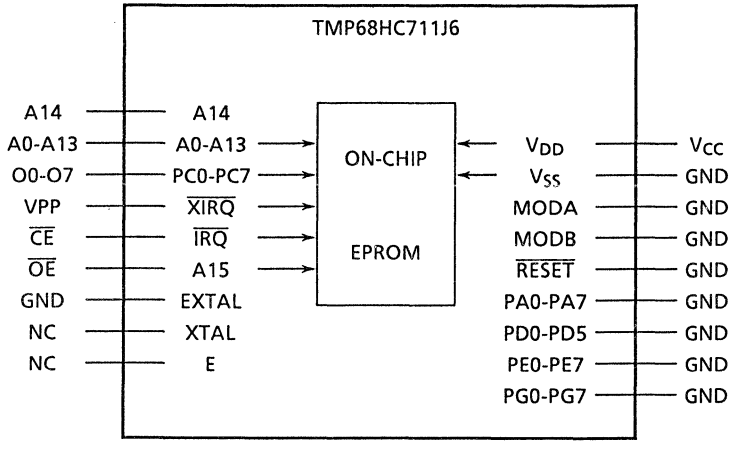


Figure 7.1 TMP68C711J6 Block Diagram in PROG Mode

7.3 PROM PROGRAMMING USING THE MCU

Programming the OTPROM or EPROM through the MCU is only allowed in special test and bootstrap modes. With the EPON bit of the CONFIG register set, the various control bits of the PPROG register are manipulated to program the PROM locations. The erased state of a PROM byte is $FF.

7.3.1 Configuration Control Register (CONFIG)

The configuration control register controls the presence of OTPROM or EPROM in the memory map and enables the COP watchdog system.

This register is writable only once in expanded and single chip modes (SMOD = 0). In these modes, the COP watchdog timer is enabled out of reset. The EPROM is forced into the memory map out of reset in all modes except expanded mode. If the user wishes to have the PROM in the map in expanded mode, he must first come up in single chip mode, then write to the HPRIO register, setting the MDA bit. This switches the MCU to expanded mode with PROM enabled.

	7	6	5	4	3	2	1	0	
$103F	0	0	0	0	0	NOCOP	EPON	0	CONFIG
RESET:	0	0	0	0	0	*	*	0	

Bits 7-3 and 0 - Not implemented. These bits always read zero.

NOCOP - Computer Operating Properly System Disable

This bit is cleared out of reset in normal modes (single chip and expanded), enabling the COP system. It is writable only once after reset in these modes (SMOD = 0). In the special modes (test and bootstrap), (SMOD = 1), this bit comes out of reset set, and is writable any time.

1 = COP system disabled

0 = COP system enabled, reset forced on timeout

EPON - PROM Enable

This bit is set out of reset, enabling the EPROM or OTPROM in all modes. In single chip, bootstrap, and test modes, the PROM is located at $C000-$FFFF. In expanded mode, the PROM is located at $4000-$7FFF. This bit is not writable in single-chip mode (SMOD = 0, MODA = 0), but is writable once in expanded mode (SMOD = 1, MODA = 1).

In special modes (SMOD = 1), this bit is written always.

1 = PROM is present in the memory map.

0 = PROM is disabled from the memory map.

7.3.2 PROM Programming Control Register (PPROG)

This register is used to control the programming of the OTPROM or EPROM. PPROG is cleared on reset so that the PROM is configured for normal read.

	7	6	5	4	3	2	1	0	
$103B	0	0	ELAT	0	0	0	0	PGM	PPROG
RESET:	0	0	0	0	0	0	0	0	

Bits 7, 6, 4, 3, 2, and 1 Not used in this register, these bits always read zero.

ELAT - EPROM (OTPROM) Latch Control

1 = PROM address and data bus are configured for programming. Writes to PROM cause address and data to be latched. The PROM cannot be read.

0 = PROM address and data bus configured for normal reads. PROM cannot be programmed.

PGM - EPROM (OTPROM) Program Command

This bit may be written only when ELAT = 1.

1 = Programming power switched on to PROM array.

0 = Programming power switched off.

7.3.3 PROM Programming Sequence

Before programming, ensure that V_{PP} voltage is available on the $\overline{XIRQ}$ pin.

Note that $\overline{XIRQ}$ must not become active (be brought low) during programming. This state would disrupt or corrupt the programming of the OTPROM or EPROM. The proper sequence for programming the PROM through the MCU (with EPON bit set in CONFIG) is as follows:

STEP	Comments
1. Write $20 to PPROG 2. Write data to PROM 3. Write $21 to PPROG 4. Delay 1 ms. 5. Write $20 to PPROG	Set ELAT bit (PGM = 0) to enable PROM latches. Store Data to PROM Address. Set PGM bit (ELAT = 1) to enable PROM high voltage. Turn off high voltage to PROM array.
6. Repeat steps 2 through 5, as needed.	
7. Write $00 to PPROG	Return to read mode.

7.3.4 Protecting the PROM

some precautions are necessary in order to protect the OTPROM or EPROM. For systems which do not make use of the on-chip programming capability, but program the MC68HC711J6 as a standard EPROM, voltage on the $\overline{XIRQ}/V_{PP}$ pin should never be permitted to be greater than V_{DD}. The PROM cannot be programmed or corrupted without high voltage on the $\overline{XIRQ}$ pin.

Systems that use the on-chip programming feature require that V_{PP} voltage be available. Regardless of which mode the MCU is in, if the part is operating with V_{PP} voltage present on the $\overline{XIRQ}/V_{PP}$ pin, then the $\overline{IRQ}/\overline{CE}$ pin must be pulled to a high level in reset.

The pin configuration necessary to place the MCU in bootstrap mode (MODA, MODB, and $\overline{RESET}$ pins being low) is also the configuration that places the MCU in the PROG state. If, at the same time, V_{PP} is present and $\overline{IRQ}$ is low, then the PROM is being programmed. A pullup resistor on the $\overline{IRQ}$ line prevents this situation by ensuring the $\overline{IRQ}$ pin is high leaving reset.

7.3.5 Erasing the PROM

OTPROM MCU devices are shipped in an erased state. Once programmed, they cannot be erased. Electrical erasing procedures cannot be performed on either OTPROM (non-windowed packages) or EPROM devices (windowed packages).

EPROM devices can be erased by exposure to a high intensity ultraviolet (UV) light with a wavelength of 2537 Angstroms. The recommended integrated dosage (UV intensity $\times$ exposure time) is 15 Ws/cm^2. UV lamps should be used without shortwave filters, and the EPROM device should be positioned about one inch from the UV lamp. the erased state of an EPROM location is $FF.

8. SERIAL COMMUNICATIONS INTERFACE (SCI)

This section contains a description of the serial communication interface (SCI).

8.1 OVERVIEW AND FEATURES

A full-duplex asynchronous Serial Communications Interface (SCI) is provided with a standard NRZ format (one start bit, eight or nine data bits, and one stop bit) and a variety of baud rates. The SCI transmitter and receiver are functionally independent, but use the same data format and bit rate. "Baud" and "bit rate" are used synonymously in the following description.

SCI Two-Wire System Features

- Standard NRZ (mark/space) format.
- Advanced error detection method includes noise detection for noise duration of up to 1/16 bit time.
- Full-duplex operation.
- Software programmable for one of 32 different baud rates.
- Software selectable word length (eight or nine bit words).
- Separate transmitter and receiver enable bits.
- Capable of being interrupt driven.
- Four separate enable bits available for interrupt control.

SCI Receiver Features

- Receiver wake-up function (idle or address bit).
- Idle line detect.
- Framing error detect.
- Noise detect.
- Overrun detect.
- Receiver data register full flag.

SCI Transmitter Features

- Transmit data register empty flag.
- Transmit complete flag.
- Send break.

8.2 DATA FORMAT

Receive data or transmit data is the serial data which is transferred to the internal data bus from the receive data input pin (RxD), or from the internal bus to the transmit data output pin (TxD). The non-return-to-zero (NRZ) data format shown in Figure 5.1 is used and must meet the following criteria:

(1) The idle line is brought to a logic one state prior to transmission/reception of a character.

(2) A start bit (logic zero) is used to indicate the start of a frame.

(3) The data is transmitted and received least-significant-bit first.

(4) A stop bit (logic one) is used to indicate the end of a frame. A frame consists of a start bit, a character of eight or nine data bits, and a stop bit.

(5) A break is defined as the transmission or reception of a low (logic zero) for at least one complete frame time.

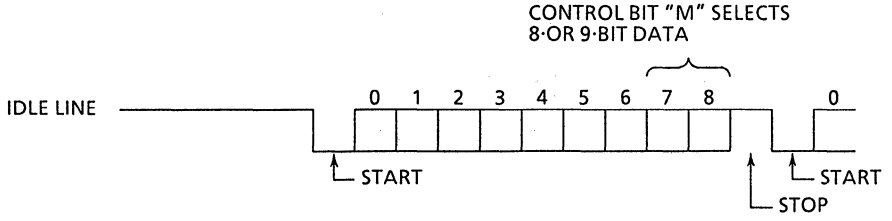


Figure 8.1 Data Format

8.3 WAKE-UP FEATURE

The receiver wake-up feature reduces SCI service overhead in multiple receiver systems. Software in each receiver evaluates the first character(s) of each message. If the message is intended for a different receiver, the SCI can be placed in a sleep mode so that the rest of the message will not generate requests for service. Whenever a new message is started, logic in the sleeping receivers causes them to wake up so they can evaluate the initial character(s) of the new message.

A sleeping SCI receiver can be configured (using the WAKE control bit in serial communications control register 1 (SCCR1)) to wake up using either of two methods:idle line wake up or address mark wake up.

In idle line wake up, a sleeping receiver wakes up as soon as the RxD line becomes idle. Idle is defined as a continuous logic high on the RxD line for ten (or eleven) full bit times. Systems using this type of wake up must provide at least one character time of

idle between messages to wake up sleeping receivers but must not allow any idle time between characters within a message.

In address mark wake up, the most significant bit (MSB) in a character is used to indicate that the character is an address (1) or a data (0) character. Sleeping receivers will wake up whenever an address character is received. Systems using this method for wake up would set the MSB of the first character in each message and leave it clear for all other characters in the message. Idle periods may be present within messages and no idle time is required between messages for this wake up method.

8.4 RECEIVE DATA (RxD)

Receive data is the serial data which is applied through the input line and the serial communications interface to the internal bus. The receiver circuitry clocks the input at a rate equal to 16 times the baud rate and this time is referred to as the RT clock.

Once a valid start bit is detected, the start bit, each data bit, and the stop bit are sampled three times at RT intervals 8 RT, 9 RT, and 10 RT (1 RT is the position where the bit is expected to start), as shown in Figure 8.2 The value of the bit is determined by voting logic which takes the value of the majority of samples.

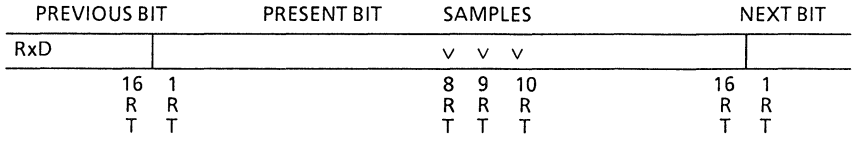


Figure 8.2 Sampling Technique Used on All Bits

8.5 START BIT DETECTION

When the RxD input is detected low, it is tested for three more sample times (referred to as the start edge verification samples in Figure 8.3). If at least two of these three verification samples detect a logic zero, a valid start bit has been detected, otherwise the line is assumed to be idle. A noise flag is set if all three verification samples do not detect a logic zero. A valid start bit could be assumed with a set noise flag present.

If there has been a framing error without detection of a break (10 zeros for 8-bit format or 11 zeros for 9-bit format), the circuit continues to operate as if there actually was a stop bit and the start edge will be placed artificially. The last bit received in the data shift register is inverted to a logic one, and the three logic one start qualifiers (shown in Figure 8.3) are forced into the sample shift register during the interval when detection of a start bit is anticipated (see Figure 8.4);therefore, the start bit will be accepted no sooner than it is anticipated.

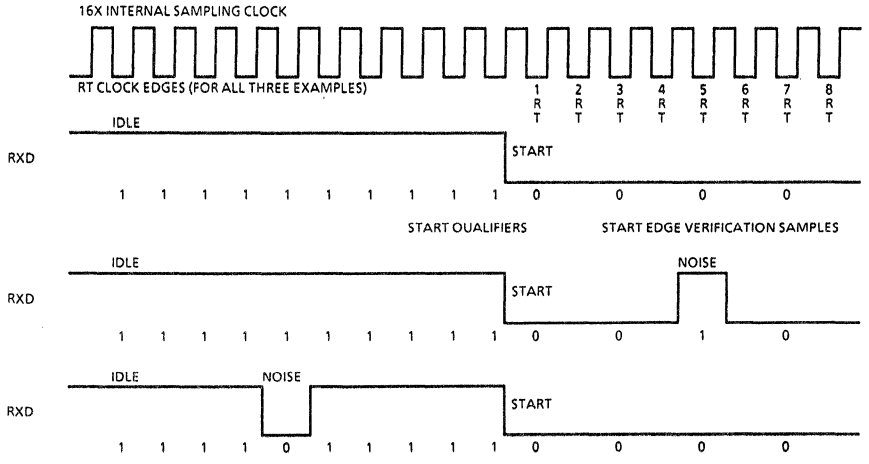
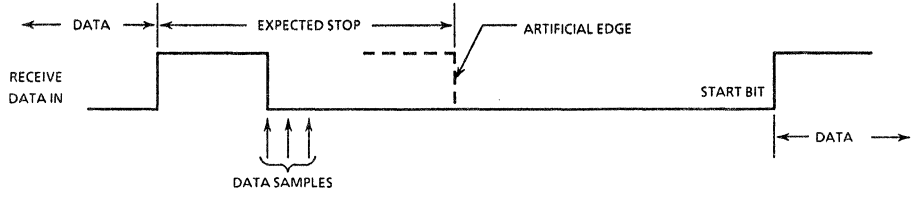
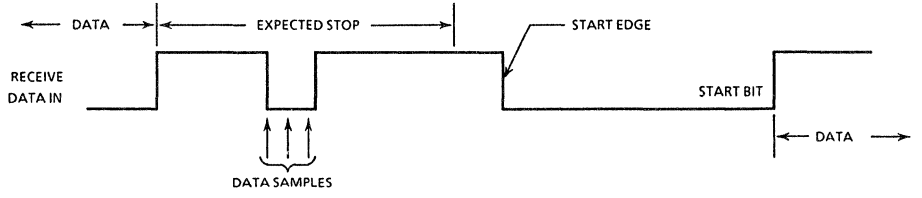


Figure 8.3 Examples of Start Bit Sampling Techniques



(a) Case1, Receive Line Low During Artificial Edge



(b) Case2, Receive Line High During Expected Start Edge

Figure 8.4 SCI Artifical Start Following a Framing Error

If the receiver detects that a break produced the framing error, the start bit will not be artificially induced and the receiver must actually detect a logic one before the start bit can be recognized. See Figure 8.5.

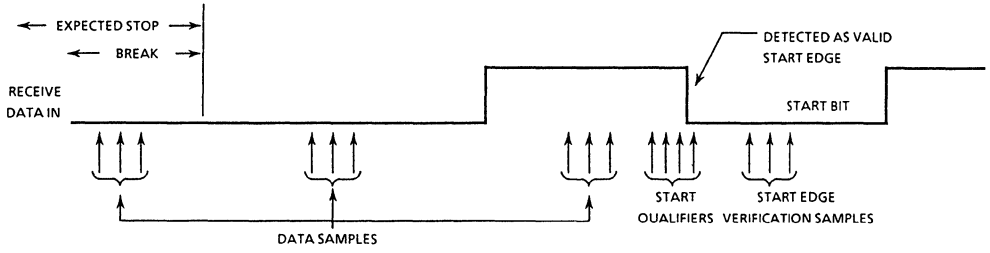


Figure 8.5 SCI Start Bit Following a Break

8.6 TRANSMIT DATA (TxD)

Transmit data is the serial data from the internal data bus which is applied through the serial communications interface to the output line. The transmitter generates a bit time by using a derivative of the RT clock, thus producing a transmission rate equal to 1/16 that of the receiver sample clock.

8.7 FUNCTIONAL DESCRIPTION

A block diagram of the SCI is shown in Figure 8.6. The user has option bits in serial communications control register 1 (SCCR1) to determine the "Wake-up" method (WAKE bit) and data word length (M bit) of the SCI. Serial communications control register 2 (SCCR2) provides control bits which individually enable/disable the transmitter or receiver (TE and RE, respectively), enable system interrupts (TIE, TCIE, ILIE) and provide the wake-up enable bit (RWU) and the send break code bit (SBK). The baud rate register (BAUD) bits allow the user to select different baud rates which may be used as the rate control for the transmitter and receiver.

Data transmission is initiated by a write to the serial communications data register (SCDR). Provided the transmitter is enabled, data stored in the SCDR register is transferred to the transmit data shift register. This transfer of data sets the TDRE bit of the SCI status register (SCSR) and may generate an interrupt if the transmit interrupt is enabled. The transfer of data to the transmit data shift register is synchronized with the bit rate clock (Figure 8.7). All data is transmitted LSB first. Upon completion of data transmission, the transmission complete (TC) bit of the SCSR register is set (provided no pending data, preamble, or break is to be sent), and an interrupt may be generated if the transmit complete interrupt is enabled. If the transmitter is disabled, and the data, preamble, or break (in the transmit shift register) has been sent, the TC bit will also be set. This will also generate an interrupt if the TCIE bit is set. If the transmitter is disabled in the middle of a transmission, that character will be completed before the transmitter gives up control of the TxD pin.

When the SCDR register is read, it contains the last data byte received, provided that the receiver is enabled. The RDRF bit of the SCSR register is set to indicate that a data byte has been transferred from the input serial shift register to the SCDR register, which can cause an interrupt if the receiver interrupt is enabled. The data transfer from the input serial shift register to the SCDR register is synchronized by the receiver bit rate clock. The OR (over-run), NF (noise), or FE (framing) error bits of the SCSR register may be set if data reception errors occurred.

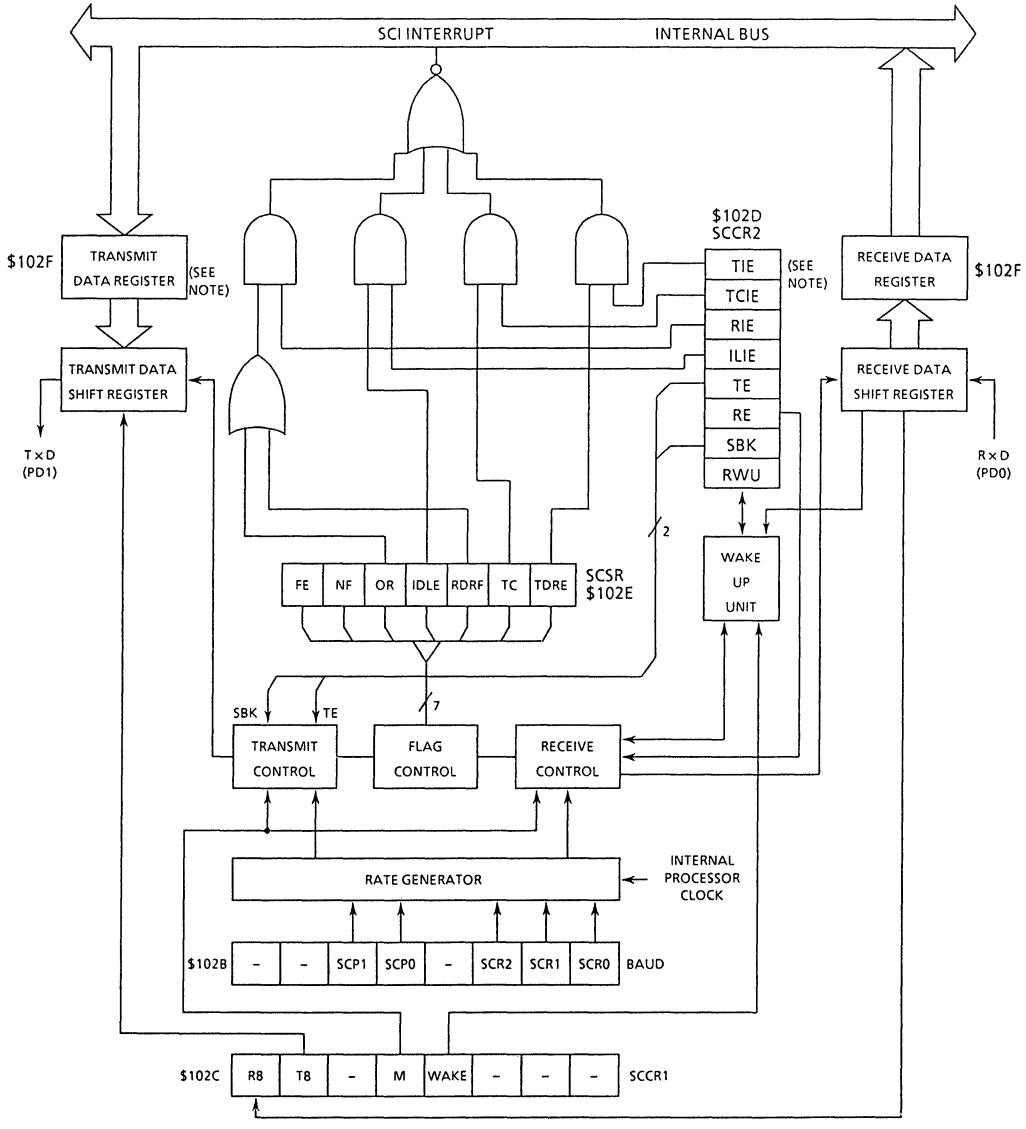
An idle line interrupt is generated if the idle line interrupt is enabled and the IDLE bit (which detects idle line transmission) of SCSR register is set. This allows a receiver that is not in the wake-up mode to detect the end of a message, the preamble of a new message, or to resynchronize with the transmitter. A valid character must be received before the idle line condition or the IDLE bit will not be set and an idle interrupt will not be generated.

8.8 SCI REGISTERS

There are five registers used in the serial communications interface and the operation of these registers is discussed in the following paragraphs. Reference should be made to the block diagram shown in Figure 8.6.

8.8.1 Serial Communications Data Register (SCDR)

The serial communications data register performs two functions;i.e., it acts as the receive data register when it is read and as the transmit data register when it is written. Figure 8.6 shows this register as two separate registers, namely:the receive data register and the transmit data register.



Note : The Serial Communications Data Register (SCDR) is controlled by the internal R/$\overline{W}$ signal. It is the transmit data register when witten and receive data register when read.

Figure 8.6 Serial Communications Interface Block Diagram

8.8.2 Serial Communications Control Register 1 (SCCR1)

The serial communications control register 1 (SCCR1) provides the control bits which: (1) determine the word length, and (2) select the method used for the wake-up feature.

	7	6	5	4	3	2	1	0	
$102C	R8	T8	0	M	WAKE	0	0	0	SCCR1
RESET	U	U	0	0	0	0	0	0	

R8 - Receive Data Bit 8

If the M bit is set, this bit provides a storage location for the ninth bit in the receive data character.

T8 - Transmit Data Bit 8

If the M bit is set, this bit provides a storage location for the ninth bit in the transmit data character. It is not necessary to write to this bit for every character transmitted, only when the sense is to be different than that for the previous character.

Bit 5 - Not Implemented

This bit always reads zero.

M - SCI Character Length

0 = 1 start bit, 8 data bits, 1 stop bit

1 = 1 start bit, 9 data bits, 1 stop bit

WAKE - Wake Up Method Select

0 = Idle Line

1 = Address Mark

Bits 2, 1 - Not Implemented

These bits always read zero.

8.8.3 Serial Communications Control Register 2 (SCCR2)

The serial communications control register 2 (SCCR2) provides the control bits which enable/disable individual SCI functions.

	7	6	5	4	3	2	1	0	
$102D	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK	SCCR2
RESET	0	0	0	0	0	0	0	0	

TIE - Transmit Interrupt Enable

0 = TDRE interrupts disabled

1 = SCI interrupt if TDRE = 1

TCIE - Transmit Complete Interrupt Enable

0 = TC interrupts disabled

1 = SCI interrupt if TC = 1

REI - Receive Interrupt Enable

0 = RDRF and OR interrupts disabled

1 = SCI interrupt if RDRF or OR = 1

ILIE - Idle Line Interrupt Enable

0 = IDLE interrupts disabled

1 = SCI interrupt if IDLE = 1

TE - Transmit Enable

When the transmit enable (TE) bit is set, the transmit shift register output is applied to the TxD line. Depending on the state of control bit M (SCCR1), a preamble of 10 (M = 0) or 11(M = 1) consecutive ones is transmitted when software sets the TE bit from a cleared state. After loading the last byte in the serial communications data register and receiving the TDRE flag, the user can clear TE bit. Transmission of the last byte will then be completed before the transmitter gives up control of the TxD pin. While the transmitter is active, the data direction register control for port D bit 1 is overridden and the line is forced to be an output.

RE - Receive Enable

When the receive enable (RE) bit is set, the receiver is enabled. When RE bit is clear, the receiver is disabled and all of the status bits associated with the receiver (RDRF, IDLE, OR, NF and FE) are inhibited. While the receiver is enabled, the data direction register control for port D bit 0 is overridden and the line is forced to be an input.

RWU - Receiver Wake Up

When the receiver wake-up (RWU) bit is set by the user's software, it puts the receiver to sleep and enables "wake up" function. If the WAKE bit is cleared, RWU bit is cleared by the SCI logic after receiving 10 (M=0) or 11 (M=1) consecutive ones. If the WAKE bit is set, RWU bit is cleared by the SCI logic after receiving a data word whose MSB is set.

SBK - Send Break

If the send break (SBK) bit is toggled and cleared, the transmitter sends 10 (M=0) or 11 (M=1) zeros and then reverts to idle or sending data. If SBK bit remains set, the transmitter will continually send whole blocks of zeros (sets of 10 or 11) until cleared. At the completion of the break code, the transmitter sends at least one high bit to guarantee recognition of a valid start bit. If the transmitter is currently empty and idle, setting and clearing SBK bit is likely to queue two character times of break because the first break transfers almost immediatly to the shift register and the second is then queued into the parallel transmit buffer.

8.8.4 Serial Communications Status Register (SCSR)

The serial communications status register (SCSR) provides inputs to the interrupt logic circuits for generation of the SCI system interrupt.

	7	6	5	4	3	2	1	0	
$102E	TDRE	TC	RDRF	IDLE	OR	NF	FE	0	SCSR
RESET	1	1	0	0	0	0	0	0	

TDRE - Transmit Data Register Empty

The transmit data register empty (TDRE) bit is set to indicate that the content of the serial communications data register have been transferred to the transmit serial shift register. This bit is cleared by reading the SCSR register (with TDRE=1) followed by a write to the SCDR register .

TC - Transmit Complete

The transmit complete (TC) bit is set at the end of a data frame, preamble, or break condition if:

1) TE=1, TDRE=1, and no pending data, preamble, or break is to be transmitter; or

2) TE=0, and the data, preamble, or break in the transmit shift register has been transmitted.

The TC bit is a status flag which indicates that one of the above conditions have occurred. The TC bit is cleared by reading the SCSR register (with TC set) followed by a write to the SCDR register.

RDRF - Receive Data Register Full

The receive data register full (RDRF) bit is set when the receiver serial shift register is transferred to the SCDR register. The RDRF bit is cleared when the SCSR register is read (with RDRF set) followed by a read of the SCDR register.

IDLE - Idle Line Detect

The idle line detect (IDLE) bit, when set, indicates the receiver has detected an idle line. The IDLE bit is cleared by reading the SCSR register with IDLE bit set followed by reading SCDR register. Once the IDLE status flag is cleared, it will not be set again until after the RxD line has been active and becomes idle again.

OR - Overrun Error

The overrun error (OR) bit is set when the next byte is ready to be transferred from the receive shift register to the SCDR register which is already full (RDRF bit is set). When an overrun error occurs, the data which caused the overrun is lost and the data which was already in SCDR register is not disturbed. The OR bit is cleared when the SCSR register is read (with OR bit set), followed by a read of the SCDR register.

NF - Noise Flag

The noise flag (NF) bit is set if there is noise on any of the received bits, including the start and stop bits. The NF bit is not set until the RDRF flag is set. The NF bit is cleared when the SCSR register is read (with NF bit set), followed by a read of the SCDR register.

FE - Framing Error

The framing error (FE) bit is set when no stop bit was detected in the received data character. The FE bit is set at the same time as the RDRF bit is set. If the byte received causes both framing and overrun errors, the processor will only recognize the overrun error. The framing error flag inhibits further transfer of data into the SCDR register until it is cleared. The FE bit is cleared when the SCSR register is read (with FE bit equal to one) followed by a read of the SCDR register.

Bit 0 - Not Implemented

This bit always reads zero.

8.8.5 Baud Rate Register (BAUD)

The baud rate register selects the different baud rates which may be used as the rate control for the transmitter and receiver. The SCP0-SCP1 bits function as a prescaler

for the SCR0-SCR2 bits. Together, these five bits provide multiple baud rate combinations for a given crystal frequency.

	7	6	5	4	3	2	1	0	
$102B	TCLR	0	SCP1	SCP0	RCKB	SCR2	SCR1	SCR0	BAUD
RESET	0	0	0	0	0	U	U	U	

TCLR - Clear Baud Rate Counters (Test)

This bit is used to clear the baud rate counter chain during factory testing. TCLR bit is zero and cannot be set while in normal operating modes.

SCP1 and SCP0 - SCI Baud Rate Prescaler Selects

The E clock is divided by the factors shown in Table 8.1. This prescaled output provides an input to a divider which is controlled by the SCR2-SCR0 bits.

Table 8.1 Second Prescaler Stage

SCR1	SCR0	Internal Processor Clock Divide By
0	0	1
0	1	3
1	0	4
1	1	13

SCR2, SCR1, and SCR0 - SCI Baud Rate Selects

These three bits select the baud rates for both the transmitter and the receiver. The prescaler output described above is further divided by the factors shown in Table 8.2.

Table 8.2 Second Prescaler Stage

SCR2	SCR1	SCR0	Prescaler Output Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

RCKB - SCI Baud Rate Clock Check (Test)

This bit is used during factory testing to enable the exclusive-OR of the receiver clock and transmitter clock to be driven out the TxD pin. RCKB bit is zero and cannot be set while in normal operating modes.

The diagram shown in Figure 8.7 and the data given in Tables 8.3 and 8.4 illustrate the divider chain used to obtain the baud rate clock. Note that there is a fixed rate divide-by-16 between the receive clock (RT) and the transmit clock (Tx). The actual divider chain is controlled by the combined SCP0-SCP1 and SCR0-SCR2 bits in the baud rate register as illustrated.

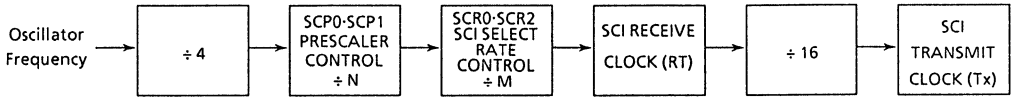


Figure 8.7 Rate Generator Division

Table 8.3 Prescaler Highest Baud Rate Frequency Output

SCP Bit		Clock* Divided By	Crystal Frequency (MHz)				
1	0		8.3886	8.0	4.9152	4.0	3.6864
0	0	1	131.072 K Baud	125.000 K Baud	76.80 K Baud	62.50 K Baud	57.60 K Baud
0	1	3	43.691 K Baud	41.666 K Baud	25.60 K Baud	20.833 K Baud	19.20 K Baud
1	0	4	32.768 K Baud	31.250 K Baud	19.20 K Baud	15.625 K Baud	14.40 K Baud
1	1	13	10.082 K Baud	9600 Baud	5.907 K Baud	4800 Baud	4430 Baud

*The cloick in the "Clock Divided By" column is the internal processor clock.

Note: The divided frequencies shown in Table8.3 represent baud rates which are the highest transmit baud rate (Tx) that can be obtained by a specific crystal frequency and only using the prescaler division. Lower baud rates may be obtained by providing a further division using the SCI rate select bits as shown below for some representative prescaler outputs.

Table 8.4 Transmit Baud Rate Output for a Given Prescaler Output

SCR Bits			Divided By	Representative Highest Prescaler Baud Rate Output				
2	1	0		131.072 K Baud	32.768 K Baud	76.80 K Baud	19.20 K Baud	9600 Baud
0	0	0	1	131.072 K Baud	32.768 K Baud	76.80 K Baud	19.20 K Baud	9600 Baud
0	0	1	2	65.536 K Baud	16.384 K Baud	38.40 K Baud	9600 Baud	4800 Baud
0	1	0	4	32.768 K Baud	8.192 K Baud	19.20 K Baud	4800 Baud	2400 Baud
0	1	1	8	16.384 K Baud	4.096 K Baud	9600 Baud	2400 Baud	1200 Baud
1	0	0	16	8.192 K Baud	2.048 K Baud	4800 Baud	1200 Baud	600 Baud
1	0	1	32	4.096 K Baud	1.024 K Baud	2400 Baud	600 Baud	300 Baud
1	1	0	64	2.048 K Baud	512 Baud	1200 Baud	300 Baud	150 Baud
1	1	1	128	1.024 K Baud	256 Baud	600 Baud	150 Baud	75 Baud

Note: Table8.4 illustrates how the SCI select bits can be used to provide lower transmitter baud rates by further dividing the prescaler output frequency. The five examples are only representative samples. In all cases, the baud rates shown are transmit baud rates (transmit clock) and the receiver clock is 16 times higher in frequency than the actual baud rate.

9. SERIAL PERIPHERAL INTERFACE (SPI)

This section contains a description on the serial peripheral interface (SPI).

9.1 OVERVIEW AND FEATURES

The serial peripheral interface (SPI) is a synchronous interface which allows several SPI microcontrollers or SPI-type peripherals to be interconnected. 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. The TMP68C711J6 SPI system may be configured either as a master or as a slave.

Features include:

- Full Duplex, Three-Wire Synchronous Transfers
- Master or Slave Operation
- 1.575 MHz (Maximum) Master Bit Frequency
- 3.15 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
- Easily Interfaces to Simple Expansion Parts (PLLs, D/As, Latches, Display Drivers, etc.)

9.2 SPI SIGNAL DESCRIPTIONS

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

Any SPI output line has to have its corresponding data direction register bit set. If this bit is clear, the line is disconnected from the SPI logic and becomes a general-purpose input line. Any SPI input line is forced to act as an input regardless of what is in the corresponding data direction register bit.

9.2.1 Master In Slave Out (MISO)

The MISO line is configured as an input in a master device and as an output in a slave device. It is one of the two lines that transfer serial data in one direction, with the most significant bit sent first. The MISO line of a slave device is placed in the high-impedance state if the slave is not selected.

9.2.2 Master Out Slave In (MOSI)

The MOSI line is configured as an output in a master device and an input in a slave device. It is one of the two lines that transfer serial data in one direction with the most significant bit sent first.

9.2.3 Serial Clock (SCK)

The serial clock is used to synchronize data movement both in and out of the device through its MOSI and MISO lines. The master and slave devices are capable of exchanging a byte of information during a sequence of eight clock cycles. Since SCK is generated by the master device, this line becomes an input on a slave device.

As shown in Figure 6.1, four possible timing relationships may be chosen by using control bits CPOL and CPHA in the serial peripheral control register (SPCR). Both master and slave devices must operation with the same timing. The master device always places data on the MOSI line a half-cycle before the clock edge (SCK), in order for the slave device to latch the data.

Two bits (SPR0 and SPR1) in the SPCR of the master device select the clock rate. In a slave device, SPR0 and SPR1 have no effect on the operation on the SPI.

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

The slave select ($\overline{SS}$) input line is used to select a slave device. It has to be low prior to data transactions and must stay low for the duration of transaction.

The $\overline{SS}$ line on the master must be tied high. If it goes low, a mode fault error flag (MODF) is set in the serial peripheral status register (SPSR). The $\overline{SS}$ pin can be selected to be a general-purpose output by writing a one in bit 5 of the port D data direction register, thus disabling the mode fault circuit. The other three SPI lines are dedicated to the SPI whenever the SPI is on.

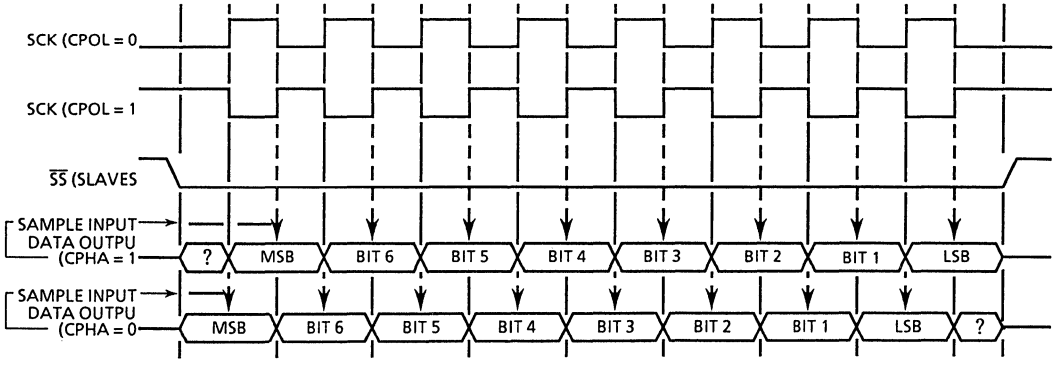


Figure 9.1 SPI Data Clock Timing Diagram

When CPHA = 0, the shift clock is the OR of $\overline{SS}$ with SCK. In this clock phase mode, $\overline{SS}$ must go high between successive characters in an SPI message. When CPHA = 1, $\overline{SS}$ may be left low for several SPI characters. In cases where there is only one SPI slave MCU, its $\overline{SS}$ line could be tied to V_{SS} as long as CPHA = 1 clock modes are used.

9.3 FUNCTIONAL DESCRIPTION

Figure 9.2 shows a block diagram of the serial peripheral interface circuitry. When a master device transmits data to a slave device via the MISO line, the slave device responds by sending data to the master device via the master's MISO line. This implies full duplex transmission with both data out and data in synchronized with the same clock signal. 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) is used to signify that the I/O operation has been completed.

The SPI is double buffered on read, but not on write. If a write is performed during data transfer, the transfer occurs uninterrupted, and the write will be unsuccessful. This condition will cause the write collision (WCOL) status bit in the SPSR to be set. After a data byte is shifted, the SPIF flag of the SPSR is set.

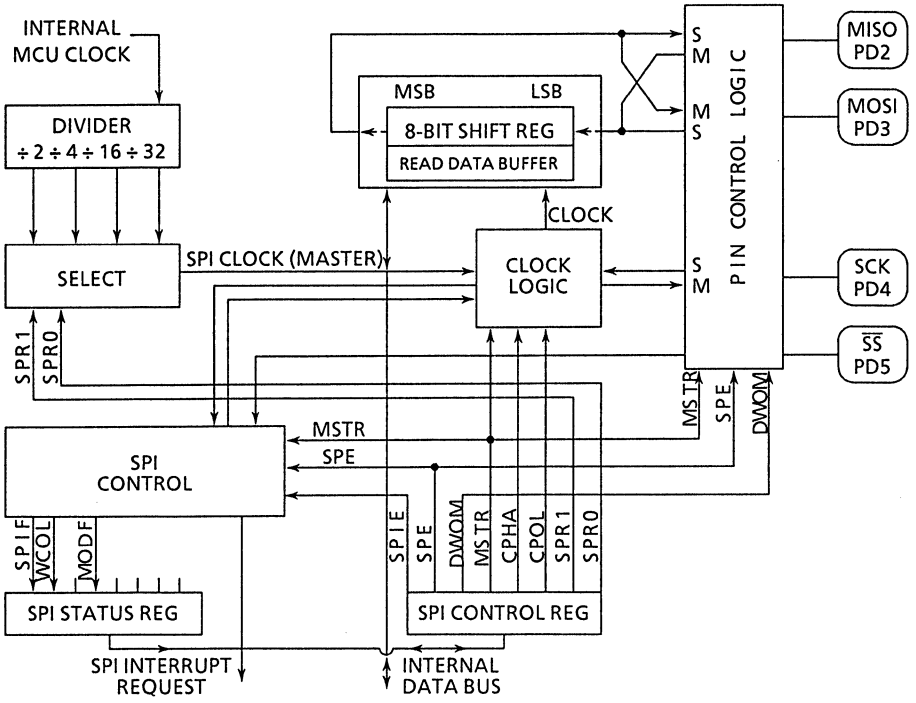


Figure 9.2 Serial Peripheral Interface Block Diagram

In the master mode, the SCK pin is an output. It idles high or low, depending on the CPOL bit in the SPCR, until data is written to the shift register, at which point eight clocks are generated to shift the eight bits of data and then SCK goes idle again.

In a slave mode, the slave start logic receives a logic low at the $\overline{SS}$ pin and a clock input at the SCK pin. Thus, the slave is synchronized with the master. Data from the master is received serially at the slave MOSI line and loads the 8-bit shift register. After the 8-bit shift register is loaded, its data is parallel transferred to the read buffer. During a write cycle, data is written into the shift register, then the slave waits for a clock train from the master to shift the data out on the slave's MISO line.

Figure 9.3 illustrates the MOSI, MISO, SCK and $\overline{SS}$ master-slave interconnections.

Due to data direction register control of SPI outputs and the port D write-OR mode (DWOM) option, the SPI system can be configured in a variety of ways. System with a single bidirectional data path rather than seperate MISO and MOSI paths can be accommodated. Since TMP68C711J6 SPI slaves can selectively disable their MISO output, a broadcast message protocol is also possible.

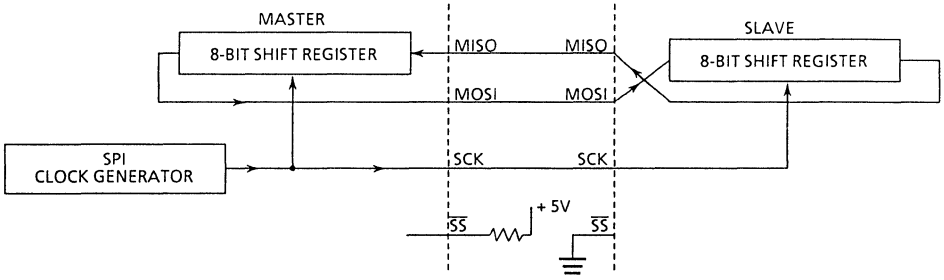


Figure 9.3 Serial Peripheral Interface Master-Slave Interconnection

9.4 SPI REGISTERS

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

9.4.1 Serial Peripheral Control Register (SPCR)

	7	6	5	4	3	2	1	0	
$1028	SPIE	SPE	DWOM	MSTR	CPOL	CPHA	SPR1	SPR0	SPCR
RESET	0	0	0	0	0	1	U	U	

SPIE - Serial Peripheral Interrupt Enable

0 = SPIF interrupts disabled

1 = SPI interrupt if SPIF = 1

SPE - Serial Peripheral System Enable

0 = SPI system off

1 = SPI system on

DWOM - Port D Wire-OR Mode Option

DWOM affects all six port D pins together.

0 = Port D outputs are normal CMOS outputs

1 = Port D outputs act as open-drain outputs

MSTR - Master Mode Select

0 = Slave mode

1 = Master mode

CPOL - Clock Polarity

When the clock polarity (CPOL) bit is cleared and data is not being transferred, a steady state low value is produced at the SCK pin of the master device. Conversely, if this bit is set, the SCK pin will idle high. This bit is also used in conjunction with the clock phase control bit to produce the desired clock-data relationship between master and slave. See Figure 9.1.

CPHA - Clock Phase

The clock phase (CPHA) bit, in conjunction with the CPOL bit, controls the clock-data relationship between master and slave. The CPOL bit can be thought of as simply inserting an inverter in series with the SCK line. The CPHA bit selects one of two fundamentally different clocking protocols. When CPHA = 0, the shift clock is the OR of SCK with $\overline{SS}$. As soon as $\overline{SS}$ goes low the transaction begins and the first edge on SCK invokes the first data sample. When CPHA = 1, the $\overline{SS}$ pin may be thought of as a simple output enable control. Refer to Figure 9.1.

SPR1 and SPR0 - SPI Clock Rate Selects

These two serial peripheral rate bits (SPR1, SPR0) select one four baud rates (Table 9.1) to be used as SCK if the device is a master; however, they have no effect in the slave mode.

Table 9.1 Serial Periperal Rate Selection

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

9.4.2 Serial Peripheral Status Register (SPSR)

	7	6	5	4	3	2	1	0	
$1029	SPIF	WCOL	0	MODF	0	0	0	0	SPSR
RESET	0	0	0	0	0	0	0	0	

SPIF - SPI Transfer Complete Flag

The serial peripheral data transfer flag (SPIF) bit is set upon completion of data transfer between the processor and external device. If SPIF bit goes high, and if SPIE bit is set, a serial peripheral interrupt is generated. Clearing the SPIF bit is accomplished by reading the SPSR register (with SPIF bit set) followed by an access of the SPDR register. Unless SPSR is read (with SPIF bit set) first, attempts to write to SPDR register are inhibited.

WCOL - Write Collision

The write collision (WCOL) bit is set when an attempt is made to write to the serial peripheral data register while data transfer is taking place. If CPHA bit is zero a transfer is said to begin when $\overline{SS}$ goes low and the transfer ends when $\overline{SS}$ goes high after eight clock cycles on SCK. When CPHA bit is one a transfer is said to begin the first time SCK becomes active while $\overline{SS}$ is low and the transfer ends when the SPIF flag gets set. Clearing the WCOL bit is accomplished by reading the SPSR register (with WCOL bit set) followed by an access to SPDR register.

Bit 5 - Not Implemented

This bit always reads zero.

MODF - Mode Fault

The mode fault flag (MODF) bit indicates that there may 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 normally clear, and is set only when the master device has its $\overline{SS}$ pin pulled low. Setting the MODF bit affects the internal serial peripheral interface system in the following ways.

1) An SPI interrupt is generated if SPIE = 1.

2) The SPE bit is cleared. This disables the SPI.

3) The MSTR bit is cleared, thus forcing the device into the slave mode.

4) DDRD bits for the four SPI pins are forced to zeros.

Clearing the MODF bit is accomplished by reading the SPSR register (with MODF set), followed by a write to the SPCR register. Control bits SPE and MSTR may be restored by user software to their original state after the MODF bit has been cleared. It is also necessary to restore DDRD bit after a mode fault.

Bits 3-0 - Not Implemented

These bits always read zero.

9.4.3 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. At the completion of transmitting a byte of data, the SPIF status bit is set in both the master and slave devices.

When the user reads the serial peripheral data I/O register, a buffer is actually being read. The first SPIF bit 98must 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. In cases of overrun the byte which causes the overrun is lost.

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

10. CPU, ADDRESSING MODES, AND INSTRUCTION SET

This section provides a description of the CPU registers, addressing modes, and a summary of the M6811 instruction set. Special operations such as subroutine calls and interrupts are described and cycle-by-cycle operations for all instructions are presented.

10.1 CPU REGISTERS

In addition to being able to execute all M6800 and M6801 instructions, the TMP68C711J6 uses a 4-page opcode map to allow execution of 91 new opcodes (see 10.2.7 Prebyte). Seven registers, discussed in the following paragraphs, are available to programmers as shown in Figure 10.1.

10.1.1 Accumulators A and B

Accumulator A and accumulator B are general-purpose 8-bit registers used to hold operands and results of arithmetic calculations or data manipulations. These two accumulators can be concatenated into a single 16-bit accumulator called the D accumulator.

10.1.2 Index Register X (IX)

The 16-bit IX register is used for indexed mode addressing. It provides a 16-bit indexing value which is added to an 8-bit offset provided in an instruction to create an effective address. The IX register can also be used as a counter or as a temporary storage register.

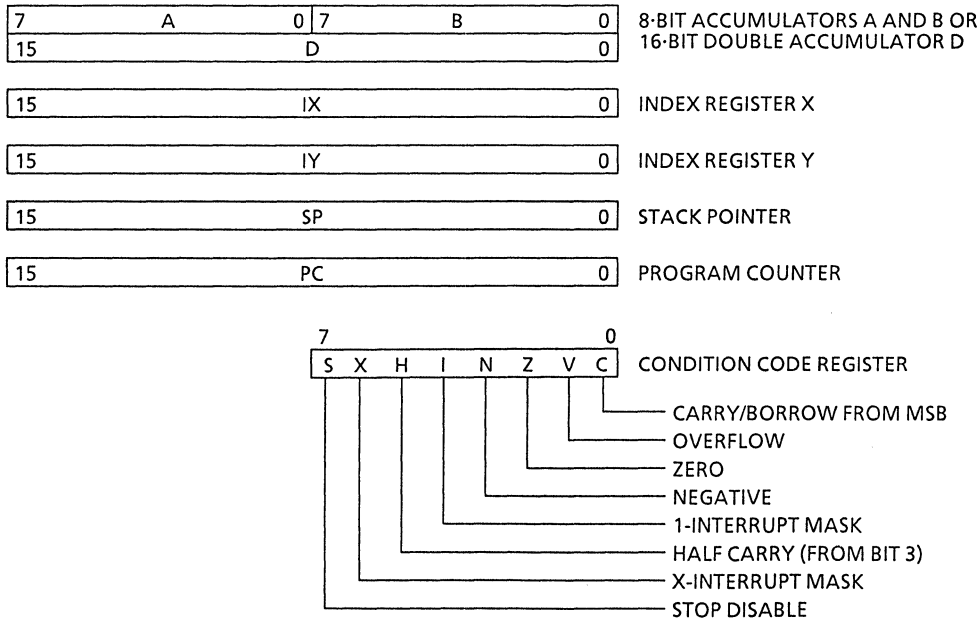


Figure 10.1 Programming Model

10.1.3 Index Register Y (IY)

The 16-bit IY register is also used for indexed mode addressing similar to the IX register;however, all instructions using the IY register require and extra byte of machine code and an extra cycle of execution time since they are two byte opcodes.

10.1.4 Stack Pointer (SP)

The stack pointer (SP) is a 16-bit register that contains the address of the next free location on the stack. The stack is configured as a sequence of last-in-first-out read/write registers which allow important data to be stored during interrupts and subroutine calls. Each time a new byte is added to the stack (a push), the SP is decremented;whereas, each time a byte is removed from the stack (a pull) the SP is incremented.

10.1.5 Program Counter (PC)

The program counter is a 16-bit register that contains the address of the next instruction to be executed.

10.1.6 Condition Code Register (CCR)

The condition code register is an 8-bit register in which each bit signifies the results of the instruction just executed. These bits can be individually tested by a program and a specific action can be taken as a result of the test. Each individual condition code register bit is explained below.

10.1.6.1 Carry/Borrow (C).

The C bit is set if there was a carry or borrow out of the arithmetic logic unit (ALU) during the last arithmetic operation. The C bit is also affected during shift and rotate instructions.

10.1.6.2 Overflow (V).

The overflow bit is set if there was an arithmetic overflow as a result of the operation; otherwise, the V bit is cleared.

10.1.6.3 Zero (Z).

The zero bit is set if the result of the last arithmetic, logic, or data manipulation operation was zero;otherwise, the Z bit is cleared.

10.1.6.4 Negative (N).

The negative bit is set if the result of the last arithmetic, logic, or data manipulation operation was negative;otherwise, the N bit is cleared. A result is said to be negative if its most significant bit is a one.

10.1.6.5 Interrupt Mask (I).

The I interrupt mask bit is set either by hardware or program instruction to disable (mask) all maskable interrupt sources (both external and internal).

10.1.6.6 Half Carry (H).

The half carry bit is set to a logic one when a carry occurs between bits 3 and 4 of the arithmetic logic unit during an ADD, ABA, or ADC instruction;otherwise, the H bit is cleared.

10.1.6.7 X interrupt Mask (X).

The X interrupt mask bit is set only by hardware ($\overline{\text{RESET}}$ or $\overline{\text{XIRQ}}$ acknowledge);and it is cleared only by program instruction (TAP or RTI).

10.1.6.8 Stop Disable (S).

The stop disable bit is set to disable the STOP instruction, and cleared to enable the STOP instruction. The S bit is program controlled. The STOP instruction is treated as no operation (NOP) if the S bit is set.

10.2 ADDRESSING MODES

Six addressing modes can be used to reference memory;they include:immediate, direct, extended, indexed (with either of two 16-bit index registers and an 8-bit offset), inherent and relative. Some instructions require and additional byte before the opcode to accommodate a multi-page opcode map;this byte is called a prebyte.

The following paragraphs provide a description of each addressing mode plus a discussion of the prebyte. In these descriptions the term effective address is used to indicate the address in memory from which the argument is fetched or stored, or from which execution is to proceed.

10.2.1 Immediate Addressing

In the immediate addressing mode, the actual argument is contained in the byte (s) immediately following the instruction, where the number of bytes matches the size of the register. These are two, three, or four (if prebyte is required) byte instructions.

10.2.2 Direct Addressing

In the direct addressing mode (sometimes called zero page addressing), the least significant byte of the operand address is contained in a single byte following the opcode and the most significant byte is assumed to be $00. Direct addressing allows the user to access $0000 through $00FF using two byte instructions and execution time is reduced by eliminating the additional memory access. In most applications, this 512-byte area is reserved for frequently referenced data. In the TMP68C711J6, software can configure the memory map so that internal RAM, and/or internal registers, or external memory space can occupy these addresses.

10.2.3 Extended Addressing

In the extended addressing mode, the second and third bytes (following the opcode) contain the absolute address of the operand. These are three or four (if prebyte is required) byte instructions: one or two for the opcode, and two for the effective address.

10.2.4 Indexed Addressing

In the indexed addressing mode, one of the index registers (X or Y) is used in calculating the effective address. In this case, the effective address is variable and depends on two factors: 1) the current contents of the index register (X or Y) being used, and 2) the 8-bit unsigned offset contained in the instruction. This addressing mode allows referencing any memory location in the 64 K byte address space. These are usually two or three (if prebyte is required) byte instructions, the opcode plus the 8-bit offset.

10.2.5 Inherent Addressing

In the inherent addressing mode, all of the information is contained in the opcode. The operands (if any) are registers and no memory reference is required. These are usually one or two byte instructions.

10.2.6 Relative Addressing

The relative addressing mode is used for branch instructions. If the branch condition is true, the contents of the 8-bit signed byte following the opcode (the offset) is added to the contents of the program counter to form the effective branch address;otherwise, control proceeds to the next instruction. These are usually two byte instructions.

10.2.7 Prebyte

In order to expand the number of instructions used in the TMP68C711J6, a prebyte instruction has been added to certain instructions. The instructions affected are usually associated with index register Y. The instruction opcodes which do not require a prebyte could be considered as page 1 of the overall opcode map. The remaining opcodes could be considered as pages 2, 3, and 4 of the opcode map and would require a prebyte;$18 for page 2, $1A for page 3, and $CD for page 4.

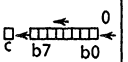
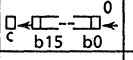
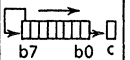
10.3 INSTRUCTION SET

The central processing unit (CPU) in the TMP68C711J6 is basically a proper extension of the MC6801 CPU. In addition to its ability to execute all M6800 and M6801 instructions, the TMP68C711J6 CPU has a paged operation code (opcode) map with a total of 91 new opcodes. Major functional additions include a second 16-bit index register (Y register), two types of 16-by-16 divide instruction, STOP and WAIT instructions, and bit manipulation instructions.

Table 10.1 shows all TMP68C711J6 instructions in all possible addressing modes. For each instruction, the operand construction is shown as well as the total number of machine code bytes and execution time in CPU E-clock cycles. Notes are provided at the end of Table 10.1 which explain the letters in the Operand and Execution Time columns for some instructions. Definitions of "Special Ops" found in the Boolean Expression column are found in Figure 10.2.

Table 10.2 through 10.8 prdovide a detailed description of the information present on the address bus, data bus, and the read/write ($R/\overline{W}$) line during each cycle of each instruction. The information is useful in comparing actual with expected results during debug of both software and hardware as the program is executed. The information is categorized in groups according to addressing mode and number of cycles per instruction. In general, instructions with the same address mode and number of cycles execute in the same manner. Exceptions are indicated in the table.

Table 10.1 TMP68C711J6 Instructions, Addressing Modes, and Execution Times (Sheet 1 of 7)

Source Form (s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes
				Opcode	Operand (s)				S X H I N Z V C
ABA	Add Accumulators	A + B→A	INH	1B		1	2	2-1	- - ↕ - ↕ ↕ ↕ ↕
ABX	Add B to X	IX + 00 : B→IX	INH	3A		1	3	2-2	- - - - - - - -
ABY	Add B to Y	IY + 00 : B→IY	INH	18 3A		2	4	2-4	- - - - - - - -
ADCA (opr)	Add with Carry to A	A + M + C→A	A IMM	89	ii	2	2	3-1	- - ↕ - ↕ ↕ ↕ ↕
			A DIR	99	dd	2	3	4-1	
			A EXT	B9	hh ll	3	4	5-2	
			A IND, X	A9	ff	2	4	6-2	
			A IND, Y	18 A9	ff	3	5	7-2	
ADCB (opr)	Add with Carry to B	B + M + C→B	B IMM	C9	ii	2	2	3-1	- - ↕ - ↕ ↕ ↕ ↕
			B DIR	D9	dd	2	3	4-1	
			B EXT	F9	hh ll	3	4	5-2	
			B IND, X	E9	ff	2	4	6-2	
			B IND, Y	18 E9	ff	3	5	7-2	
ADDA (opr)	Add Memory to A	A + M→A	A IMM	8B	ii	2	2	3-1	- - ↕ - ↕ ↕ ↕ ↕
			A DIR	9B	dd	2	3	4-1	
			A EXT	BB	hh ll	3	4	5-2	
			A IND, X	AB	ff	2	4	6-2	
			A IND, Y	18 AB	ff	3	5	7-2	
ADDB (opr)	Add Memory to B	B + M→B	B IMM	CB	ii	2	2	3-1	- - ↕ - ↕ ↕ ↕ ↕
			B DIR	DB	dd	2	3	4-1	
			B EXT	FB	hh ll	3	4	5-2	
			B IND, X	EB	ff	2	4	6-2	
			B IND, Y	18 EB	ff	3	5	7-2	
ADDD (opr)	Add 16-Bit to D	D + M : M + 1 →D	IMM	C3	jj kk	3	4	3-3	- - - - ↕ ↕ ↕ ↕
			DIR	D3	dd	2	5	4-7	
			EXT	F3	hh ll	3	6	5-10	
			IND, X	E3	ff	2	6	6-10	
			IND, Y	18 E3	ff	3	7	7-8	
ANDA (opr)	AND A with Memory	A • M→A	A IMM	84	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			A DIR	94	dd	2	3	4-1	
			A EXT	B4	hh ll	3	4	5-2	
			A IND, X	A4	ff	2	4	6-2	
			A IND, Y	18 A4	ff	3	5	7-2	
ANDB (opr)	AND B with Memory	B • M→B	B IMM	C4	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			B DIR	D4	dd	2	3	4-1	
			B EXT	F4	hh ll	3	4	5-2	
			B IND, X	E4	ff	2	4	6-2	
			B IND, Y	18 E4	ff	3	5	7-2	
ASL (opr)	Arithmetic Shift Left		EXT	78	hh ll	3	6	5-8	- - - - ↕ ↕ ↕ ↕
			IND, X	68	ff	2	6	6-3	
			IND, Y	18 68	ff	3	7	7-3	
ASLA			A INH	48		1	2	2-1	
ASLB			B INH	58		1	2	2-1	
ASLD	Arithmetic Shift Left Double		INH	05		1	3	2-2	- - - - ↕ ↕ ↕ ↕
ASR (opr)	Arithmetic Shift Right		EXT	77	hh ll	3	6	5-8	- - - - ↕ ↕ ↕ ↕
			IND, X	67	ff	2	6	6-3	
			IND, Y	18 67	ff	3	7	7-3	
ASRA			A INH	47		1	2	2-1	
ASRB			B INH	57		1	2	2-1	
BCC (rel)	Branch if Carry Clear	?C = 0	REL	24	rr	2	3	8-1	- - - - - - - -
BCLR (opr) (msk)	Clear Bit (s)	M • ($\overline{mm}$) →M	DIR	15	dd mm	3	6	4-10	- - - - ↕ ↕ 0 -
			IND, X	1D	ff mm	3	7	6-13	
			IND, Y	18 1D	ff mm	4	8	7-10	
BCS (rel)	Branch if Carry Set	?C = 1	REL	25	rr	2	3	8-1	- - - - - - - -
BEQ (rel)	Branch if = Zero	?Z = 1	REL	27	rr	2	3	8-1	- - - - - - - -

* Cycle-by-cycle number provides a reference to Table 10.2 through 10.8 which detail cycle-by-cycle operation.
Example: Table 10.1 cycle-by-cycle column reference number 2.4 equals Table 10.2 line item 2.4.

Table 10.1 TMP68C711J6 Instructions, Addressing Modes, and Execution Times (Sheet 2 of 7)

Source Form (s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes
				Opcode	Operand (s)				S X H I N Z V C
BGE (rel)	Branch if ≧ Zero	?N⊕V = 0	REL	2C	rr	2	3	8-1	- - - - - - - -
BGT (rel)	Branch if > Zero	?Z + (N⊕V) = 0	REL	2E	rr	2	3	8-1	- - - - - - - -
BHI (rel)	Branch if Higher	?C + Z = 0	REL	22	rr	2	3	8-1	- - - - - - - -
BHS (rel)	Branch if Higher or Same	?C = 0	REL	24	rr	2	3	8-1	- - - - - - - -
BITA (opr)	Bit (s) Test A with Memory	A • M	A IMM	85	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			A DIR	95	dd	2	3	4-1	
			A EXT	B5	hh ll	3	4	5-2	
			A IND, X	A5	ff	2	4	6-2	
			A IND, Y	18 A5	ff	3	5	7-2	
BITB (opr)	Bit (s) Test B with Memory	B • M	B IMM	C5	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			B DIR	D5	dd	2	3	4-1	
			B EXT	F5	hh ll	3	4	5-2	
			B IND, X	E5	ff	2	4	6-2	
			B IND, Y	18 E5	ff	3	5	7-2	
BLE (rel)	Branch if ≦ Zero	?Z + (N⊕V) = 1	REL	2F	rr	2	3	8-1	- - - - - - - -
BLO (rel)	Branch if Lower	?C = 1	REL	25	rr	2	3	8-1	- - - - - - - -
BLS (rel)	Branch if Lower or Same	?C + Z = 1	REL	23	rr	2	3	8-1	- - - - - - - -
BLT (rel)	Branch if < Zero	?N⊕V = 1	REL	2D	rr	2	3	8-1	- - - - - - - -
BMI (rel)	Branch if Minus	?N = 1	REL	2B	rr	2	3	8-1	- - - - - - - -
BNE (rel)	Branch if Not = Zero	?Z = 0	REL	26	rr	2	3	8-1	- - - - - - - -
BPL (rel)	Branch if Plus	?N = 0	REL	2A	rr	2	3	8-1	- - - - - - - -
BRA (rel)	Branch Always	?1 = 1	REL	20	rr	2	3	8-1	- - - - - - - -
BRCLR (opr) (msk) (rel)	Branch if Bit (s) Clear	?M • mm = 0	DIR	13	dd mm rr	4	6	4-11	- - - - - - - -
			IND, X	1F	ff mm rr	4	7	6-14	
			IND, Y	18 1F	ff mm rr	5	8	7-11	
BRN (rel)	Branch Never	?1 = 0	REL	21	rr	2	3	8-1	- - - - - - - -
BRSET (opr) (msk) (rel)	Branch if Bit (s) Set	? ($\overline{M}$) • mm = 0	DIR	12	dd mm rr	4	6	4-11	- - - - - - - -
			IND, X	1E	ff mm rr	4	7	6-14	
			IND, Y	18 1E	ff mm rr	5	8	7-11	
BSET (opr) (msk)	Set Bit (s)	M + mm→M	DIR	14	dd mm	3	6	4-10	- - - - ↕ ↕ 0 -
			IND, X	1C	ff mm	3	7	6-13	
			IND, Y	18 1C	ff mm	4	8	7-10	
BSR (rel)	Branch to Subroutine	See Special Ops	REL	8D	rr	2	6	8-2	- - - - - - - -
BVC (rel)	Branch if Overflow Clear	?V = 0	REL	28	rr	2	3	8-1	- - - - - - - -
BVS (rel)	Branch if Overflow Set	?V = 1	REL	29	rr	2	3	8-1	- - - - - - - -
CBA	Compare A to B	A-B	INH	11		1	2	2-1	- - - - ↕ ↕ ↕ ↕
CLC	Clear Carry Bit	0→C	INH	0C		1	2	2-1	- - - - - - - 0
CLI	Clear Interrupt Mask	0→I	INH	0E		1	2	2-1	- - - 0 - - - -
CLR (opr)	Clear Memory Byte	0→M	EXT	7F	hh ll	3	6	5-8	- - - - 0 1 0 0
			IND, X	6F	ff	2	6	6-3	
			IND, Y	18 6F	ff	3	7	7-3	
CLRA	Clear Accumulator A	0→A	A INH	4F		1	2	2-1	- - - - 0 1 0 0
CLRB	Clear Accumulator B	0→B	B INH	5F		1	2	2-1	- - - - 0 1 0 0
CLV	Clear Overflow Flag	0→V	INH	0A		1	2	2-1	- - - - - - 0 -
CMPA (opr)	Compare A to Memory	A-M	A IMM	81	ii	2	2	3-1	- - - - ↕ ↕ ↕ ↕
			A DIR	91	dd	2	3	4-1	
			A EXT	B1	hh ll	3	4	5-2	
			A IND, X	A1	ff	2	4	6-2	
			A IND, Y	18 A1	ff	3	5	7-2	

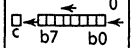
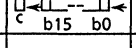
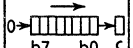
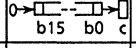
* Cycle-by-cycle number provides a reference to Table 10.2 through 10.8 which detail cycle-by-cycle operation.
Example: Table 10.1 cycle-by-cycle column reference number 2.4 equals Table 10.2 line item 2.4.

Table 10.1 TMP68C711J6 Instructions, Addressing Modes, and Execution Times (Sheet 3 of 7)

Source Form (s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes
				Opcode	Operand (s)				S X H I N Z V C
CMPB (opr)	Compare B to Memory	B-M	B IMM	C1	ii	2	2	3-1	- - - - ↕ ↕ ↕ ↕
			B DIR	D1	dd	2	3	4-1	
			B EXT	F1	hh ll	3	4	5-2	
			B IND, X	E1	ff	2	4	6-2	
			B IND, Y	18 E1	ff	3	5	7-2	
COM(opr)	1's Complement Memory Byte	$FF-M→M	EXT	73	hh ll	3	6	5-8	- - - - ↕ ↕ 0 1
			IND, X	63	ff	2	6	6-3	
			IND, Y	18 63	ff	3	7	7-3	
COMA	1's Complement A	$FF-A→A	A INH	43		1	2	2-1	- - - - ↕ ↕ 0 1
COMB	1's Complement B	$FF-B→B	B INH	53		1	2	2-1	- - - - ↕ ↕ 0 1
CPD (opr)	Compare D to Memory 16-Bit	D-M : M + 1	IMM	1A 83	jj kk	4	5	3-5	- - - - ↕ ↕ ↕ ↕
			DIR	1A 93	dd	3	6	4-9	
			EXT	1A B3	hh ll	4	7	5-11	
			IND, X	1A A3	ff	3	7	6-11	
			IND, Y	CD A3	ff	3	7	7-8	
CPX (opr)	Compare X to Memory 16-Bit	IX-M : M + 1	IMM	8C	jj kk	3	4	3-3	- - - - ↕ ↕ ↕ ↕
			DIR	9C	dd	2	5	4-7	
			EXT	BC	hh ll	3	6	5-10	
			IND, X	AC	ff	2	6	6-10	
			IND, Y	CD AC	ff	3	7	7-8	
CPY (opr)	Compare Y to Memory 16-Bit	IY-M : M + 1	IMM	18 8C	jj kk	4	5	3-5	- - - - ↕ ↕ ↕ ↕
			DIR	18 9C	dd	3	6	4-9	
			EXT	18 BC	hh ll	4	7	5-11	
			IND, X	1A AC	ff	3	7	6-11	
			IND, Y	18 AC	ff	3	7	7-8	
DAA	Decimal Adjust A	Adjust Sum to BCD	INH	19		1	2	2-1	- - - - ↕ ↕ ↕ ↕
DEC (opr)	Decrement Memory Byte	M-1→M	EXT	7A	hh ll	3	6	5-8	- - - - ↕ ↕ ↕ -
			IND, X	6A	ff	2	6	6-3	
			IND, Y	18 6A	ff	3	7	7-3	
DECA	Decrement Accumulator A	A-1→A	A INH	4A		1	2	2-1	- - - - ↕ ↕ ↕ -
DECB	Decrement Accumulator B	B-1→B	B INH	5A		1	2	2-1	- - - - ↕ ↕ ↕ -
DES	Decrement Stack Pointer	SP-1→SP	INH	34		1	3	2-3	- - - - - - - -
DEX	Decrement Index Register X	IX-1→IX	INH	09		1	3	2-2	- - - - - ↕ - -
DEY	Decrement Index Register Y	IY-1→IY	INH	18 09		2	4	2-4	- - - - - ↕ - -
EORA (opr)	Exclusive OR A with Memory	A ⊕ M→A	A IMM	88	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			A DIR	98	dd	2	3	4-1	
			A EXT	B8	hh ll	3	4	5-2	
			A IND, X	A8	ff	2	4	6-2	
			A IND, Y	18 A8	ff	3	5	7-2	
EORB (opr)	Exclusive OR B with Memory	B ⊕ M→B	B IMM	C8	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			B DIR	D8	dd	2	3	4-1	
			B EXT	F8	hh ll	3	4	5-2	
			B IND, X	E8	ff	2	4	6-2	
			B IND, Y	18 E8	ff	3	5	7-2	
FDIV	Fractional Divide 16 by 16	D / IX→IX ; r→D	INH	03		1	41	2-17	- - - - - ↕ ↕ ↕
IDIV	Integer Divide 16 by 16	D / IX→IX ; r→D	INH	02		1	41	2-17	- - - - - ↕ 0 ↕
INC (opr)	Increment Memory Byte	M + 1→M	EXT	7C	hh ll	3	6	5-8	- - - - ↕ ↕ ↕ -
			IND, X	6C	ff	2	6	6-3	
			IND, Y	18 6C	ff	3	7	7-3	
INCA	Increment Accumulator A	A + 1→A	A INH	4C		1	2	2-1	- - - - ↕ ↕ ↕ -
INCB	Increment Accumulator B	B + 1→B	B INH	5C		1	2	2-1	- - - - ↕ ↕ ↕ -
INS	Increment Stack Pointer	SP + 1→SP	INH	31		1	3	2-3	- - - - - - - -

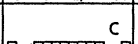
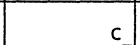
* Cycle-by-cycle number provides a reference to Table 10.2 through 10.8 which detail cycle-by-cycle operation.
Example: Table 10.1 cycle-by-cycle column reference number 2.4 equals Table 10.2 line item 2.4.

Table 10.1 TMP68C711J6 Instructions, Addressing Modes, and Execution Times (Sheet 4 of 7)

Source Form (s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes
				Opcode	Operand (s)				S X H I N Z V C
INX	Increment Index Register X	IX + 1→IX	INH	08		1	3	2-2	- - - - - ↕ - -
INY	Increment Index Register Y	IY + 1→IY	INH	18 08		2	4	2-4	- - - - - ↕ - -
JMP (opr)	Jump	See Special Ops	EXT	7E	hh ll	3	3	5-1	- - - - - - - -
			IND, X	6E	ff	2	3	6-1	
			IND, Y	18 6E	ff	3	4	7-1	
JSR (opr)	Jump to Subroutine	See Special Ops	DIR	9D	dd	2	5	4-8	- - - - - - - -
			EXT	BD	hh ll	3	6	5-12	
			IND, X	AD	ff	2	6	6-12	
			IND, Y	18 AD	ff	3	7	7-9	
LDAA (opr)	Load Accumulator A	M→A	A IMM	86	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			A DIR	96	dd	2	3	4-1	
			A EXT	B6	hh ll	3	4	5-2	
			A IND, X	A6	ff	2	4	6-2	
			A IND, Y	18 A6	ff	3	5	7-2	
LDAB (opr)	Load Accumulator B	M→B	B IMM	C6	ii	2	2	3-1	- - - - ↕ ↕ 0 -
			B DIR	D6	dd	2	3	4-1	
			B EXT	F6	hh ll	3	4	5-2	
			B IND, X	E6	ff	2	4	6-2	
			B IND, Y	18 E6	ff	3	5	7-2	
LDD (opr)	Load Double Accumulator D	M→A, M + 1→B	IMM	CC	jj kk	3	3	3-2	- - - - ↕ ↕ 0 -
			DIR	DC	dd	2	4	4-3	
			EXT	FC	hh ll	3	5	5-4	
			IND, X	EC	ff	2	5	6-6	
			IND, Y	18 EC	ff	3	6	7-6	
LDS (opr)	Load Stack Pointer	M : M + 1→SP	IMM	8E	jj kk	3	3	3-2	- - - - ↕ ↕ 0 -
			DIR	9E	dd	2	4	4-3	
			EXT	BE	hh ll	3	5	5-4	
			IND, X	AE	ff	2	5	6-6	
			IND, Y	18 AE	ff	3	6	7-6	
LDX (opr)	Load Index Register X	M : M + 1→IX	IMM	CE	jj kk	3	3	3-2	- - - - ↕ ↕ 0 -
			DIR	DE	dd	2	4	4-3	
			EXT	FE	hh ll	3	5	5-4	
			IND, X	EE	ff	2	5	6-6	
			IND, Y	CD EE	ff	3	6	7-6	
LDY (opr)	Load Index Register Y	M : M + 1→IY	IMM	18 CE	jj kk	4	4	3-4	- - - - ↕ ↕ 0 -
			DIR	18 DE	dd	3	5	4-5	
			EXT	18 FE	hh ll	4	6	5-6	
			IND, X	1A EE	ff	3	6	6-7	
			IND, Y	18 EE	ff	3	6	7-6	
LSL (opr)	Logical Shift Left		EXT	78	hh ll	3	6	5-8	- - - - ↕ ↕ ↕ ↕
			IND, X	68	ff	2	6	6-3	
			IND, Y	18 68	ff	3	7	7-3	
LSLA			A INH	48		1	2	2-1	
LSLB			A INH	58		1	2	2-1	
LSLD	Logical Shift Left Double		INH	05		1	3	2-2	- - - - ↕ ↕ ↕ ↕
LSR (opr)	Logical Shift Right		EXT	74	hh ll	3	6	5-8	- - - - 0 ↕ ↕ ↕
			IND, X	64	ff	2	6	6-3	
			IND, Y	18 64	ff	3	7	7-3	
LSRA			A INH	44		1	2	2-1	
LSRB			B INH	54		1	2	2-1	
LSRD	Logical Shift Right Double		INH	04		1	3	2-2	- - - - 0 ↕ ↕ ↕
MUL	Multiply 8 by 8	A x B→D	INH	3D		1	10	2-13	- - - - - - - ↕

* Cycle-by-cycle number provides a reference to Table 10.2 through 10.8 which detail cycle-by-cycle operation.
Example: Table 10.1 cycle-by-cycle column reference number 2.4 equals Table 10.2 line item 2.4.

Table 10.1 TMP68C711J6 Instructions, Addressing Modes, and Execution Times (Sheet 5 of 7)

Source Form (s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes							
				Opcode	Operand (s)				S	X	H	I	N	Z	V	C
NEG (opr)	2's Complement Memory Byte	0 – M→M	EXT	70	hh ll	3	6	5-8	-	-	-	-	↕	↕	↕	↕
			IND, X	60	ff	2	6	6-3								
			IND, Y	18 60	ff	3	7	7-3								
NEGA	2's Complement A	0 – A→A	A INH	40		1	2	2-1	-	-	-	-	↕	↕	↕	↕
NEGB	2's Complement B	0 – B→B	B INH	50		1	2	2-1	-	-	-	-	↕	↕	↕	↕
NOP	No Operation	No Operation	INH	01		1	2	2-1	-	-	-	-	-	-	-	-
ORAA (opr)	OR Accumulator A (Inclusive)	A + M→A	A IMM	8A	ii	2	2	3-1	-	-	-	-	↕	↕	0	-
			A DIR	9A	dd	2	3	4-1								
			A EXT	BA	hh ll	3	4	5-2								
			A IND, X	AA	ff	2	4	6-2								
			A IND, Y	18 AA	ff	3	5	7-2								
ORAB (opr)	OR Accumulator B (Inclusive)	B + M→B	B IMM	CA	ii	2	2	3-1	-	-	-	-	↕	↕	0	-
			B DIR	DA	dd	2	3	4-1								
			B EXT	FA	hh ll	3	4	5-2								
			B IND, X	EA	ff	2	4	6-2								
			B IND, Y	18 EA	ff	3	5	7-2								
PSHA	Push A onto Stack	A→Stk, SP = SP – 1	A INH	36		1	3	2-6	-	-	-	-	-	-	-	-
PSHB	Push B onto Stack	B→Stk, SP = SP – 1	B INH	37		1	3	2-6	-	-	-	-	-	-	-	-
PSHX	Push X onto Stack (Lo First)	IX→Stk, SP = SP – 2	INH	3C		1	4	2-7	-	-	-	-	-	-	-	-
PSHY	Push Y onto Stack (Lo First)	IY→Stk, SP = SP – 2	INH	18 3C		2	5	2-8	-	-	-	-	-	-	-	-
PULA	Pull A from Stack	SP = SP + 1, A←Stk	A INH	32		1	4	2-9	-	-	-	-	-	-	-	-
PULB	Pull B from Stack	SP = SP + 1, B←Stk	B INH	33		1	4	2-9	-	-	-	-	-	-	-	-
PULX	Pull X from Stack (Hi First)	SP = SP + 2, IX←Stk	INH	38		1	5	2-10	-	-	-	-	-	-	-	-
PULY	Pull Y from Stack (Hi First)	SP = SP + 2, IY←Stk	INH	18 38		2	6	2-11	-	-	-	-	-	-	-	-
ROL (opr)	Rotate Left	 C, c b7 ← b0	EXT	79	hh ll	3	6	5-8	-	-	-	-	↕	↕	↕	↕
			IND, X	69	ff	2	6	6-3								
			IND, Y	18 69	ff	3	7	7-3								
ROLA			A INH	49		1	2	2-1								
ROLB			B INH	59		1	2	2-1								
ROR (opr)	Rotate Right	 C, c b7 → b0	EXT	76	hh ll	3	6	5-8	-	-	-	-	↕	↕	↕	↕
			IND, X	66	ff	2	6	6-3								
			IND, Y	18 66	ff	3	7	7-3								
RORA			A INH	46		1	2	2-1								
RORB			B INH	56		1	2	2-1								
RTI	Return from Interrupt	See Special Ops	INH	3B		1	12	2-14	↕	↕	↕	↕	↕	↕	↕	↕
RTS	Return from Subroutine	See Special Ops	INH	39		1	5	2-12	-	-	-	-	-	-	-	-
SBA	Subtract B from A	A – B→A	INH	10		1	2	2-1	-	-	-	-	↕	↕	↕	↕
SBCA (opr)	Subtract with Carry from A	A – M – C→A	A IMM	82	ii	2	2	3-1	-	-	-	-	↕	↕	↕	↕
			A DIR	92	dd	2	3	4-1								
			A EXT	B2	hh ll	3	4	5-2								
			A IND, X	A2	ff	2	4	6-2								
			A IND, Y	18 A2	ff	3	5	7-2								
SBCB (opr)	Subtract from Carry from B	B – M – C→B	B IMM	C2	ii	2	2	3-1	-	-	-	-	↕	↕	↕	↕
			B DIR	D2	dd	2	3	4-1								
			B EXT	F2	hh ll	3	4	5-2								
			B IND, X	E2	ff	2	4	6-2								
			B IND, Y	18 E2	ff	3	5	7-2								
SEC	Set Carry	1→C	INH	0D		1	2	2-1	-	-	-	-	-	-	-	1
SEI	Set Interrupt Mask	1→I	INH	0F		1	2	2-1	-	-	-	1	-	-	-	-
SEV	Set Overflow Flag	1→V	INH	0B		1	2	2-1	-	-	-	-	-	-	1	-

* Cycle-by-cycle number provides a reference to Table 10.2 through 10.8 which detail cycle-by-cycle operation.
Example: Table 10.1 cycle-by-cycle column reference number 2.4 equals Table 10.2 line item 2.4.

Table 10.1 TMP68C711J6 Instructions, Addressing Modes, and Execution Times (Sheet 6 of 7)

Source Form (s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand (s)	Bytes	Cycle	Cycle by Cycle*	Condition Codes S X H I N Z V C
STAA (opr)	Store Accumulator A	A→M	A DIR	97	dd	2	3	4-2	- - - - ↕ ↕ 0 -
			A EXT	B7	hh ll	3	4	5-3	
			A IND, X	A7	ff	2	4	6-5	
			A IND, Y	18 A7	ff	3	5	7-5	
STAB (opr)	Store Accumulator B	B→M	B DIR	D7	dd	2	3	4-2	- - - - ↕ ↕ 0 -
			B EXT	F7	hh ll	3	4	5-3	
			B IND, X	E7	ff	2	4	6-5	
			B IND, Y	18 E7	ff	3	5	7-5	
STD (opr)	Store Accumulator D	A→M, B→M + 1	DIR	DD	dd	2	4	4-4	- - - - ↕ ↕ 0 -
			EXT	FD	hh ll	3	5	5-5	
			IND, X	ED	ff	2	5	6-8	
			IND, Y	18 ED	ff	3	6	7-7	
STOP	Stop Internal Clocks		INH	CF		1	2	2-1	- - - - - - - -
STS (opr)	Store Stack Pointer	SP→M : M + 1	DIR	9F	dd	2	4	4-4	- - - - ↕ ↕ 0 -
			EXT	BF	hh ll	3	5	5-5	
			IND, X	AF	ff	2	5	6-8	
			IND, Y	18 AF	ff	3	6	7-7	
STX (opr)	Store Index Register X	IX→M : M + 1	DIR	DF	dd	2	4	4-4	- - - - ↕ ↕ 0 -
			EXT	FF	hh ll	3	5	5-5	
			IND, X	EF	ff	2	5	6-8	
			IND, Y	CD EF	ff	3	6	7-7	
STY (opr)	Store Index Register Y	IY→M : M + 1	DIR	18 DF	dd	3	5	4-6	- - - - ↕ ↕ 0 -
			EXT	18 FF	hh ll	4	6	5-7	
			IND, X	1A EF	ff	3	6	6-9	
			IND, Y	18 EF	ff	3	6	7-7	
SUBA (opr)	Subtract Memory from A	A – M→A	A IMM	80	ii	2	2	3-1	- - - - ↕ ↕ ↕ ↕
			A DIR	90	dd	2	3	4-1	
			A EXT	B0	hh ll	3	4	5-2	
			A IND, X	A0	ff	2	4	6-2	
			A IND, Y	18 A0	ff	3	5	7-2	
SUBB (opr)	Subtract Memory from B	B – M→B	B IMM	C0	ii	2	2	3-1	- - - - ↕ ↕ ↕ ↕
			B DIR	D0	dd	2	3	4-1	
			B EXT	F0	hh ll	3	4	5-2	
			B IND, X	E0	ff	2	4	6-2	
			B IND, Y	18 E0	ff	3	5	7-2	
SUBD (opr)	Subtract Memory from D	D – M : M + 1→D	IMM	83	jj kk	3	4	3-3	- - - - ↕ ↕ ↕ ↕
			DIR	93	dd	2	5	4-7	
			EXT	B3	hh ll	3	6	5-10	
			IND, X	A3	ff	2	6	6-10	
			IND, Y	18 A3	ff	3	7	7-8	
SWI	Software Interrupt	See Special Ops	INH	3F		1	14	2-15	- - - 1 - - - -
TAB	Transfer A to B	A→B	INH	16		1	2	2-1	- - - - ↕ ↕ 0 -
TAP	Transfer A to CC Register	A→CCR	INH	06		1	2	2-1	↕ ↕ ↕ ↕ ↕ ↕ ↕ ↕
TBA	Transfer B to A	B→A	INH	17		1	2	2-1	- - - - ↕ ↕ 0 -
TEST	TEST (Only in Test Modes)	Address Bus Counts	INH	00		1	**	2-20	- - - - - - - -
TPA	Transfer CC Register to A	CCR→A	INH	07		1	2	2-1	- - - - - - - -
TST (opr)	Test for Zero or Minus	M – 0	EXT	7D	hh ll	3	6	5-9	- - - - ↕ ↕ 0 0
			IND, X	6D	ff	2	6	6-4	
			IND, Y	18 6D	ff	3	7	7-4	
TSTA		A – 0	A INH	4D		1	2	2-1	- - - - ↕ ↕ 0 0
TSTB		B – 0	B INH	5D		1	2	2-1	- - - - ↕ ↕ 0 0
TSX	Transfer Stack Pointer to X	SP + 1→IX	INH	30		1	3	2-3	- - - - - - - -
TSY	Transfer Stack Pointer to Y	SP + 1→IY	INH	18 30		2	4	2-5	- - - - - - - -

* Cycle-by-cycle number provides a reference to Table 10.2 through 10.8 which detail cycle-by-cycle operation.

Example: Table 10.1 cycle-by-cycle column reference number 2.4 equals Table 10.2 line item 2.4.

Table 10.1 TMP68C711J6 Instructions, Addressing Modes, and Execution Times (Sheet 7 of 7)

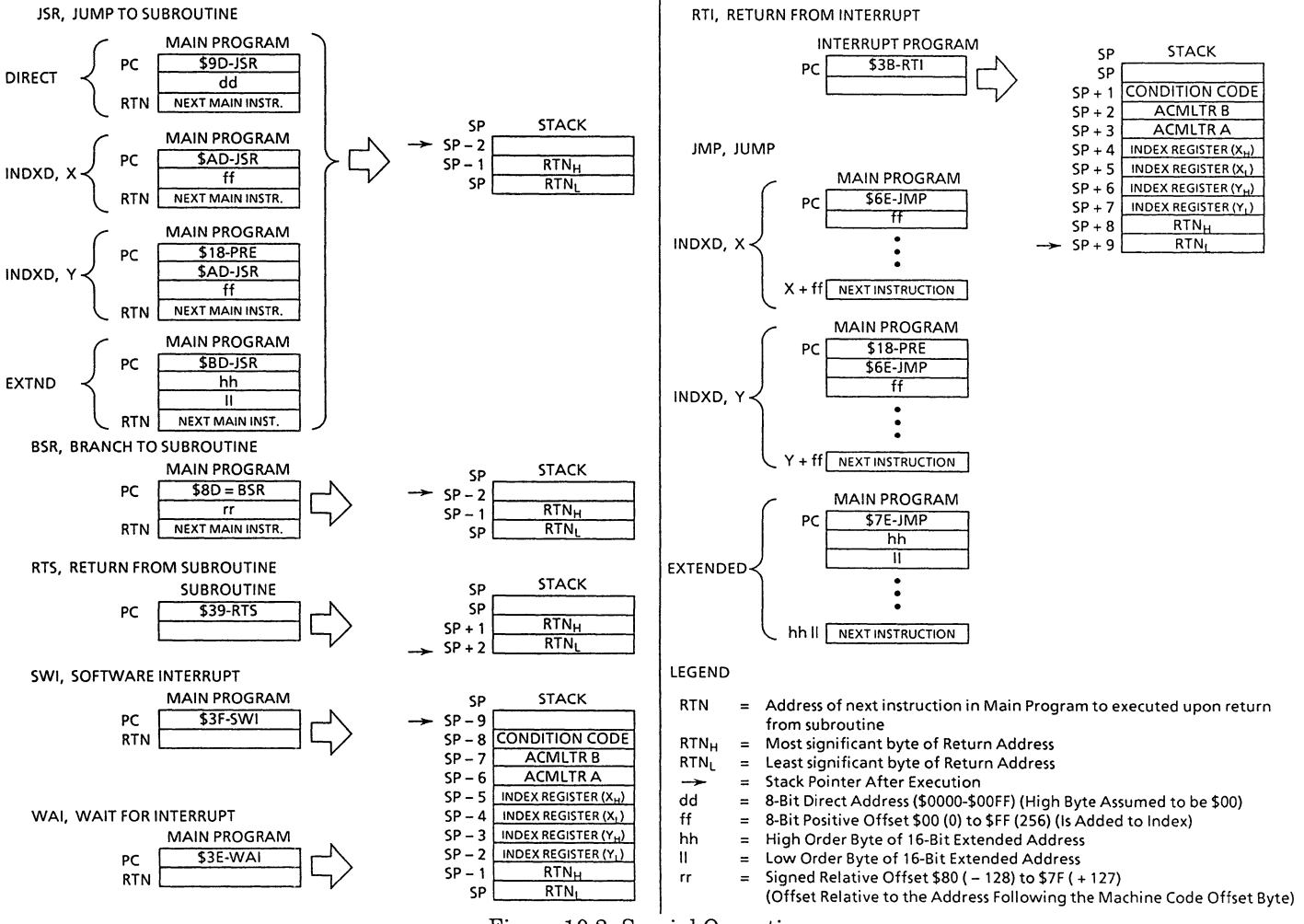
Source Form (s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal)		Bytes	Cycle	Cycle by Cycle*	Condition Codes
				Opcode	Operand (s)				S X H I N Z V C
TXS	Transfer X to Stack Pointer	IX – 1→SP	INH	35		1	3	2-2	- - - - - - - -
TYS	Transfer Y to Stack Pointer	IY – 1→SP	INH	18 35		2	4	2-4	- - - - - - - -
WAI	Wait for Interrupt	Stack Regs & WAIT	INH	3E		1	***	2-16	- - - - - - - -
XGDX	Exchange D with X	IX→D, D→IX	INH	8F		1	3	2-2	- - - - - - - -
XGDY	Exchange D with Y	IY→D, D→IY	INH	18 8F		2	4	2-4	- - - - - - - -

* Cycle-by-cycle number provides a reference to Table 10.2 through 10.8 which detail cycle-by-cycle operation.
Example:Table 10.1 cycle-by-cycle column reference number 2.4 equals Table 10.2 line item 2.4.

** Infinity or Until Reset Occurs

*** 12 Cycles are used beginning with the opcode fetch. A wait state is entered which remains in effect for an integer number of MPUE-clock cycles (n) until an interrupt is recognized. Finally, two additional cycles are used to fetch the appropriate interrupt vector (14 + n total).

dd = 8-Bit Direct Address ($0000-$00FF) (High Byte Assumed to be $00)
ff = 8-Bit Positive Offset $00 (0) to $FF (255) (Is Added to Index)
hh = High Order Byte of 16-Bit Extended Address
ii = One Byte of Immediate Data
jj = High Order Byte of 16-Bit Immediate Data
kk = Low Order Byte of 16-Bit Immediate Data
ll = Low Order Byte of 16-Bit Extended Address
mm = 8-Bit Mask (Set Bits to be Affected)
rr = Signed Relative Offset $80 (– 128) to $7F (+127)
(Offset Relative to the Address Following the Machine Code Offset Byte)



LEGEND

RTN	=	Address of next instruction in Main Program to executed upon return from subroutine
RTN_H	=	Most significant byte of Return Address
RTN_L	=	Least significant byte of Return Address
→	=	Stack Pointer After Execution
dd	=	8-Bit Direct Address ($0000-$00FF) (High Byte Assumed to be $00)
ff	=	8-Bit Positive Offset $00 (0) to $FF (256) (Is Added to Index)
hh	=	High Order Byte of 16-Bit Extended Address
ll	=	Low Order Byte of 16-Bit Extended Address
rr	=	Signed Relative Offset $80 (– 128) to $7F (+ 127) (Offset Relative to the Address Following the Machine Code Offset Byte)

Figure 10.2 Special Operations

Table 10.2 Cycle-by-Cycle Operation — Inherent Mode (Sheet 1 of 4)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
2-1	ABA, ASLA, ASLB, ASRA, ASRB, CBA, CLC, CLI, CLRA, CLRB, CLV, COMA, COMB, DAA, DECA, DECB, INCA, INCB, LSLA, LSLB, LSRA, LSRB, NEGA, NEGB, NOP, ROLA, ROLB, RORA, RORB, SBA, SEC, SEI, SEV, STOP, TAB, TAP, TBA, TPA, TSTA, TSTB	2	1 2	Opcode Address Opcode Address + 1	1 1	Opcode Irrelevant Data
2-2	ABX, ASLD, DEX, INX, LSLD, LSRD, TXS, XGDX	3	1 2 3	Opcode Address Opcode Address + 1 $FFFF	1 1 1	Opcode Irrelevant Data Irrelevant Data
2-3	DES, INS, TSX	3	1 2 3	Opcode Address Opcode Address + 1 Previous SP Value	1 1 1	Opcode Irrelevant Data Irrelevant Data
2-4	ABY, DEY, INY, TYS, XGDY	4	1 2 3 4	Opcode Address Opcode Address + 1 Opcode Address + 2 $FFFF	1 1 1 1	Opcode (Page Select Byte) ($18) Opcode (Second Byte) Irrelevant Data Irrelevant Data
2-5	TSY	4	1 2 3 4	Opcode Address Opcode Address + 1 Opcode Address + 2 Stack Pointer	1 1 1 1	Opcode (Page Select Byte) ($18) Opcode (Second Byte) ($30) Irrelevant Data Irrelevant Data
2-6	PSHA, PSHB	3	1 2 3	Opcode Address Opcode Address + 1 Stack Pointer	1 1 0	Opcode Irrelevant Data Accumulator Data
2-7	PSHX	4	1 2 3 4	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer – 1	1 1 0 0	Opcode ($3C) Irrelevant Data IXL (Low Byte) to Stack IXH (High Byte) to Stack
2-8	PSHY	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Opcode Address + 2 Stack Pointer Stack Pointer – 1	1 1 1 0 0	Opcode (Page Select Byte) ($18) Opcode (Second Byte) ($3C) Irrelevant Data IXL (Low Byte) to Stack IXH (High Byte) to Stack

*The reference number is given to provide a cross-reference to Table 10.1.

Table 10.2 Cycle-by-Cycle Operation — Inherent Mode (Sheet 2 of 4)

Reference Number*	Address Mode and Instructions	Cycle	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
2-9	PULA, PULB	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	1	Irrelevant Data
			4	Stack Pointer + 1	1	Operand Data from Stack
2-10	PULX	5	1	Opcode Address	1	Opcode ($38)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	1	Irrelevant Data
			4	Stack Pointer + 1	1	IXH (High Byte) from Stack
			5	Stack Pointer + 2	1	IXL (Low Byte) from Stack
2-11	PULY	6	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($38)
			3	Opcode Address + 2	1	Irrelevant Data
			4	Stack Pointer	1	Irrelevant Data
			5	Stack Pointer + 1	1	IYH (High Byte) from Stack
			6	Stack Pointer + 2	1	IYL (Low Byte) from Stack
2-12	RTS	5	1	Opcode Address	1	Opcode ($39)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	1	Irrelevant Data
			4	Stack Pointer + 1	1	Address of Next Instruction (High Byte)
			5	Stack Pointer + 2	1	Address of Next Instruction (Low Byte)
2-13	MUL	10	1	Opcode Address	1	Opcode ($3D)
			2	Opcode Address + 1	1	Irrelevant Data
			3	$FFFF	1	Irrelevant Data
			4	$FFFF	1	Irrelevant Data
			5	$FFFF	1	Irrelevant Data
			6	$FFFF	1	Irrelevant Data
			7	$FFFF	1	Irrelevant Data
			8	$FFFF	1	Irrelevant Data
			9	$FFFF	1	Irrelevant Data
			10	$FFFF	1	Irrelevant Data
2-14	RTI	12	1	Opcode Adderess	1	Opcode ($3B)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	1	Irrelevant Data
			4	Stack Pointer + 1	1	Condition Code Register from Stack
			5	Stack Pointer + 2	1	B Accumulator from Stack
			6	Stack Pointer + 3	1	A Accumulator from Stack
			7	Stack Pointer + 4	1	IXH (High Byte) from Stack
			8	Stack Pointer + 5	1	IXL (Low Byte) from Stack
			9	Stack Pointer + 6	1	IYH (High Byte) from Stack

*The reference number is given to provide a cross-reference to Table 10.1.

Table 10.2 Cycle-by-Cycle Operation — Inherent Mode (Sheet 3 of 4)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
2-14 (Continued)	RTI	12	10	Stack Pointer + 7	1	IYL (Low Byte) from Stack
			11	Stack Pointer + 8	1	Address of Next Instruction (High Byte)
			12	Stack Pointer + 9	1	Address of Next Instruction (Low Byte)
2-15	SWI	14	1	Opcode Adderess	1	Opcode ($3F)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	0	Return Address (Low Byte)
			4	Stack Pointer – 1	0	Return Address (High Byte)
			5	Stack Pointer – 2	0	IYL (Low Byte) to Stack
			6	Stack Pointer – 3	0	IYH (High Byte) to Stack
			7	Stack Pointer – 4	0	IXL (Low Byte) to Stack
			8	Stack Pointer – 5	0	IXH (High Byte) to Stack
			9	Stack Pointer – 6	0	A Accumulator to Stack
			10	Stack Pointer – 7	0	B Accumulator to Stack
			11	Stack Pointer – 8	0	Condition Code Register to Stack
			12	Stack Pointer – 8	1	Irrelevant Data
			13	Address of SWI Vector (First Location)	1	SWI Service Routine Address (High Byte)
			14	Address of Vector + 1 (Second Location)	1	SWI Service Routine Address (Low Byte)
2-16	WAI	14 + n	1	Opcode Adderess	1	Opcode ($3E)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Stack Pointer	0	Return Address (Low Byte)
			4	Stack Pointer – 1	0	Return Address (High Byte)
			5	Stack Pointer – 2	0	IYL (Low Byte) to Stack
			6	Stack Pointer – 3	0	IYH (High Byte) to Stack
			7	Stack Pointer – 4	0	IXL (Low Byte) to Stack
			8	Stack Pointer – 5	0	IXH (High Byte) to Stack
			9	Stack Pointer – 6	0	A Accumulator to Stack
			10	Stack Pointer – 7	0	B Accumulator to Stack
			11	Stack Pointer – 8	0	Condition Code Register to Stack
			12 to n + 12	Stack Pointer – 8	1	Irrelevant Data
			n + 13	Address of Vector (First Location)	1	Service Routine Address (High Byte)
			n + 14	Address of Vector (Second Location)	1	Service Routine Address (Low Byte)
2-17	FDIV, IDIV	41	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Irrelevant Data
			3 – 41	$FFFF	1	Irrelevant Data

*The reference number is given to provide a cross-reference to Table 10.1.

Table 10.2 Cycle-by-Cycle Operation — Inherent Mode (Sheet 4 of 4)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
2-18	Page 1 Illegal Opcodes	15	1	Opcode Address	1	Opcode (Illegal)
			2	Opcode Address + 1	1	Irrelevant Data
			3	$FFFF	1	Irrelevant Data
			4	Stack Pointer	0	Return Address (Low Byte)
			5	Stack Pointer – 1	0	Return Address (High Byte)
			6	Stack Pointer – 2	0	IYL (Low Byte) to Stack
			7	Stack Pointer – 3	0	IYH (High Byte) to Stack
			8	Stack Pointer – 4	0	IXL (Low Byte) to Stack
			9	Stack Pointer – 5	0	IXH (High Byte) to Stack
			10	Stack Pointer – 6	0	A Accumulator
			11	Stack Pointer – 7	0	B Accumulator
			12	Stack Pointer – 8	0	Condition Code Register to Stack
			13	Stack Pointer – 8	1	Irrelevant Data
			14	Address of Vector (First Location)	1	Service Routine Address (High Byte)
			15	Address of Vector + 1 (Second Location)	1	Service Routine Address (Low Byte)
2-19	Pages 2, 3, or 4 Illegal Opcodes	16	1	Opcode Address	1	Opcode (Legal Page Select)
			2	Opcode Address + 1	1	Opcode (Illegal Second Byte)
			3	Opcode Address + 2	1	Irrelevant Data
			4	$FFFF	1	Irrelevant Data
			5	Stack Pointer	0	Return Address (Low Byte)
			6	Stack Pointer – 1	0	Return Address (High Byte)
			7	Stack Pointer – 2	0	IYL (Low Byte) to Stack
			8	Stack Pointer – 3	0	IYH (High Byte) to Stack
			9	Stack Pointer – 4	0	IXL (Low Byte) to Stack
			10	Stack Pointer – 5	0	IXH (High Byte) to Stack
			11	Stack Pointer – 6	0	A Accumulator
			12	Stack Pointer – 7	0	B Accumulator
			13	Stack Pointer – 8	0	Condition Code Register to Stack
			14	Stack Pointer – 8	1	Irrelevant Data
			15	Address of Vector (First Location)	1	Service Routine Address (High Byte)
			16	Address of Vector + 1 (Second Location)	1	Service Routine Address (Low Byte)
2-20	TEST	Infinite	1	Opcode Address	1	Opcode ($00)
			2	Opcode Address + 1	1	Irrelevant Data
			3	Opcode Address + 1	1	Irrelevant Data
			4	Opcode Address + 2	1	Irrelevant Data
			5 – n	Previous Address + 1	1	Irrelevant Data

*The reference number is given to provide a cross-reference to Table 10.1.

Table 10.3 Cycle-by-Cycle Operation — Immediate Mode

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
3-1	ADCA, ADCB, ADDA, ADDB, ANDA, ANDB, BITA, BITB, CMPA, CMPB, EORA, EORB, LDAA, LDAB, ORAA, ORAB, SBCA, SBCB, SUBA, SUBB	2	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Data
3-2	LDD, LDS, LDX	3	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Data (High Byte)
			3	Opcode Address + 2	1	Operand Data (Low Byte)
3-3	ADDD, CPX, SUBD	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Data (High Byte)
			3	Opcode Address + 2	1	Operand Data (Low Byte)
			4	$FFFF	1	Irrelevant Data
3-4	LDY	4	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($EC)
			3	Opcode Address + 2	1	Operand Data (High Byte)
			4	Opcode Address + 3	1	Operand Data (Low Byte)
3-5	CPD, CPY	5	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Operand Data (High Byte)
			4	Opcode Address + 3	1	Operand Data (Low Byte)
			5	$FFFF	1	Irrelevant Data

*The reference number is given to provide a cross-reference to Table 10.1.

Table 10.4 Cycle-by-Cycle Operation — Direct Mode (Sheet 1 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
4-1	ADCA, ADCB, ADDA, ADDB, ANDA, ANDB, BITA, BITB, CMPA, CMPB, EORA, EORB, LDAA, LDAB, ORAA, ORAB, SBCA, SBCB, SUBA, SUBB	3	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	1	Operand Data
4-2	STAA, STAB	3	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	0	Data from Accumulator
4-3	LDD, LDS, LDX	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	1	Operand Data (High Byte)
			4	Operand Address + 1	1	Operand Data (Low Byte)
4-4	STD, STS, STX	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	0	Register Data (High Byte)
			4	Operand Address + 1	0	Register Data (Low Byte)
4-5	LDY	5	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($DE)
			3	Opcode Address + 2	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			4	Operand Address	1	Operand Data (High Byte)
			5	Operand Address + 1	1	Operand Data (Low Byte)
4-6	STY	5	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($DE)
			3	Opcode Address + 2	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			4	Operand Address	0	Register Data (High Byte)
			5	Operand Address + 1	0	Register Data (Low Byte)
4-7	ADDD, CPX, SUBD	5	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low byte) (High Byte Assumed to be $00)
			3	Operand Address	1	Operand Data (High Byte)
			4	Operand Address + 1	1	Operand Data (Low Byte)
			5	$FFFF	1	Irrelevant Data

*The reference number is given to provide a cross-reference to Table 10.1.

Table 10.4 Cycle-by-Cycle Operation — Direct Mode (Sheet 2 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
4-8	JSR	5	1	Opcode Address	1	Opcode ($9D)
			2	Opcode Address + 1	1	Subroutine Address (Low Byte) (High Byte Assumed to be $00)
			3	Subroutine Address	1	First Subroutine Opcode
			4	Stack Pointer	0	Return Address (Low Byte)
			5	Stack Pointer – 1	0	Return Address (High Byte)
4-9	CPD, CPY	6	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			4	Operand Address	1	Operand Data (High Byte)
			5	Operand Address + 1	1	Operand Data (Low Byte)
			6	$FFFF	1	Irrelevant Data
4-10	BCLR, BSET	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	1	Original Operand Data
			4	Opcode Address + 2	1	Mask Byte
			5	$FFFF	1	Irrelevant Data
			6	Operand Address	0	Result Operand Data
4-11	BRCLR, BRSET	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (Low Byte) (High Byte Assumed to be $00)
			3	Operand Address	1	Original Operand Data
			4	Opcode Address + 2	1	Mask Byte
			5	Opcode Address + 3	1	Branch Offset
			6	$FFFF	1	Irrelevant Data

*The reference number is given to provide a cross-reference to Table 10.1.

Table 10.5 Cycle-by-Cycle Operation — Extended Mode (Sheet 1 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
5-1	JMP	3	1	Opcode Address	1	Opcode ($7E)
			2	Opcode Address + 1	1	Jump Address (High Byte)
			3	Opcode Address + 2	1	Jump Address (Low Byte)
5-2	ADCA, ADCB, ADDA, ADDB, ANDA, ANDB, BITA, BITB, CMPA, CMPB, EORA, EORB, LDAA, LDAB, ORAA, ORAB, SBCA, SBCB, SUBA, SUBB	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	1	Operand Data
5-3	STAA, STAB	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	0	Accumulator Data
5-4	LDD, LDS, LDX	5	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	1	Operand Data (High Byte)
			5	Operand Address + 1	1	Operand Data (Low Byte)
5-5	STD, STS, STX	5	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	0	Register Data (High Byte)
			5	Operand Address + 1	0	Register Data (Low Byte)
5-6	LDY	6	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($FF)
			3	Opcode Address + 2	1	Operand Address (High Byte)
			4	Opcode Address + 3	1	Operand Address (Low Byte)
			5	Operand Address	1	Operand Data (High Byte)
			6	Operand Address + 1	1	Operand Data (Low Byte)
5-7	STY	6	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($FF)
			3	Opcode Address + 2	1	Operand Address (High Byte)
			4	Opcode Address + 3	1	Operand Address (Low Byte)
			5	Operand Address	0	Register Data (High Byte)
			6	Operand Address + 1	0	Register Data (Low Byte)
5-8	ASL, ASR, CLR, COM, DEC, INC, LSL, LSR, NEG, ROL, ROR	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	1	Original Operand Data

*The reference number is given to provide a cross-reference to Table 10.1.

Table 10.5 Cycle-by-Cycle Operation — Extended Mode (Sheet 2 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
5-8 (Continued)	ASL, ASR, CLR, COM, DEC, INC, LSL, LSR, NEG, ROL, ROR	6	5	$FFFF	1	Irrelevant Data
			6	Operand Address	0	Result Operand Data
5-9	TST	6	1	Opcode Address	1	Opcode ($7D)
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	1	Original Operand Data
			5	$FFFF	1	Irrelevant Data
			6	$FFFF	1	Irrelevant Data
5-10	ADDD, CPX, SUBD	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Operand Address (High Byte)
			3	Opcode Address + 2	1	Operand Address (Low Byte)
			4	Operand Address	1	Operand Data (High Byte)
			5	Operand Address + 1	1	Operand Data (Low Byte)
			6	$FFFF	1	Irrelevant Data
5-11	CPD, CPY	7	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Operand Address (High Byte)
			4	Opcode Address + 3	1	Operand Address (Low Byte)
			5	Operand Address	1	Operand Data (High Byte)
			6	Operand Address + 1	1	Operand Data (Low Byte)
			7	$FFFF	1	Irrelevant Data
5-12	JSR	6	1	Opcode Address	1	Opcode ($BD)
			2	Opcode Address + 1	1	Subroutine Address (High Byte)
			3	Opcode Address + 2	1	Subroutine Address (Low Byte)
			4	Subroutine Address	1	First Opcode in Subroutine
			5	Stack Pointer	0	Return Address (Low Byte)
			6	Stack Pointer – 1	0	Return Address (High Byte)

* The reference number is given to provide a cross-reference to Table 10.1.

Table 10.6 Cycle-by-Cycle Operation — Indexed X Mode (Sheet 1 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
6-1	JMP	3	1	Opcode Address	1	Opcode ($6E)
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
6-2	ADCA, ADCB, ADDA, ADDB, ANDA, ANDB, BITA, BITB, CMPA, CMPB, EORA, EORB, LDAA, LDAB, ORAA, ORAB, SBCA, SBCB, SUBA, SUBB	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Operand Data
6-3	ASL, ASR, CLR, COM, DEC, INC, LSL, LSR, NEG, ROL, ROR	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Original Operand Data
			5	$FFFF	1	Irrelevant Data
			6	(IX) + Offset	0	Result Operand Data
6-4	TST	6	1	Opcode Address	1	Opcode ($6D)
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Original Operand Data
			5	$FFFF	1	Irrelevant Data
			6	$FFFF	1	Irrelevant Data
6-5	STAA, STAB	4	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	0	Accumulator Data
6-6	LDD, LDS, LDX	5	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Operand Data (High Byte)
			5	(IX) + Offset + 1	1	Operand Data (Low Byte)
6-7	LDY	6	1	Opcode Address	1	Opcode (Page Select Byte) ($1A)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($EE)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IX) + Offset	1	Operand Data (High Byte)
			6	(IX) + Offset + 1	1	Operand Data (Low Byte)
6-8	STD, STS, STX	5	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data

*The reference number is given to provide a cross-reference to Table 10.1.

Table 10.6 Cycle-by-Cycle Operation — Indexed X Mode (Sheet 2 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
6-8 (Continued)	STD, STS, STX	5	4	(IX) + Offset	0	Register Data (High Byte)
			5	(IX) + Offset + 1	0	Register Data (Low Byte)
6-9	STY	6	1	Opcode Address	1	Opcode (Page Select Byte) ($1A)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($EF)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IX) + Offset	0	Register Data (High Byte)
			6	(IX) + Offset + 1	0	Register Data (Low Byte)
6-10	ADDD, CPX, SUBD	6	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Register Data (High Byte)
			5	(IX) + Offset + 1	1	Register Data (Low Byte)
			6	$FFFF	1	Irrelevant Data
6-11	CPD, CPY	7	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IX) + Offset	1	Register Data (High Byte)
			6	(IX) + Offset + 1	1	Register Data (Low Byte)
			7	$FFFF	1	Irrelevant Data
6-12	JSR	6	1	Opcode Address	1	Opcode ($AD)
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	First Opcode in Subroutine
			5	Stack Pointer	0	Return Address (Low Byte)
			6	Stack Pointer – 1	0	Return Address (High Byte)
6-13	BCLR, BSET	7	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Original Operand Data
			5	Opcode Address + 2	1	Mask Byte
			6	$FFFF	1	Irrelevant Data
			7	(IX) + Offset	0	Result Operand Data
6-14	BRCLR, BRSET	7	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Index Offset
			3	$FFFF	1	Irrelevant Data
			4	(IX) + Offset	1	Original Operand Data
			5	Opcode Address + 2	1	Mask Byte
			6	Opcode Address + 3	1	Branch Offset
			7	$FFFF	1	Irrelevant Data

*The reference number is given to provide a cross-reference to Table 10.1.

Table 10.7 Cycle-by-Cycle Operation — Indexed Y Mode (Sheet 1 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
7-1	JMP	4	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($6E)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
7-2	ADCA, ADCB, ADDA, ADDB, ANDA, ANDB, BITA, BITB, CMPA, CMPB, EORA, EORB, LDAA, LDAB, ORAA, ORAB, SBCA, SBCB, SUBA, SUBB	5	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Operand Data
7-3	ASL, ASR, CLR, COM, DEC, INC, LSL, LSR, NEG, ROL, ROR	7	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Original Operand Data
			6	$FFFF	1	Irrelevant Data
			7	(IY) + Offset	0	Result Operand Data
7-4	TST	7	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($6D)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Original Operand Data
			6	$FFFF	1	Irrelevant Data
			7	$FFFF	1	Irrelevant Data
7-5	STAA, STAB	5	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	0	Accumulator Data
7-6	LDD, LDS, LDX, LDY	6	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Operand Data (High Byte)
			6	(IY) + Offset + 1	1	Operand Data (Low Byte)
7-7	STD, STS, STX, STY	6	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data

*The reference number is given to provide a cross-reference to Table 10.1.

Table 10.7 Cycle-by-Cycle Operation — Indexed Y Mode (Sheet 2 of 2)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
7-7 (Continued)	STD, STS, STX, STY	6	5	(IY) + Offset	0	Register Data (High Byte)
			6	(IY) + Offset + 1	0	Register Data (Low Byte)
7-8	ADDD, CPD, CPX, CPY, SUBD	7	1	Opcode Address	1	Opcode (Page Select Byte)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Operand Data (High Byte)
			6	(IY) + Offset + 1	1	Operand Data (Low Byte)
			7	$FFFF	1	Irrelevant Data
7-9	JSR	7	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($AD)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	First Opcode in Subroutine
			6	Stack Pointer	0	Return Address (Low Byte)
			7	Stack Pointer – 1	0	Return Address (High Byte)
7-10	BCLR, BSET	8	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Original Operand Data
			6	Opcode Address + 3	1	Mask Byte
			7	$FFFF	1	Irrevant Data
			8	(IY) + Offset	0	Result Operand Data
7-11	BRCLR, BRSET	8	1	Opcode Address	1	Opcode (Page Select Byte) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte)
			3	Opcode Address + 2	1	Index Offset
			4	$FFFF	1	Irrelevant Data
			5	(IY) + Offset	1	Original Operand Data
			6	Opcode Address + 3	1	Mask Byte
			7	Opcode Address + 4	1	Branch Offset
			8	$FFFF	1	Irrelevant Data

*The reference number is given to provide a cross-reference to Table 10.1.

Table 10.8 Cycle-by-Cycle Operation – Relative Mode

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
8-1	BCC, BCS, BEQ, BGE, BGT, BHI, BHS, BLE, BLO, BLS, BLT, BMI, BNE, BPL, BRA, BRN, BVC, BVS	3	1	Opcode Address	1	Opcode
			2	Opcode Address + 1	1	Branch Offset
			3	$FFFF	1	Irrelevant Data
8-2	BSR	6	1	Opcode Address	1	Opcode ($8D)
			2	Opcode Address + 1	1	Branch Offset
			3	$FFFF	1	Irrelevant Data
			4	Subroutine Address	1	Opcode of Next Instruction
			5	Stack Pointer	0	Return Address (Low Byte)
			6	Stack Pointer – 1	0	Return Address (High Byte)

*The reference number is given to provide a cross-reference to Table 10.1.

10.4 OPCODE MAP SUMMARY

Table 10-2 is an opcode map for the instructions and addressing modes used by the TMP68C711J6.

Table 10.9. Opcode Map (Sheet 1 of 2)

DIR

OPCODE MAP PAGE 1

										ACCA				ACCB				
		INH	INH	REL	INH	ACCA	ACCB	IND. X	EXT	IMM	DIR	IND. X	EXT	IMM	DIR	IND. X	EXT	
MSB		0000	0001	0010	0011	0100	0101	0110	01110	1000	1001	1010	1011	1100	1101	1110	1111	
LSB		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0000	0	TEST	SBA	BRA	TSX	NEG				SUB								0
0001	1	NOP	CBA	BRN	INS					CMP								1
0010	2	IDIV	BRSET	BHI	PULA					SBC								2
0011	3	FDIV	BRCLR	BLS	PULB	COM				SUBD				ADDD				3
0100	4	LSRD	BSET	BCC	DES	LSR				AND								4
0101	5	ASLD	BCLR	BCS	TXS					BIT								5
0110	6	TAP	TAB	BNE	PSHA	ROR				LAD								6
0111	7	TPA	TBA	BEQ	PSHB	ASR					STA				STA			7
1000	8	INX	PG2	BVC	PULX	ASL				EOR								8
1001	9	DEX	DAA	BVS	RTS	ROL				ADC								9
1010	A	CLV	PG3	BPL	ABX	DEC				ORA								A
1011	B	SEV	ABA	BMI	PTI					ADD								B
1100	C	CLC	BSET	BGE	PSHX	INC				CPX				LDD				C
1101	D	SEC	BCLR	BLT	MUL	TST				BSR	JSR			PG4	STD			D
1110	E	CLI	BRSET	BGT	WAI	JMP				LDS				LDX				E
1111	F	SEI	BRCLR	BLE	SWI	CLR				XGDX	STS			STOP	STX			F
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	

IND. X

OPCODE MAP PAGE 2 (18xx)

										ACCA				ACCB				
		INH			INH			IND. Y		IMM	DIR	IND. Y	EXT	IMM	DIR	IND. Y	EXT	
MSB		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111	
LSB		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0000	0				TYS			NEG				SUB				SUB		0
0001	1											CMP				CMP		1
0010	2											SBC				SBC		2
0011	3							COM				SUBD				ADDD		3
0100	4							LSR				AND				AND		4
0101	5				TYS							BIT				BIT		5
0110	6							ROR				LDA				LDA		6
0111	7							ASR				STA				STA		7
1000	8	INY			PULY			ASL				EOR				EOR		8
1001	9	DEY						ROL				ADC				ADC		9
1010	A				ABY			DEC				ORA				ORA		A
1011	B											ADD				ADD		B
1100	C		BEST		PSHY			INC			CPY					LDD		C
1101	D		BCLR					TST				JSR				STD		D
1110	E		BRSET					JMP				LDS			LDY			E
1111	F		BRCLR					CLR			XGDY	STS			STY			F
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	

IND. Y

Table 10.9. Opcode Map (Sheet 2 of 2)

OPCODE MAP PAGE 3 (1Axx)

									ACCA				ACCB			
									IMM	DIR	IND. X	EXT			IND. X	
	MSB	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
LSB		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0																0
0001	1																1
0010	2																2
0011	3									CPD							3
0100	4																4
0101	5																5
0110	6																6
0111	7																7
1000	8																8
1001	9																9
1010	A																A
1011	B																B
1100	C											CPY					C
1101	D																D
1110	E															LDY	E
1111	F															STY	F
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

OPCODE MAP PAGE 4 (CDxx)

									ACCA				ACCB			
											IND. Y				IND. Y	
	MSB	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
LSB		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0																0
0001	1																1
0010	2																2
0011	3											CPD					3
0100	4																4
0101	5																5
0110	6																6
0111	7																7
1000	8																8
1001	9																9
1010	A																A
1011	B																B
1100	C											CPX					C
1101	D																D
1110	E															LDX	E
1111	F															STX	F
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

11. ELECTRICAL SPECIFICATIONS

This section contains the electrical specifications and associated timing information for the TMP68C711J6 MCU.

11.1 MAXIMUM RATINGS

Raing	Symbol	Value	Unit
Supply Voltage	V_{DD}	− 0.3 to + 7.0	V
Input Voltage	Vin	− 0.3 to + 7.0	V
Operating Temperature Range TMP68C711J6	T_A	T_L to T_H − 40 to 85	℃
Storage Temperature Range	T_{stg}	− 55 to 150	℃
Current Drain per Pin* Excluding VDD,VSS	I_D	25	mA

* One pin at a time observing maximum power dissipation limits.

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. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (e.g., either GND or V_{DD}).

11.2 THERMAL CHARACTERISTICS

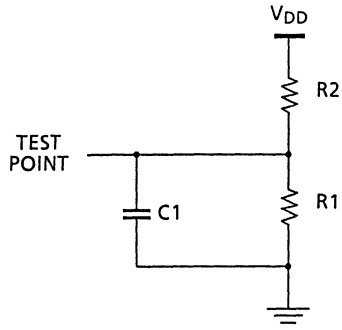
Characteristic	Symbol	Value	Unit
Thermal Resistance	θ_{JA}		℃/W
Plastic 68-Pin Quad Pack (PLCC)		TBD	
Plastic 64-Pin SDIP		TBD	

11.3 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	Max	Unit
Output Voltage (see Note 1) ILoad = ±10.0μA	All Outputs All Outputs Except RESET and MODA	V_{OL} V_{OH}	– V_{DD} – 0.1	0.1 –	V
Output High Voltage I_{Load} = -0.8mA, V_{DD} = 4.5V (see Note 1)	All Outputs Except $\overline{RESET}$, XTAL, and MODA	V_{OH}	V_{DD} – 0.8	–	V
Output Low Voltage I_{Load} = 1.6mA	All Outputs Except XTAL	V_{OL}	–	0.4	V
Input High Voltage	All Inputs Except $\overline{RESET}$ $\overline{RESET}$	V_{IH}	0.7×V_{DD} 0.8×V_{DD}	V_{DD} V_{DD}	V
Input Low Voltage	All Inputs	V_{IL}	V_{SS}	0.2×V_{DD}	V
I/O Ports, Three-State Leakage V_{in} = V_{IH} or V_{IL}	PA0-PA7, PB0-PB7,PC0-PC7, PD0-PD5,PF0-PF7,PG0-PG7 MODA $\overline{LIR}$, $\overline{RESET}$	I_{OZ}	–	±10	μA
Input Current (see Note 2) V_{in} = V_{DD} or V_{SS} V_{in} = V_{DD} or V_{SS} V_{in} = V_{DD} or V_{SS}	 PE0-PE7 $\overline{IRQ}$ $\overline{XIRQ}$ MODB/V_{STBY}	I_{in}	 – –	 ±400 ±1 ±10	 nA μA
RAM Standby Voltage	Powerdown	V_{SB}	4.0	V_{DD}	V
RAM Standby Current	Powerdown	I_{SB}	–	20	μA
Total Supply Current (see Note 3) RUN: Single Chip Expanded Multiplexed WAIT: All Peripheral Functions Shut Down Single-Chip Mode Expanded Multiplexed Mode STOP: No Clocks, Single-Chip Mode		I_{DD} W_{IDD} S_{IDD}	 – – – – –	 20 27 8 10 100	 mA mA mA mA μA
Input Capacitance PB0-PB7, PC0-PC7, PD0-PD5, PF0-PF7, PG0-PG7, MODA/$\overline{LIR}$, $\overline{RESET}$	PE0-PE7, $\overline{IRQ}$, $\overline{XIRQ}$, EXTAL, PA0-PA7,	C_{in}	– –	8 12	pF
Power Dissipation	Single Chip Mode Expanded Multiplexed Mode	P_D	– –	110 150	mW

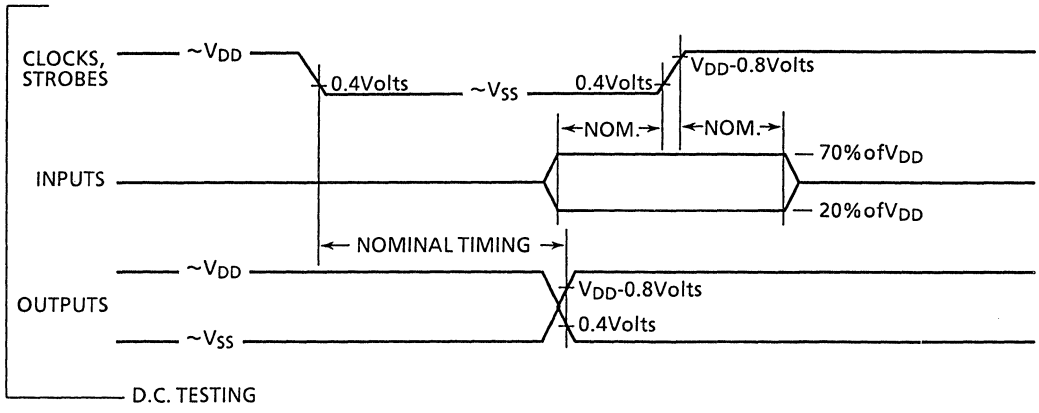
Notes :

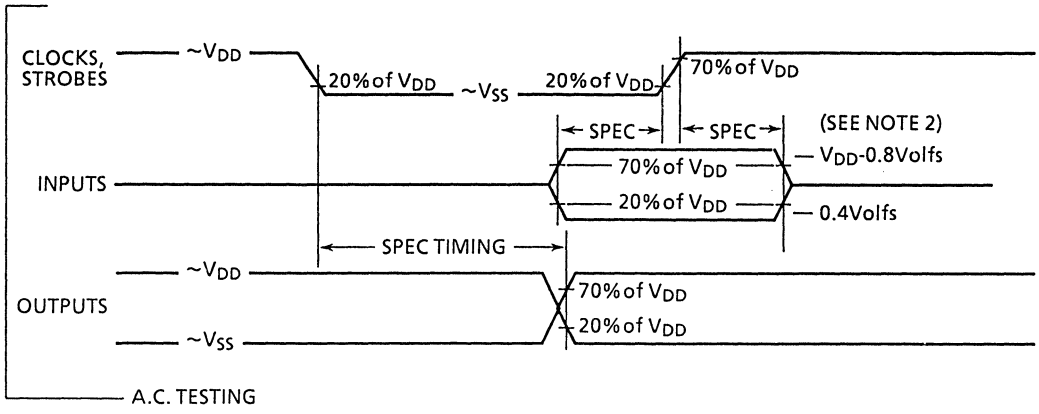
1. V_{OH} specification for $\overline{RESET}$ and MODA is not applicable because they are open-drain pins. V_{OH} specification not applicable to ports C, D and G in wire-OR mode.
2. All ports configured as inputs,
   $V_{IL} \leqq 0.2$ V,
   $V_{IH} \geqq V_{DD} - 0.2$ V,
   No dc loads,
   EXTAL is driven with a square wave, and
   t_{cyc} = 476.5 ns.



Equivalent Test Load[1]

Pins	R1	R2	C1
PA0-PA7 PB0-PB7 PC0-PC7 PD0, PD5 PF0-PF7 PG0-PG7 E, R/W	3.26KΩ	2.38KΩ	90pF
PD1-PD4	3.26KΩ	2.38KΩ	200pF





Notes:
1. Full test loads are applied during all dc electrical and ac timing measurements.
2. During ac timing measurements inputs are driven to 0.4 volts and V_{DD}-0.8 volts while timing measurements are taken at the 20% and 70% of V_{DD} points.

Figure 11.1 Test Methods

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

Characteristic	Symbol	2.1MHz		3.15MHz		Unit
		Min	Max	Min	Max	
Frequency of Operation	fo	dc	2.1	dc	3.15	MHz
E Clock Period	t_{Cyc}	476		317		ns
Crystal Frequency	f_{XTAL}	–	8.4	–	12.6	MHz
External Oscillator Frequency	4 f_o	dc	8.4	dc	12.6	MHz
Processor Control Setup Time(see Figures 11.3, 11.5, and 11.6) $t_{PCS} = 1/4\ t_{cyc} - 50$ ns	t_{PCS}	69	–	29	–	ns
Reset Input Pulse Width (see Note 1 and Figure 11.3) (To Guarantee External Reset Vector)	PW_{RSTL}	8	–	8	–	t_{Cyc}
(Minimum Input Time; May be Preempted by Internal Reset)		2	–	2	–	
Mode Programming Setup Time (see Figure 11.3)	t_{MPS}	2	–	2	–	t_{Cyc}
Mode Programming Hold Time (see Figure 11.3)	t_{MPH}	0	–	0	–	ns
Interrupt Pulse Width, $\overline{IRQ}$ Edge Sensitive Mode (see Figures 11.4 and 11.6) $PW_{IRQ} = t_{cyc} + 20$ ns	PW_{IRQ}	496	–	337	–	ns
Wait Recovery Startup Time (See Figure 11.5)	t_{WRS}	–	4	–	4	t_{Cyc}
Timer Pulse Width Input Capture, Pulse Accumulator Input (see Figure 11.2) $PW_{TIM} = t_{cyc} + 20$ ns	PW_{TIM}	496	–	337	–	ns

Notes :

1. $\overline{RESET}$ will be recognized during the first clock cycle it is held low. Internal circuity then drives the pin low for four clock cycles, releases the pin, and samples the pin level two cycles later to detemine the source of the interrupt. See SECTION 5 RESETS, INTERRUPT, AND LOW POWER MODES for details.
2. All timing is shown with respect to 20% V_{DD} and 70% V_{DD} unless otherwise noted.

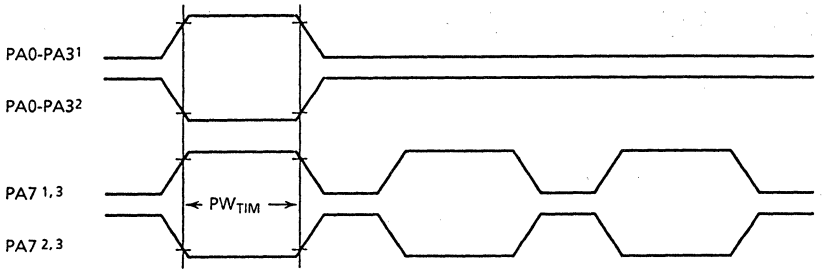
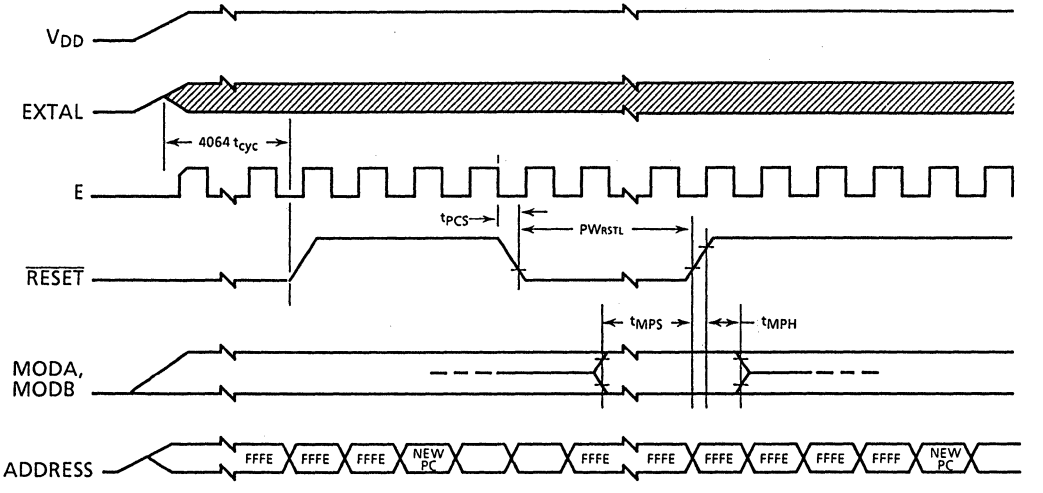


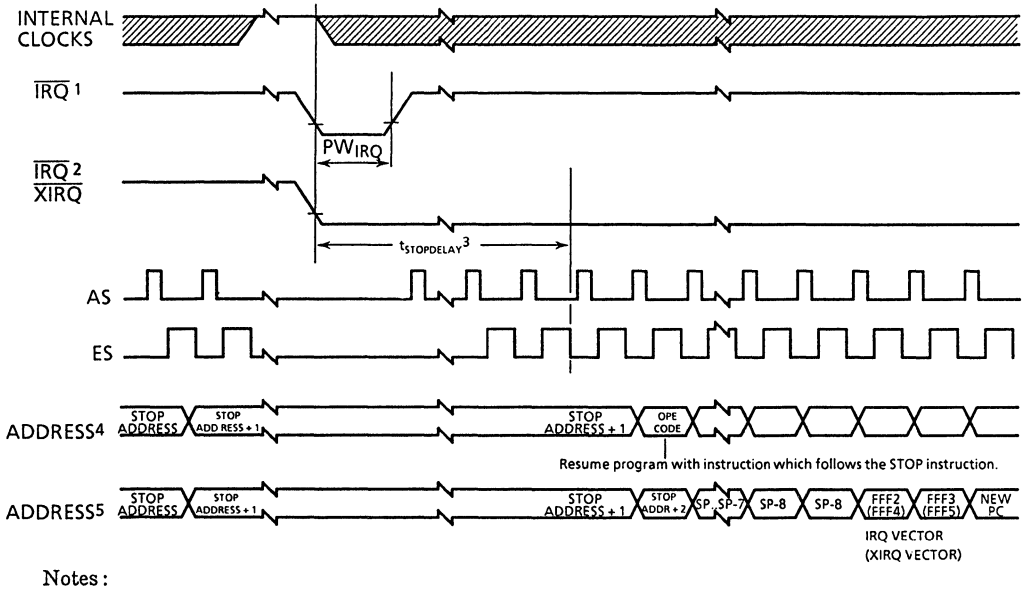
Figure 11.2 Timer Inputs Timing Diagram

Notes :
1. Rising edge sensitive input
2. Falling edge sensitive input
3. Maximum pulse accumulator clocking rate is E frequency divided by 2.



Note : $\overline{\text{RESET}}$ Timing may be changed from 4 cycles high to new PC to 8 cycles high to new PC.

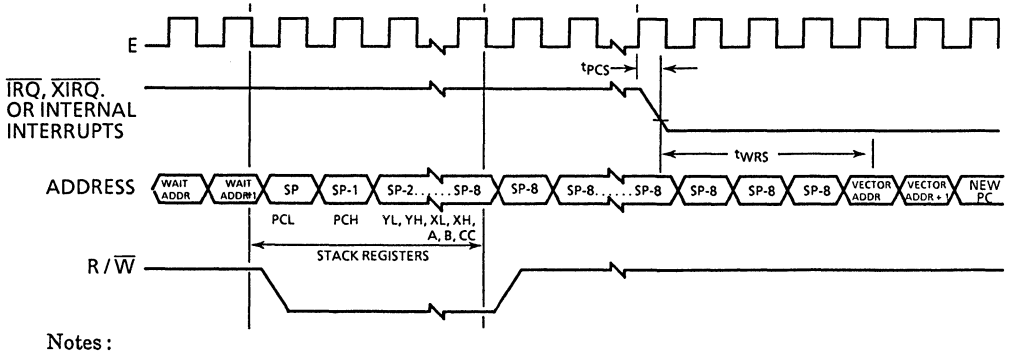
Figure 11.3 POR and External Reset Timing Diagram



Notes :

1. Edge sensitive $\overline{IRQ}$ pin (IRQE bit = 1)
2. Level sensitive $\overline{IRQ}$ pin (IRQE bit = 0)
3. tSTOPDELAY = 4064 tcyc if DLY bit = 1 or 4 tcyc if DLY = 0.
4. $\overline{XIRQ}$ WITH X bit CCR = 1 .
5. $\overline{IRQ}$, or ($\overline{XIRQ}$ with X bit in CCR = 0) .
6. Refer to Table 5.7 for pin states during STOP.

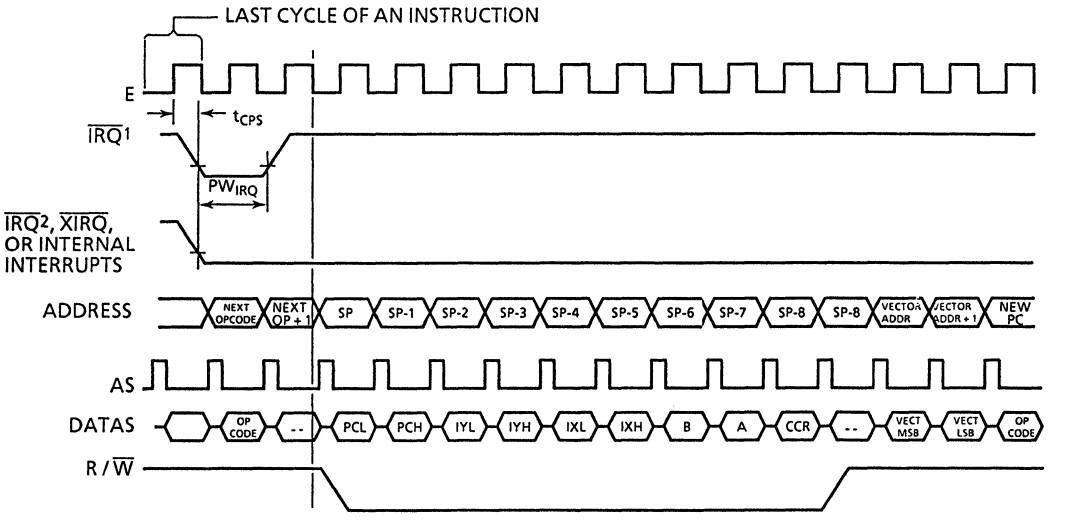
Figure 11.4 STOP Recovery Diagram



Notes :

1 Refer to Table 5.7 for pin states during WAIT.
2 $\overline{RESET}$ will also cause recovery from WAIT.

Figure 11.5 WAIT Recovery from Interrupt Timing Diagram



Notes: 1. Edge sensitive $\overline{IRQ}$ pin (IRQE bit = 1).
2. Level sensitive $\overline{IRQ}$ pin (IRQE bit = 0).

Figure 11.6 Interrupt Timing Diagram

11.5 PERIPHERAL PORT TIMING (V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = T_L to T_H)

Characteristic	Symbol	2.1MHz		3.15MHz		Unit
		Min	Max	Min	Max	
Frequency of Operation (E Clock Frequency)	f_o	2.1	2.1	3.15	3.15	MHz
E-Clock Period	t_{cyc}	476	–	317	–	ns
Peripheral Data Setup Time (MCU Read of Ports A, C, D, E and G)	t_{PDSU}	100	–	100	–	ns
Peripheral Data Hole Time (MCU Read of Ports A, C, D, E, F and G)	t_{PDH}	50	–	50	–	ns
Delay Time, Peripheral Data Write (MCU Writes to Port A) (MCU Writes t o Ports B, C, Dand G) t_{PDW} = 1/4 t_{cyc} + 100 ns	t_{PWD}	– –	200 219	– –	200 169	ns

Notes :

1. Port C, D and G timing is valid for active (DWOM bit not set in SPCR register and CWOM and GWOM bits not set in OPT2 register).
2. All timing is shown with respect to 20% V_{DD} and 70% V_{DD} unless otherwise noted.

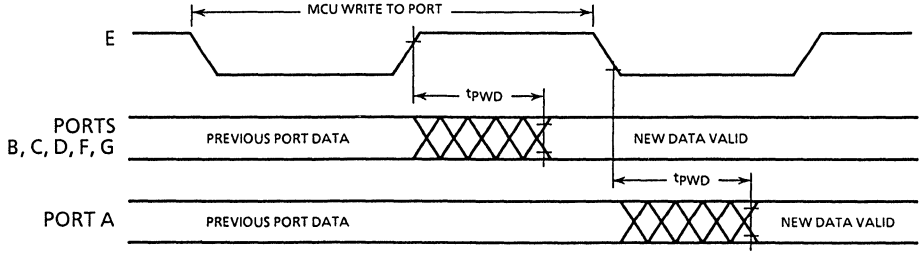


Figure 11.7 Port Write Timing Diagram

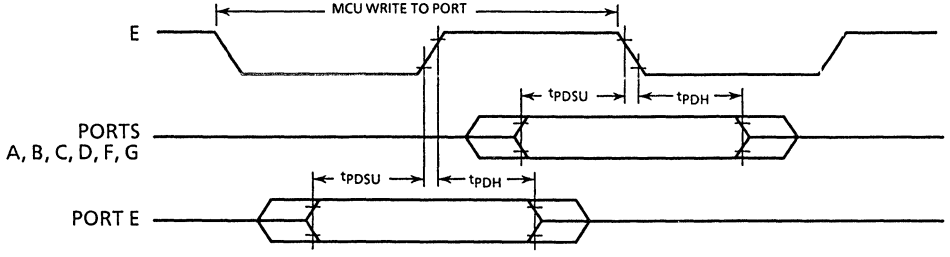


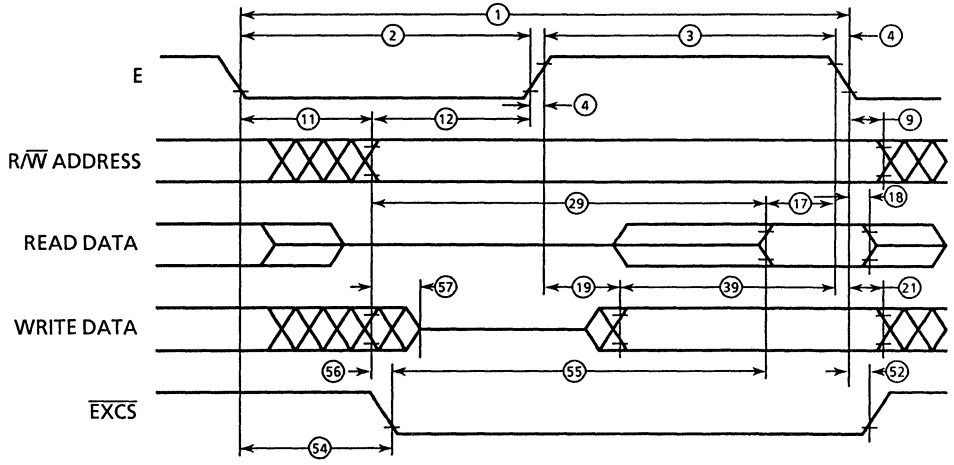
Figure 11.8 Port Read Timing Diagram

11.6 EXPANSION BUS TIMING (V_{DD}=5.0 Vdc±10%, V_{SS}=0 Vdc, T_A=T_L to T_H)

Number	Characteristic	Symbol	2.1MHz		3.15MHz		Unit
			Min	Max	Min	Max	
1	Cycle Time	t_{cyc}	476	–	318	–	ns
2	Pulse Width, E Low ($1/2t_{cyc}$-23ns)	PW_{EL}	215	–	140	–	ns
3	Pulse Width, E High ($1/2tc_{yc}$-28ns)	PW_{EH}	210	–	135	–	ns
4	E Rise and Fall Time	t_r, t_f	–	20	–	15	ns
9	Address Hold Time	t_{AH}	30	–	20	–	ns
11	Address Delay Time ($1/8t_{cyc}$ + 65ns)	t_{AD}	–	125	–	100	ns
12	Address Valid Time (PW_{EL}-t_{AD})	t_{AV}	90	–	40	–	ns
17	Read Data Setup Time	t_{DSR}	30	–	25	–	ns
18	Read Data Hold Time	t_{DHR}	5	–	5	–	ns
19	Write Data Delay Time	t_{DDW}	–	45	–	50	ns
21	Write Data Hold Time	t_{DHW}	30	–	30	–	ns
29	MPU Address Access Time (t_{cyc}-t_{AD}-t_f-t_{DSR})	t_{ACCA}	301	–	178	–	ns
39	Write Data Setup Time (PW_{EH}-t_{DDW})	t_{DSW}	165	–	85	–	ns
52	Chip Select Hold Time	t_{CH}	0	20	0	20	ns
54	Address Valid Chip Select Delay Time ($1/4t_{cyc}$ + 40ns)	t_{ACSD}	130	160	85	120	ns
55	Address Valid Chip Select Access Time (t_{cyc}-t_{ACSD}-t_f-t_{DSR})	t_{ACSA}	266	–	163	–	ns
56	Address Valid To Chip Select Time	t_{AVCS}	10	–	10	–	ns
57	Address Valid To Data Three-State Time	t_{AVDZ}	–	10	–	10	ns

Notes:

1. All timing is shown with respect to 20% V_{DD} and 70% V_{DD}.



Note : Measurement points shown are 20% and 70% V_{DD}

Figure 11.9 Expansion Bus Timing Diagram

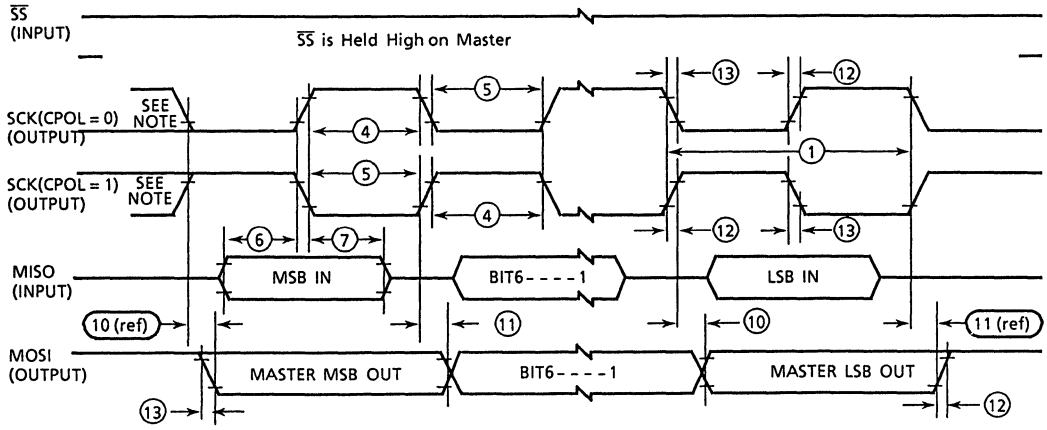
11.7 SERIAL PERIPHERAL INTERFACE (SPI) TIMING (V_{DD}=5.0 Vdc±10%, V_{SS}=0 Vdc, T_A=T_L to T_H, see Figure 11.10)

Num.	Characteristic	Symbol	f=2.1MHz		f=3.15MHz		Unit
			Min	Max	Min	Max	
	Operating Frequency Master Slave	 fOP(m) fOP(s)	 dc dc	 0.5 2.1	 dc dc	 0.5 3.15	 fop MHz
1	Cycle Time Master Slave	 t_{cyc}(m) t_{cyc}(s)	 2.0 480	 – –	 2.0 320	 – –	 t_{cyc} ns
2	Enable Lead Time Master Slave	 t_{lead}(m) t_{lead}(s)	 * 240	 – –	 * 160	 – –	 ns ns
3	Enable Lag Time Master Slave	 t_{lag}(m) t_{lag}(s)	 * 240	 – –	 * 160	 – –	 ns ns
4	Clock (SCK) High Time Master Slave	 t_w(SCKH)m t_w(SCKH)s	 340 190	 – –	 227 127	 – –	 ns ns
5	Clock (SCK) Low Time Master Slave	 t_w(SCKL)m t_w(SCKL)s	 340 190	 – –	 227 127	 – –	 ns ns
6	Data Setup Time (Inputs) Master Slave	 t_{su}(m) t_{su}(s)	 100 100	 – –	 100 100	 – –	 ns ns
7	Data Hold Time (Inputs) Master Slave	 t_h(m) t_h(s)	 100 100	 – –	 100 100	 – –	 ns ns
8	Access Time (Time to Data Active from High-Impedance State) Slave	 t_a	 0	 120	 0	 80	 ns
9	Disable Time (Hold Time to High-Impedance State) Slave	 t_{dis}	 –	 240	 –	 160	 ns
10	Data Valid (After Enable Edge)**	t_v(S)	–	240	–	160	ns
11	Data Hold Time (Outputs) (After Enable Edge)	t_{ho}	0	–	0	–	ns
12	Rise Time (20% V_{DD} to 70% V_{DD}, C_L=200pF) SPI Outputs(SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and $\overline{SS}$)	 t_{rm} t_{rs}	 – –	 100 2.0	 – –	 100 2.0	 ns μs
13	Fall Time (70% V_{DD} to 20% V_{DD}, C_L=200pF) SPI outputs(SCK, MOSI, and MISO) SPI Inputs(SCK, MOSI, MISO, and $\overline{SS}$)	 t_{fm} t_{fs}	 – –	 100 2.0	 – –	 100 2.0	 ns μs

* Signal production depends on software.

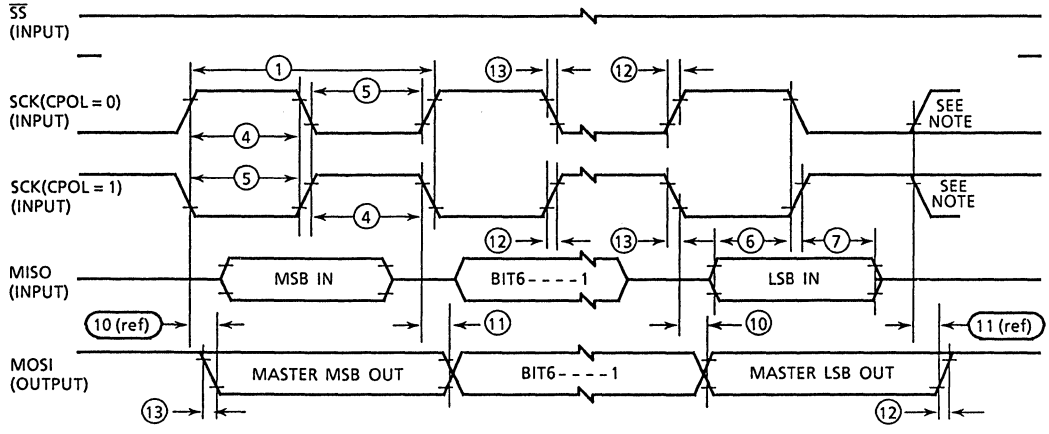
** Assumes 200 pF load on all SPI pins.

Note: All timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted.



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

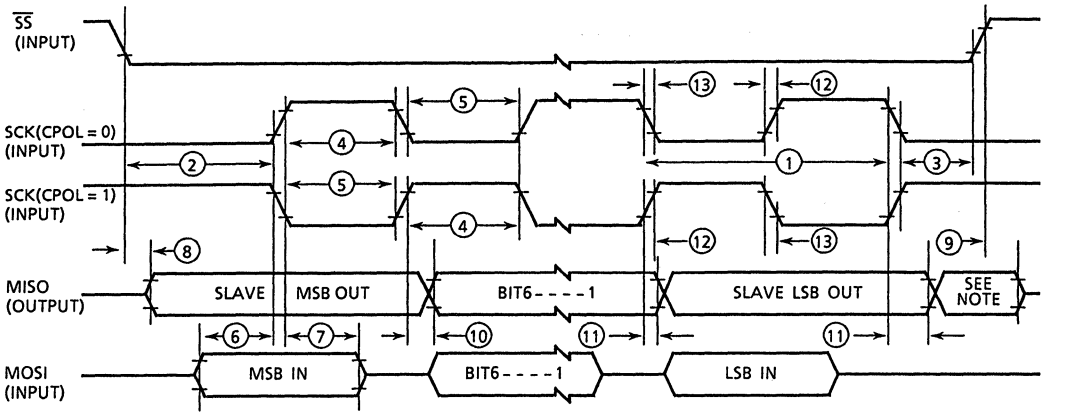
a) SPI MASTER TIMING (CPHA = 0)



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

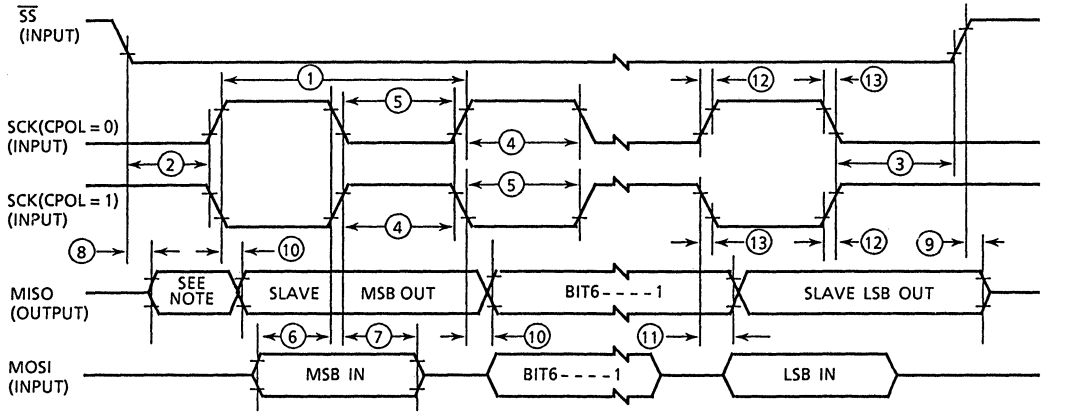
b) SPI MASTER TIMING (CPHA = 1)

Figure 11.10 SPI Timing Diagrams (Sheet 1 of 2)



Note : Not defined but normally MSB of character just received.

c) SPI SLAVE TIMING (CPHA = 0)



Note : Not defined but normally LSB of character previously transmitted.

d) SPI SLAVE TIMING (CPHA = 1)

Figure 11.10 SPI Timing Diagrams (Sheet 2 of 2)

11.8.1 EPROM PROGRAM/Verify OPERATION (PROG MODE)

(V_{DD} = 6.0Vdc ± 0.25Vdc, V_{SS} = 0Vdc, V_{PP} = 12.5Vdc ± 0.5Vdc, T_A = 25 ± 5°C)

Num.	Characteristic	Symbol	Min	Max	Unit
1	Address Setup Time	t_{AV}	2	–	μs
2	Program Pulse Width	t_{PPW}	1	80	ms
3	$\overline{CE}$ Hold Time	t_{CEH}	0	–	μs
4	$\overline{CE}$ Setup Time	t_{CES}	0	–	μs
5	Address Hold Time	t_{AH}	2	–	μs
6	Data Setup Time	t_{DV}	2	–	μs
7	Data Hold Time	t_{DH}	2	–	μs
8	$\overline{OE}$ to Output Data Valid	t_{DS}	–	200	ns
9	$\overline{OE}$ to Output Data High-Z	t_{DHZ}	–	130	ns
10	$\overline{OE}$ Setup Time	t_{OES}	2	–	μs
11	V_{PP} Setup Time	t_{VPS}	2	–	μs

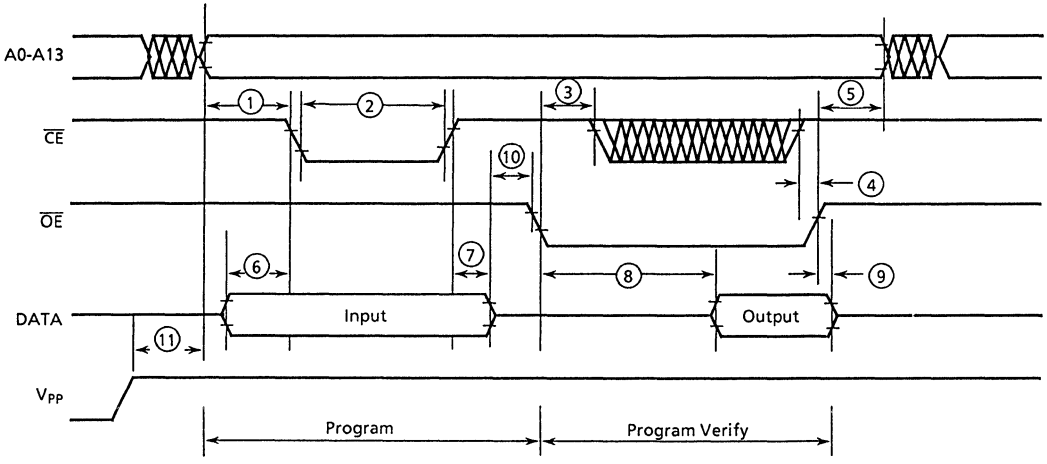


Figure 11.11 EPROM Timing Diagram (Program/Verify)

11.8.2 EPROM READ OPERATION (PROG MODE)

(V_{DD}=5.0Vdc±10%, V_{SS}=0Vdc, V_{PP}=V_{DD}±0.6V, T_A=T_L to T_H)

Num.	Characteristic	Symbol	Min	Max	Unit
20	Address Access Time	t_{RAC}	–	250	ns
21	Output Data Hold Time	t_{RDH}	0	–	ns
22	$\overline{CE}$ to Output Data Valid	t_{RDCV}	–	250	ns
23	$\overline{CE}$ to Output Data High-Z	t_{RDCH}	0	90	ns
24	$\overline{OE}$ to Output Data Valid	t_{RDOV}	–	200	ns
25	$\overline{CE}$ to Output Data High-Z	t_{RDOH}	0	–	ns

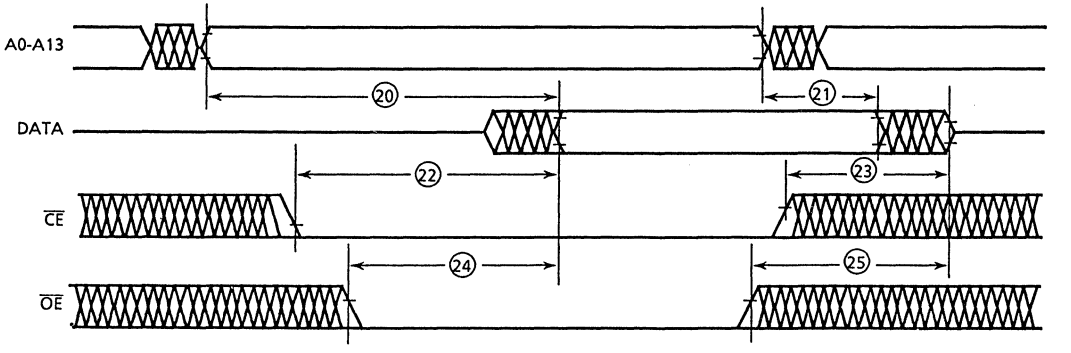


Figure 11.12 EPROM Timing Diagram (Read)

12. MECHANICAL DATA AND ORDERING INFORMATION

The following section contains the pin assignments, package dimensions and ordering information for the TMP68C711J6 MCU.

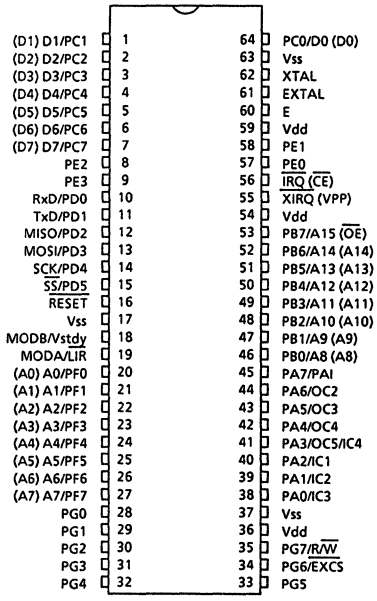
12.1 ORDERING INFORMATION

Package Type	Temperature	MC Order Number Ceramic (EPROM)	Plastic (OTPROM)
68 - Pin PLCC	– 40°C to 85°C		TMP68C711J6T
64 - Pin SDIP	– 40°C to 85°C		TMP68C711J6N
64 - Pin SDIC (Window)	– 40°C to 85°C	TMP68C711J6E	

12.1 PIN ASSIGNMENTS

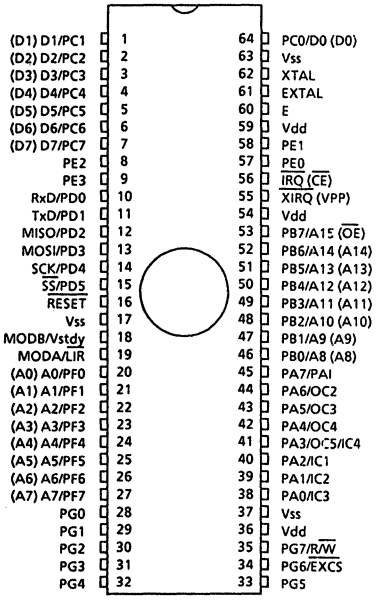
The TMP68C711J6 is available in a 64-pin plastic shrink dual-in-line package (S-DIP) a 68-pin plastic lead chip carrier (PLCC) package. The following paragraph provide pin assignments.

N SUFFIX
64 PIN S-DIP



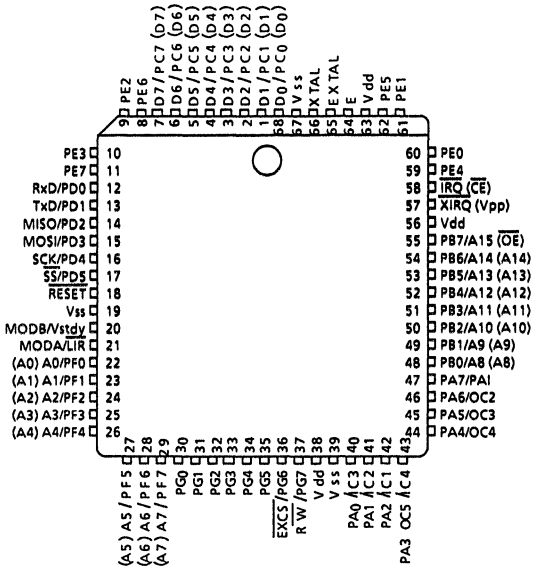
Note: ( ) EPROM Mode

E SUFFIX
64 PIN SDIC



Note: ( ) EPROM Mode

T SUFFIX
68 PIN PLCC

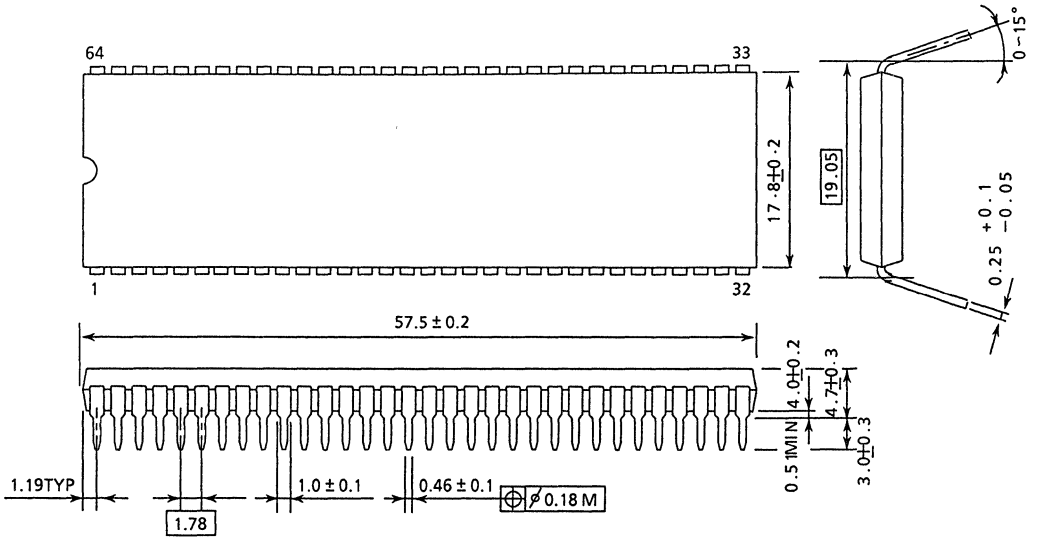


Note: ( ) EPROM Mode

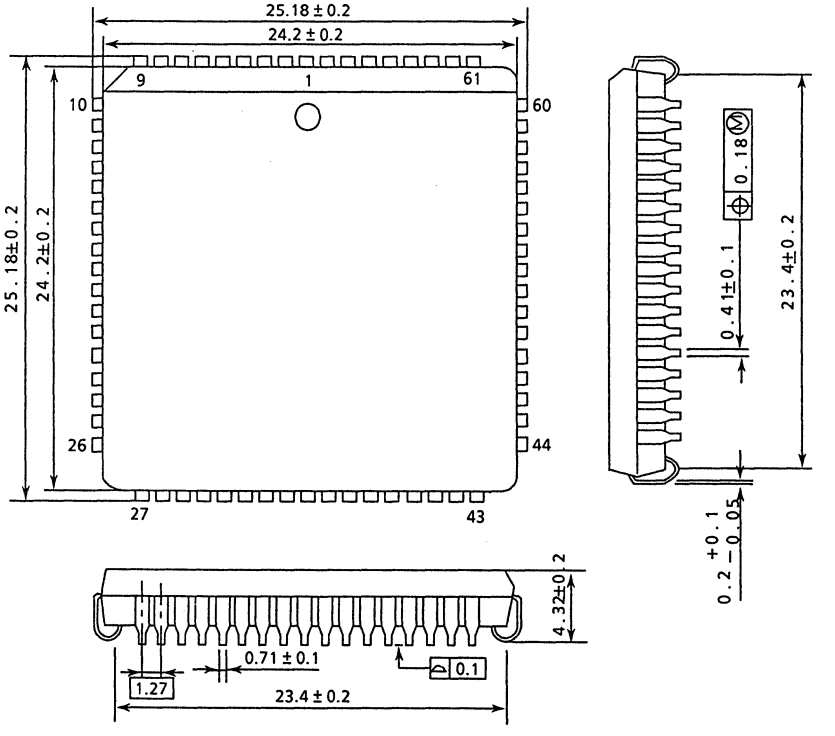
12.3 PACKAGE DIMENSIONS

N SUFFIX 64PIN S-DIP

(UNIT : mm)



T SUFFIX 68PIN PLCC



The products described in this document are strategic products subject to COCOM regulations. They should not be exported without authorization from the appropriate governmental authorities.
The products described in this document contain components made in the United States and subject to export control of the U.S. authorities. Diversion contrary to the U.S. law is prohibited.

TOSHIBA CORPORATION

TOSHIBA AMERICA ELECTRONIC COMPONENTS, INC.

REGIONAL SALES OFFICES

NORTHWEST REGION
1220 Midas Way
Sunnyvale, CA 94086
TEL: (408) 737-9844
FAX: (408) 737-9905

SOUTHWEST REGION
15621 Redhill Ave.
Suite 205
Tustin, CA 92680
TEL: (714) 259-0368
FAX: (714) 259-9439

CENTRAL REGION
One Parkway North
Suite 500
Deerfield, IL 60015-2547
TEL: (708) 945-1500
FAX: (708) 945-1044
TWX: 29-7131

SOUTHCENTRAL REGION
777 E. Campbell Road
Suite 650
Richardson, TX 75081
TEL: (214) 480-0470
FAX: (214) 235-4114

EASTERN REGION
25 Mall Road
5th Floor
Burlington, MA 01803
TEL: (617) 272-4352
FAX: (617) 272-3089
TWX: 710-321-6730

SOUTHEAST REGION
Waterford Centre
5555 Triangle Parkway
Suite 300
Norcross, GA 30092
TEL: (404) 368-0203
FAX: (404) 368-0075

TOSHIBA

CSM-0068/10-90