MC68HC11A8

HCMOS Single-Chip Microcontroller

TABLE OF CONTENTS

Paragraph Number	Title	Page Number

1 INTRODUCTION

2 SIGNAL DESCRIPTIONS AND OPERATING MODES

3 ON-CHIP MEMORY

4 PARALLEL I/O

5 SERIAL COMMUNICATIONS INTERFACE

6 SERIAL PERIPHERAL INTERFACE

7 ANALOG-TO-DIGITAL CONVERTER

8 PROGRAMMABLE TIMER, RTI, AND PULSE ACCUMULATOR

9 RESETS, INTERRUPTS, AND LOW POWER MODES

10 CPU, ADDRESSING MODES, AND INSTRUCTION SET

A ELECTRICAL CHARACTERISTICS

B MECHANICAL DATA AND ORDERING INFORMATION

C DEVELOPMENT SUPPORT

SUMMARY OF CHANGES

LIST OF ILLUSTRATIONS

Figure **Page**

LIST OF TABLES

Table **Page**

1 INTRODUCTION

The HCMOS MC68HC11A8 is an advanced 8-bit microcontroller (MCU) with highly sophisticated on-chip peripheral capabilities. A fully static design and high-density complementary metal-oxide semiconductor (HCMOS) fabrication process allow E-series devices to operate at frequencies from 3 MHz to dc, with very low power consumption.

1.1 Features

The following are some of the hardware and software highlights.

1.1.1 Hardware Features

1

- 8 Kbytes 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

1.1.2 Software Features

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

1.2 General Description

The high-density CMOS technology (HCMOS) used on the MC68HC11A8 combines smaller size and higher speeds with the low power and high noise immunity of CMOS. On-chip memory systems include 8 Kbytes of ROM, 512 bytes of electrically erasable programmable ROM (EEPROM), and 256 bytes of static RAM.

A block diagram of the MC68HC11A8 is shown in **Figure 1-1**. 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.

1

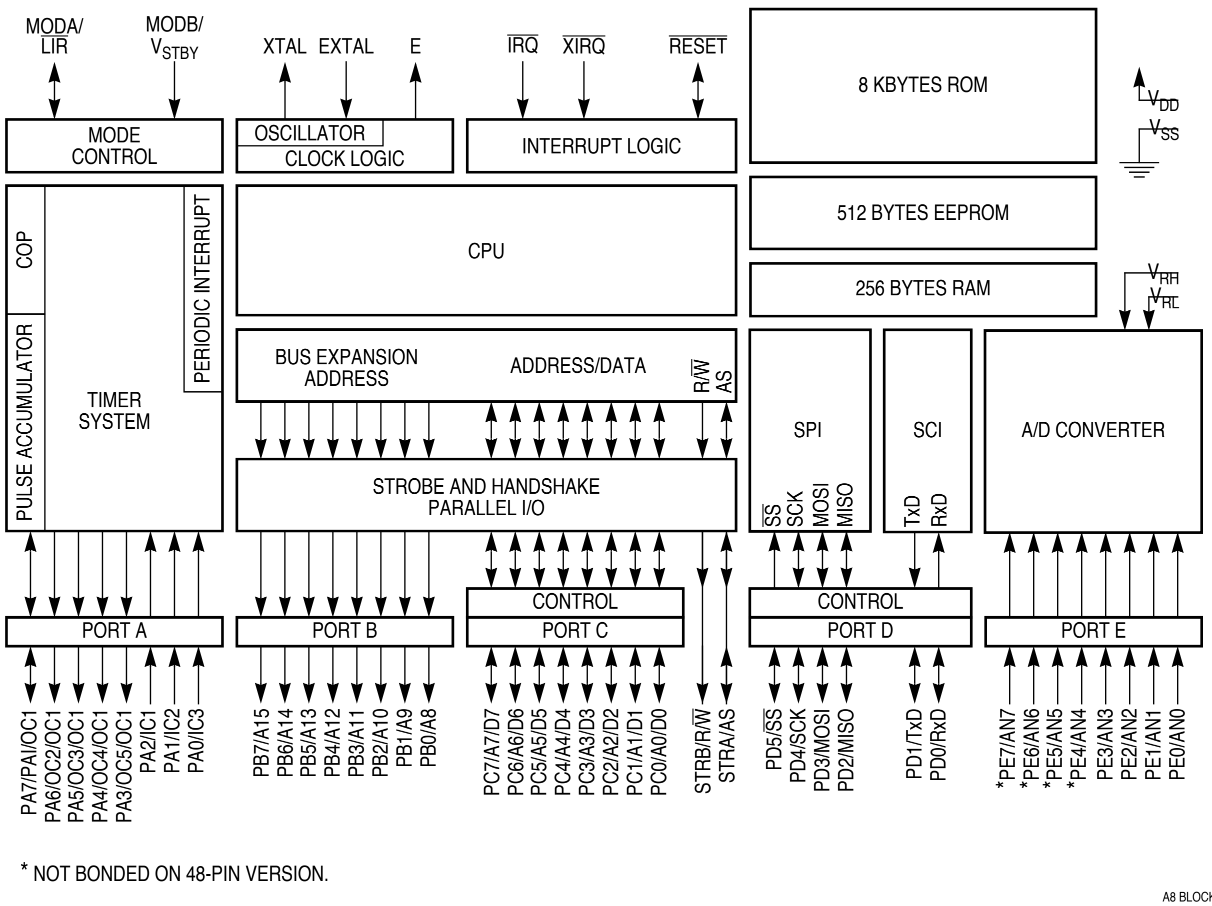
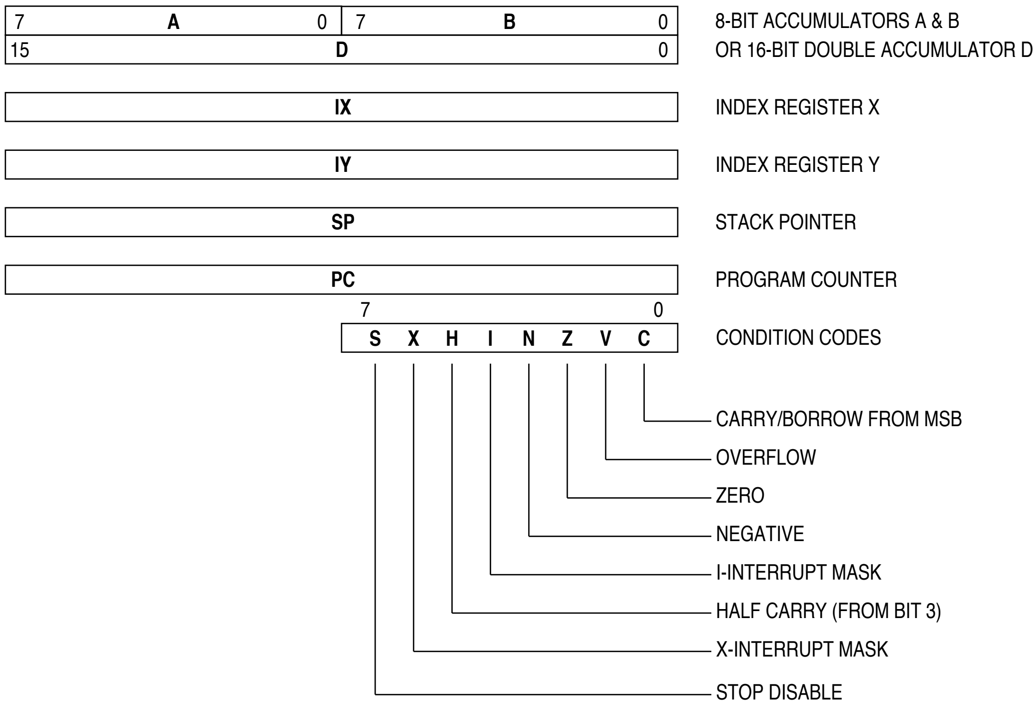


Figure 1-1 Block Diagram

1.3 Programmer's Model

In addition to being able to execute all M6800 and M6801 instructions, the MC68HC11A8 allows execution of 91 new opcodes. **Figure 1-2** shows the seven CPU registers which are available to the programmer.



1

Figure 1-2 Programming Model

1.4 Summary of M68HC11 Family

Table 1-1 and the following paragraphs summarize the current members of the M68HC11 family of MCUs. This technical data book describes the MC68HC11A8 version and can be used as a primary reference for several other versions of the M68HC11 family. However, with the exception of the CPU, some newer members differ greatly from the MC68HC11A8 MCU and their respective technical literature should be referenced.

Several of the device series within the M68HC11 family have ′x1 and ′x0 versions. These are identical to the main member of the series but have some of their on-chip resources disabled. For instance, an MC68HC11A1 is identical to the MC68HC11A8 except that its ROM is disabled. An MC68HC11A0 has disabled EPROM **and** EEPROM arrays. Refer to **Table 1-1**.

Nearly all series within the M68HC11 family have both a ROM version and an EPROM version. Any device in the M68HC11 family that has a 7 preceding the 11 is a device containing EPROM instead of ROM (e.g., MC68HC711E9). These devices operate exactly as the custom ROM-based version (e.g., MC68HC11E9) but can be programmed by the user. EPROM-based devices in a windowed package can be erased and reprogrammed indefinitely. EPROM-based devices in standard packages are one-time-programmable (OTP). Refer to **Table 1-1**.

1

Table 1-1 M68HC11 Family Devices

Device	RAM	ROM	EPROM	EEPROM	COMMENTS
MC68HC11A8	256	8K	0	512	16-bit timer; 8 channel 8-bit A/D, SCI, SPI
MC68HC11A7	256	8K	0	0	
MC68HC11A1	256	0	0	512	
MC68HC11A0	256	0	0	0	
MC68HC11D3	192	4K	0	0	16-bit timer; SCI, SPI
MC68HC711D3	192	0	4K	0	
MC68HC11D0	192	0	0	0	
MC68HC11ED0	512	0	0	0	16-bit timer; SCI, SPI
MC68HC11E9	512	12K	0	512	16-bit timer; SCI, SPI, 8 channel 8-bit A/D
MC68HC711E9	512	0	12K	512	
MC68HC11E8	512	12K	0	0	
MC68HC11E1	512	0	0	512	
MC68HC11E0	512	0	0	0	
MC68HC811E2	256	0	0	2048	16-bit timer; SCI, SPI, 8 channel 8-bit A/D, 2K EEPROM
MC68HC11E20	768	20K	0	512	16-bit timer; SCI, SPI, 8 channel 8-bit A/D,
MC68HC711E20	768	0	20K	512	20K ROM/EPROM
MC68HC11F1	1024	0	0	512	nonmultiplexed bus, 8 channel 8-bit A/D, 4 chip selects, SCI, SPI
MC68HC11G7	512	24K	0	0	nonmultiplexed bus, 8 channel 10-bit A/D, 4 channel PWM,
MC68HC11G5	512	16K	0	0	SCI, SPI, 66 I/O pins
MC68HC711G5	512	0	16K	0	
MC68HC11G0	512	0	0	0	
MC68HC11K4	768	24K	0	640	nonmultiplexed bus, memory expansion to 1MB,
MC68HC711K4	768	0	24K	640	8 channel 8-bit A/D, 4 channel PWM, 4 chip selects
MC68HC11K3	768	24K	0	0	
MC68HC11K1	768	0	0	640	
MC68HC11K0	768	0	0	0	
MC68HC11KA4	768	24K	0	640	nonmultiplexed bus, 8 channel 8-bit A/D, SCI, SPI,
MC68HC711KA4	768	0	24K	640	4 channel PWM
MC68HC11KA2	1024	32K	0	640	
MC68HC711KA2	1024	0	32K	640	
MC68HC11L6	512	16K	0	512	multiplexed bus, 16-bit timer; 8 channel 8-bit A/D, SCI, SPI
MC68HC711L6	512	0	16K	512	
MC68HC11L5	512	16K	0	0	
MC68HC11L1	512	0	0	512	
MC68HC11L0	512	0	0	0	
MC68HC11M2	1280	32K	0	640	nonmultiplexed bus, 8 channel 8-bit A/D, 4 channel PWM,
MC68HC711M2	1280	0	32K	640	DMA, on-chip math coprocessor, SCI, 2 SPI
MC68HC11N4	768	24K	0	640	nonmultiplexed bus, 12 channel 8-bit A/D, 2 channel 8-bit D/A,
MC68HC711N4	768	0	24K	640	6 channel PWM, on-chip math coprocessor, SCI, SPI
MC68HC11P2	1024	32K	0	640	nonmultiplexed bus, PLL, 8 channel 8-bit A/D, 4 channel PWM,
MC68HC711P2	1024	0	32K	640	3 SCI (2 with MI bus), SPI, 62 I/O pins

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 MC68HC11A8 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 circuitry in the system, should also be present on the circuit board.

2

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

This active low bidirectional control signal is used as an input to initialize the MC68HC11A8 to a known start-up 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 **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 M68HC11.

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 **Figure 2-1**, **Figure 2-2**, and **Figure 2-3** for diagrams of oscillator circuits.

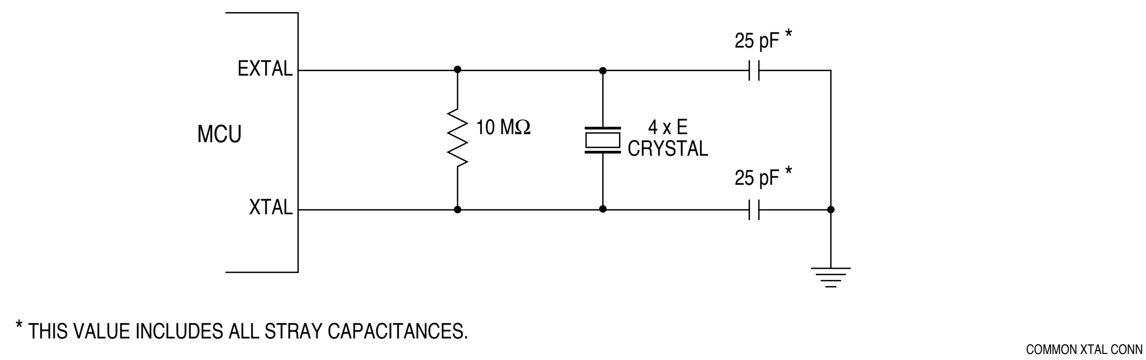


Figure 2-1 Common Crystal Connections

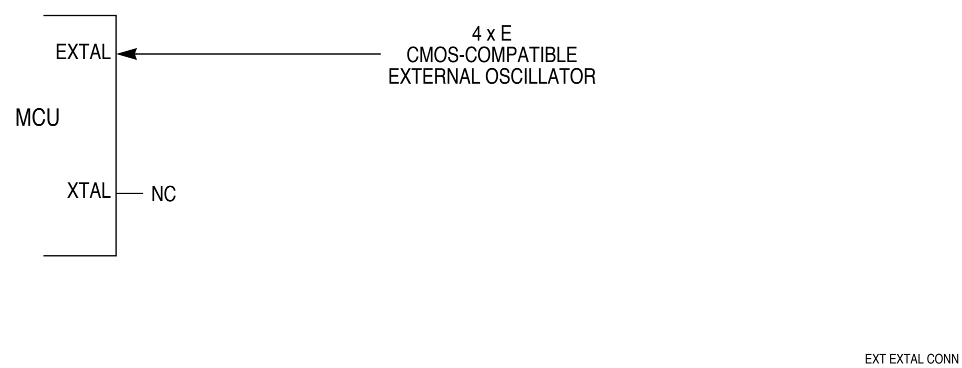


Figure 2-2 External Oscillator Connections

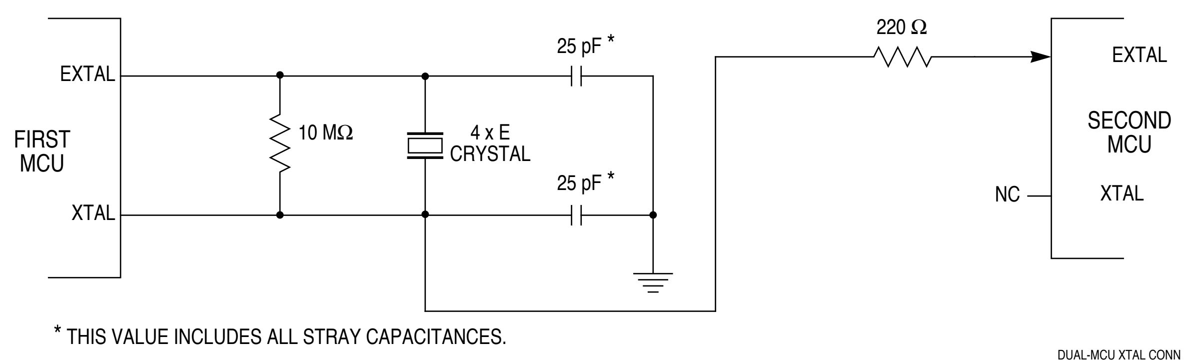


Figure 2-3 One Crystal Driving Two MCUs

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 ($\overline{\text{IRQ}}$)

The $\overline{\text{IRQ}}$ input provides a means for requesting asynchronous interrupts to the MC68HC11A8. 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 $\overline{\text{IRQ}}$ pin requires an external pull-up resistor to V_{DD} (typically 4.7K ohm).

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

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 $\overline{\text{XIRQ}}$ input is level sensitive and requires an external pull-up resistor to V_{DD}.

2

2.1.7 Mode A/Load Instruction Register and Mode B/Standby Voltage (MODA/$\overline{\text{LIR}}$, 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 vs. 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 $\overline{\text{LIR}}$ 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 $\overline{\text{LIR}}$ 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 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 **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 **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 singlechip 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.

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 used 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/$\overline{\text{W}}$ 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 ($\overline{\text{SS}}$) 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 MC68HC11A8: 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

2

In single-chip operating mode, the MC68HC11A8 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 **4 PARALLEL I/O** for additional information.

2.2.2 Expanded Multiplexed Operating Mode

In expanded multiplexed operating mode, the MC68HC11A8 has the capability of accessing a 64 Kbyte 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.

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/OC1	PA3/OC5/OC1
A-4	PA4/OC4/OC1	PA4/OC4/OC1
A-5	PA5/OC3/OC1	PA5/OC3/OC1
A-6	PA6/OC2/OC1	PA6/OC2/OC1
A-7	PA7/PAI/OC1	PA7/PAI/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/SS	PD5/SS
	STRA	AS
	STRB	R/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##

Not bonded in 48-pin versions

2

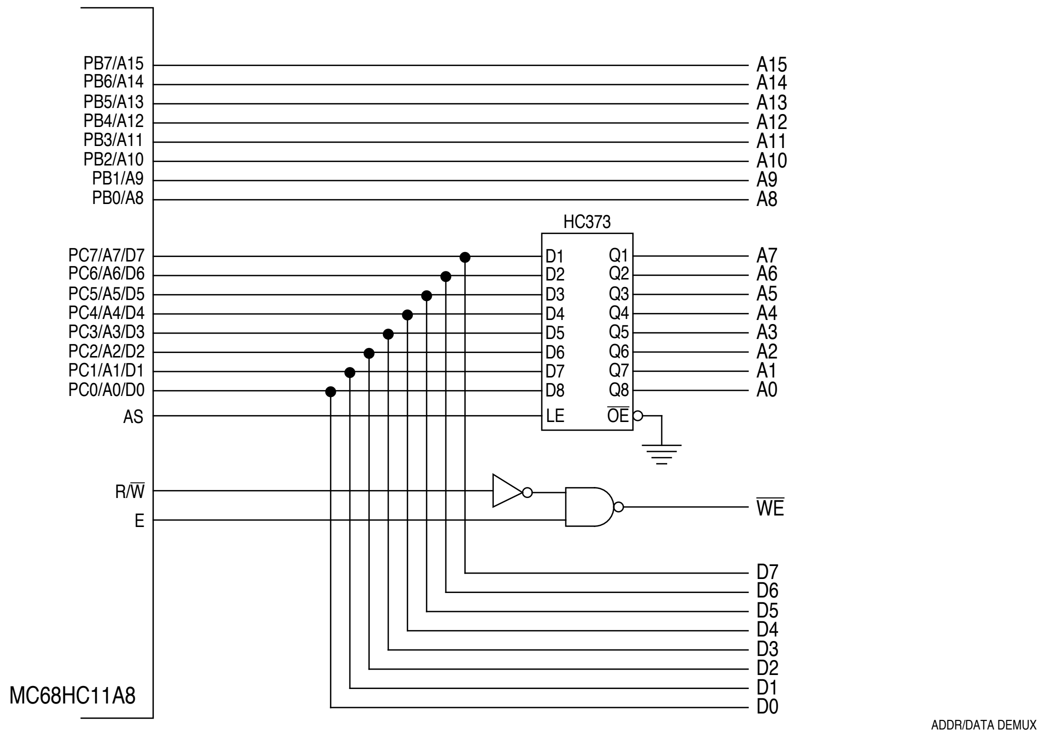


Figure 2-4 Address/Data Demultiplexing

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 MC68HC11A8 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 = 2 MHz) or E clock/104 (1200 baud for E clock = 2 MHz).

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 pull-up 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.

2

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 MEMORY

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.

3

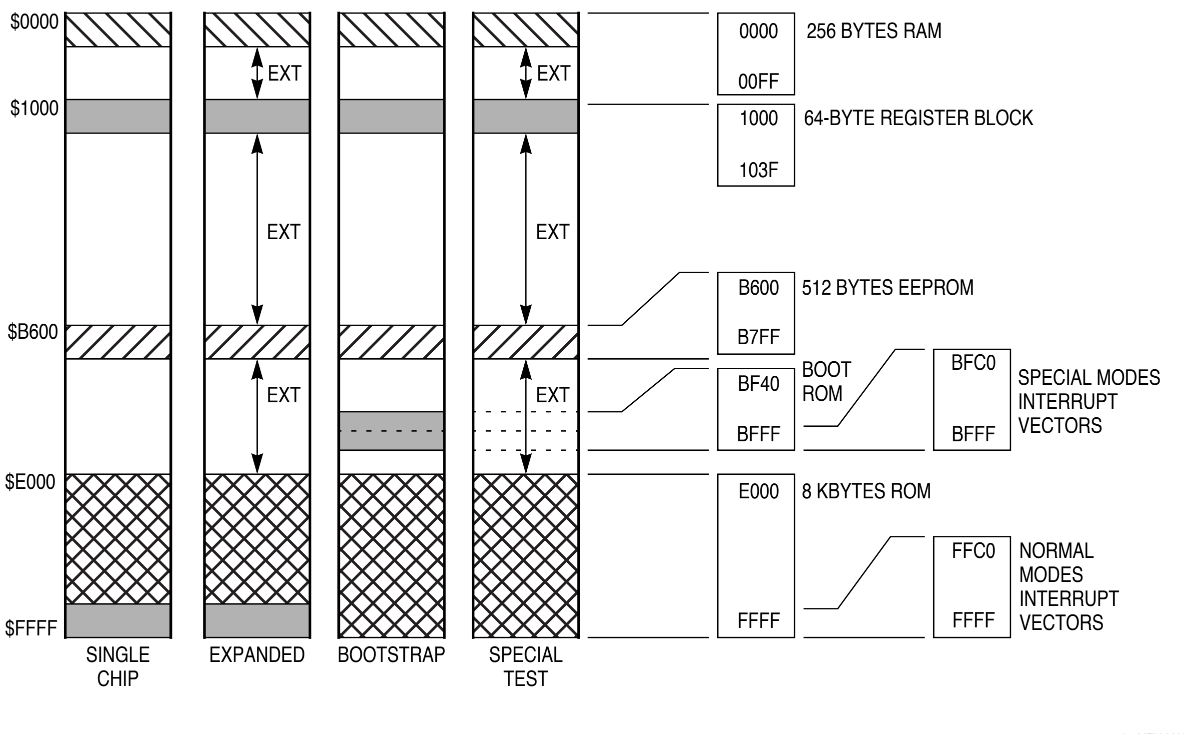


Figure 3-1 Memory Maps

3

Table 3-1 Register and Control Bit Assignments (Sheet 1 of 2)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0		
$1000	Bit 7	—	—	—	—	—	—	Bit 0	PORTA	I/O Port A
$1001									Reserved	
$1002	STAF	STAI	CWOM	HNDS	OIN	PLS	EGA	INVB	PIOC	Parallel I/O Control Register
$1003	Bit 7	—	—	—	—	—	—	Bit 0	PORTC	I/O Port C
$1004	Bit 7	—	—	—	—	—	—	Bit 0	PORTB	Output Port B
$1005	Bit 7	—	—	—	—	—	—	Bit 0	PORTCL	Alternate Latched Port C
$1006									Reserved	
$1007	Bit 7	—	—	—	—	—	—	Bit 0	DDRC	Data Direction for Port C
$1008			Bit 5	—	—	—	—	Bit 0	PORTD	I/O Port D
$1009			Bit 5	—	—	—	—	Bit 0	DDRD	Data Direction for Port D
$100A	Bit 7	—	—	—	—	—	—	Bit 0	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	Bit 15	—	—	—	—	—	—	Bit 8	TCNT	Timer Counter Register
$100F	Bit 7	—	—	—	—	—	—	Bit 0		
$1010	Bit 15	—	—	—	—	—	—	Bit 8	TIC1	Input Capture 1 Register
$1011	Bit 7	—	—	—	—	—	—	Bit 0		
$1012	Bit 15	—	—	—	—	—	—	Bit 8	TIC2	Input Capture 2 Register
$1013	Bit 7	—	—	—	—	—	—	Bit 0		
$1014	Bit 15	—	—	—	—	—	—	Bit 8	TIC3	Input Capture 3 Register
$1015	Bit 7	—	—	—	—	—	—	Bit 0		
$1016	Bit 15	—	—	—	—	—	—	Bit 8	TOC1	Output Compare 1 Register
$1017	Bit 7	—	—	—	—	—	—	Bit 0		
$1018	Bit 15	—	—	—	—	—	—	Bit 8	TOC2	Output Compare 2 Register
$1019	Bit 7	—	—	—	—	—	—	Bit 0		
$101A	Bit 15	—	—	—	—	—	—	Bit 8	TOC3	Output Compare 3 Register
$101B	Bit 7	—	—	—	—	—	—	Bit 0		
$101C	Bit 15	—	—	—	—	—	—	Bit 8	TOC4	Output Compare 4 Register
$101D	Bit 7	—	—	—	—	—	—	Bit 0		
$101E	Bit 15	—	—	—	—	—	—	Bit 8	TCO5	Output Compare 5 Register
$101F	Bit 7	—	—	—	—	—	—	Bit 0		

Table 3-1 Register and Control Bit Assignments (Sheet 2 of 2)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0		
$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	Bit 7	—	—	—	—	—	—	Bit 0	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	Bit 7	—	—	—	—	—	—	Bit 0	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	TRDE	TC	RDRF	IDLE	OR	NF	FE		SCSR	SCI Status Register
$102F	Bit 7	—	—	—	—	—	—	Bit 0	SCDR	SCI Data (Read RDR, Write TDR)
$1030	CCF		SCAN	MULT	CD	CC	CB	CA	ADCTL	A/D Control Register
$1031	Bit 7	—	—	—	—	—	—	Bit 0	ADR1	A/D Result Register 1
$1032	Bit 7	—	—	—	—	—	—	Bit 0	ADR2	A/D Result Register 2
$1033	Bit 7	—	—	—	—	—	—	Bit 0	ADR3	A/D Result Register 3
$1034	Bit 7	—	—	—	—	—	—	Bit 0	ADR4	A/D Result Register 4
$1035 thru $1038									Reserved	
$1039	ADPU	CSEL	IRQE	DLY	CME		CR1	CR0	OPTION	System Configuration Options
$103A	Bit 7	—	—	—	—	—	—	Bit 0	COPRST	Arm/Reset COP Timer Circuitry
$103B	ODD	EVEN		BYTE	ROW	ERASE	EELAT	EEPGM	PPROG	EEPROM Program Control Register
$103C	RBOOT	SMOD	MDA	IRV	PSEL3	PSEL2	PSEL1	PSEL0	HPRIO	Highest Priority I-Bit Int and Misc
$103D	RAM3	RAM2	RAM1	RAM0	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, 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/$\overline{\text{W}}$ signal in their development. The R/$\overline{\text{W}}$, 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.

3

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 64 E clock cycles after a reset and thereafter becomes a read-only register.

	7	6	5	4	3	2	1	0	
$**1**03D	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 at 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 MC68HC11A8. 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, $\overline{\text{RESET}}$ 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 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

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. 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.

3

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.

3.5.2.1 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 = 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

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

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 MC68HC11A8 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 MC68HC11A8s.

	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 Configuration 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 submitted. 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 the 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 MC68HC11 Family member you are using.

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 onto 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

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 start-up 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.

CAUTION

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 MC68HC11A8 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 registers. 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)

4

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/$\overline{\text{W}}$ 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.

4

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 impractical to use some port C lines as static inputs, while using others as handshake outputs, 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.

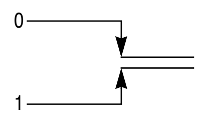
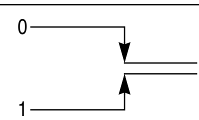
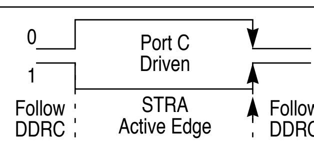
4.5 Parallel I/O Control Register (PIOC)

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.

4

Table 4-1 Handshake I/O Operations 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	X	X		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 I bit in the condition code register is clear and STAI is set, STAF (when set) will request an interrupt.

CWOM — Port C Wire-OR Mode

CWOM affects all eight port C pins 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 OIN 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 select
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

4

5 SERIAL COMMUNICATIONS INTERFACE

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.

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.
- 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.

5

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.
- 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.

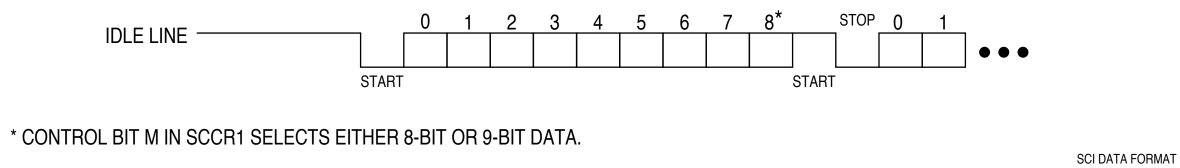


Figure 5-1 Data Format

5

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.

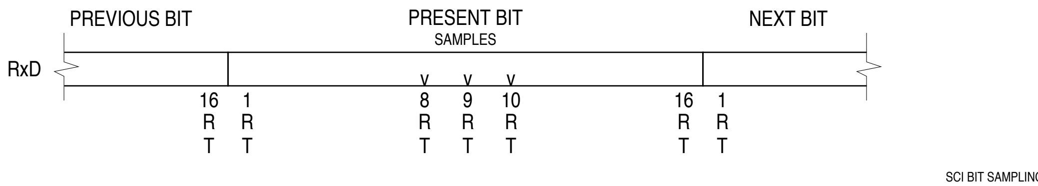


Figure 5-2 Sampling Technique Used on All Bits

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.

5

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**.

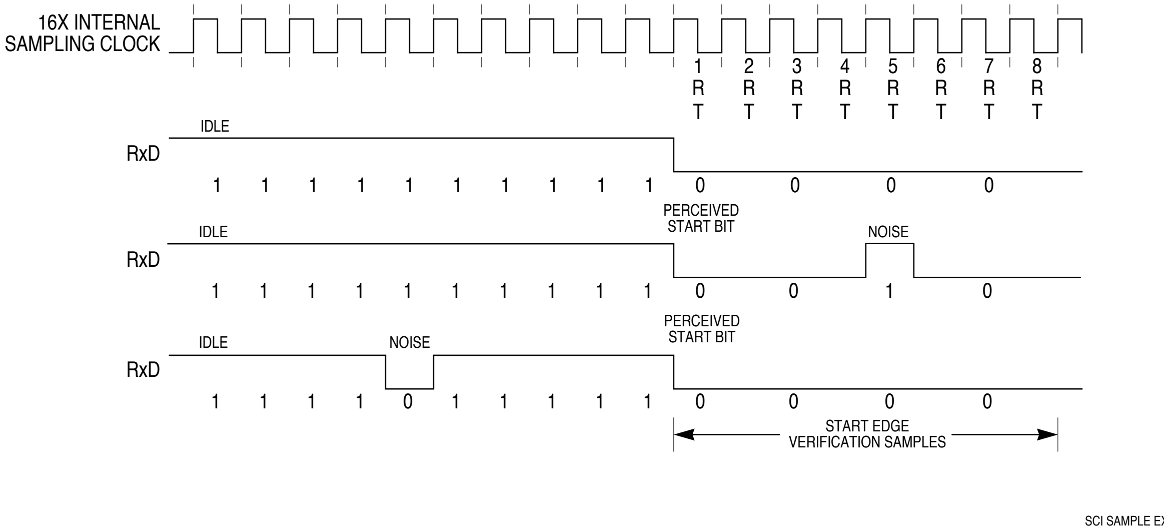
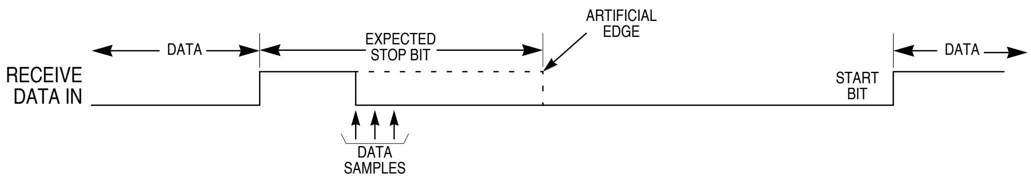
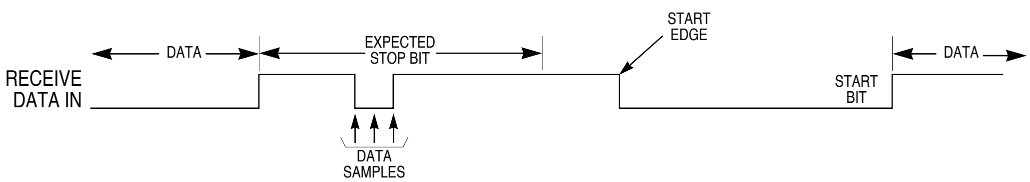


Figure 5-3 Examples of Start Bit Sampling Techniques

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

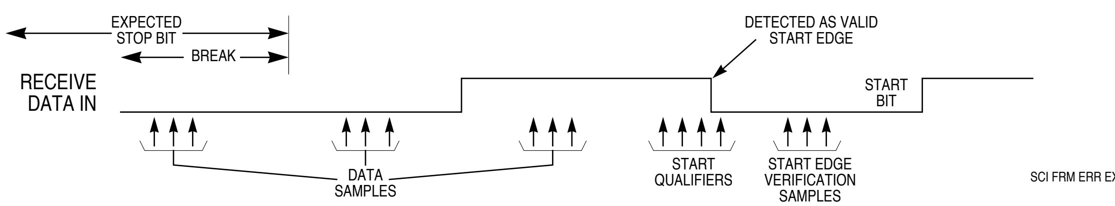


Figure 5-5 SCI Start Bit Following a Break

5.6 Transmit Data (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.

5

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 (over-run), NF (noise), or 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 line interrupt will not be generated.

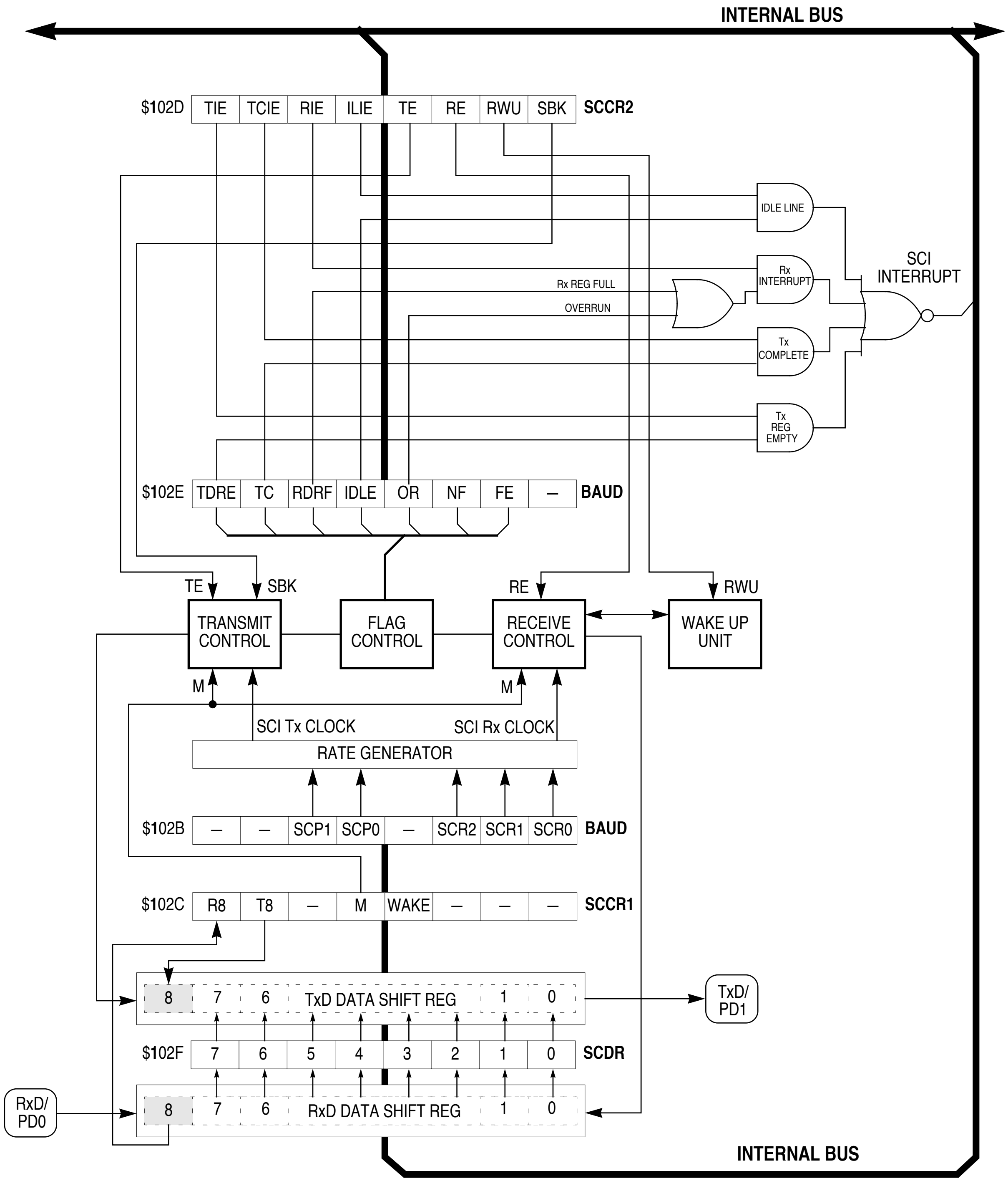
5.8 SCI Registers

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.

5



5

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.

5

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 2-0 — 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

RIE — 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 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. 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 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. 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 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 is cleared by the SCI logic after receiving 10 (M = 0) or 11 (M = 1) consecutive ones. If the WAKE bit is set, RWU is cleared by the SCI logic after receiving a data word whose MSB is set.

SBK — Send Break

If the send break bit is toggled set and cleared, the transmitter sends 10 (M = 0) or 11 (M = 1) zeros and then reverts to idle or sending data. If SBK 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 is likely to queue two character times of break because the first break transfers almost immediately 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 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 (with TDRE = 1) followed by a write to the SCDR.

5

TC — Transmit Complete

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 have occurred.

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

RDRF — Receive Data Register Full

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

IDLE — Idle Line Detect

The idle line detect bit, when set, indicates the receiver has detected an idle line. The IDLE bit is cleared by reading the SCSR with IDLE set followed by reading SCDR. 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 bit is set when the next byte is ready to be transferred from the receive shift register to the SCDR 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 is not disturbed. The OR is cleared when the SCSR is read (with OR set), followed by a read of the SCDR.

NF — Noise Flag

The noise flag 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 is read (with NF set), followed by a read of the SCDR.

FE — Framing Error

The framing error 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 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 until it is cleared. The FE bit is cleared when the SCSR is read (with FE equal to one) followed by a read of the SCDR.

Bit 0 — Not Implemented

This bit always reads zero.

5.8.5 Baud Rate Register (BAUD)

5

The baud rate register selects the different baud rates which may be used as the rate control for the transmitter and receiver. The SCP[0:1] bits function as a prescaler for the SCR[0:2] 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 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 First Prescaler Stage

SCP1	SCP0	Internal Processor Clock Divided 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 is zero and cannot be set while in normal operating modes.

5

The diagram shown in **Figure 5-7** and the data given in **Table 5-3** and **Table 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 SCP[1:0] and SCR[2:0] bits in the baud rate register as illustrated.

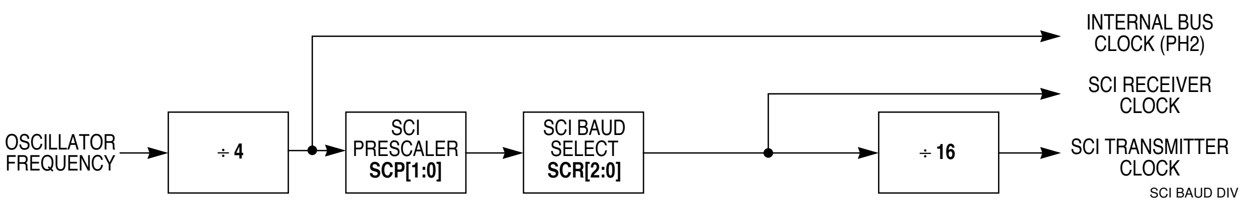


Figure 5-7 Rate Generator Division

Table 5-3 Prescaler Highest Baud Rate Frequency Output

SCP Bit		Clock*	Crystal Frequency (MHz)					
1	0	Divided By	12.0	8.3886	8.0	4.9152	4.0	3.6864
0	0	1	187.50 K Baud	131.072 K Baud	125.000 K Baud	76.80 K Baud	62.50 K Baud	57.60 K Baud
0	1	3	62.50 K Baud	43.690 K Baud	41.666 K Baud	25.60 K Baud	20.833 K Baud	19.20 K Baud
1	0	4	46.875 K Baud	32.768 K Baud	31.250 K Baud	19.20 K Baud	15.625 K Baud	14.40 K Baud
1	1	13	14.423 K Baud	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 **Table 5-3** 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 Bit			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**	**4800 Baud**
0	0	0	1	131.072 K Baud	32.768 K Baud	76.80 K Baud	19.20 K Baud	9600 Baud	4800 Baud
0	0	1	2	65.536 K Baud	16.384 K Baud	38.40 K Baud	9600 Baud	4800 Baud	2400 Baud
0	1	0	4	32.768 K Baud	8.192 K Baud	19.20 K Baud	4800 Baud	2400 Baud	1200 Baud
0	1	1	8	16.384 K Baud	4.096 K Baud	9600 Baud	2400 Baud	1200 Baud	600 Baud
1	0	0	16	8.192 K Baud	2.048 K Baud	4800 Baud	1200 Baud	600 Baud	300 Baud
1	0	1	32	4.096 K Baud	1.024 K Baud	2400 Baud	600 Baud	300 Baud	150 Baud
1	1	0	64	2.048 K Baud	512 Baud	1200 Baud	300 Baud	150 Baud	75 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 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.

5

6 SERIAL PERIPHERAL INTERFACE

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 MC68HC11A8 SPI system may be configured either as a master or as a slave. The SPI contains the following features:

- Full Duplex, Three-Wire Synchronous Transfers
- Master or Slave Operation
- 1.5 MHz (Maximum) Master Bit Frequency
- 3 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

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 as 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 operate 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 of 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 the 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.

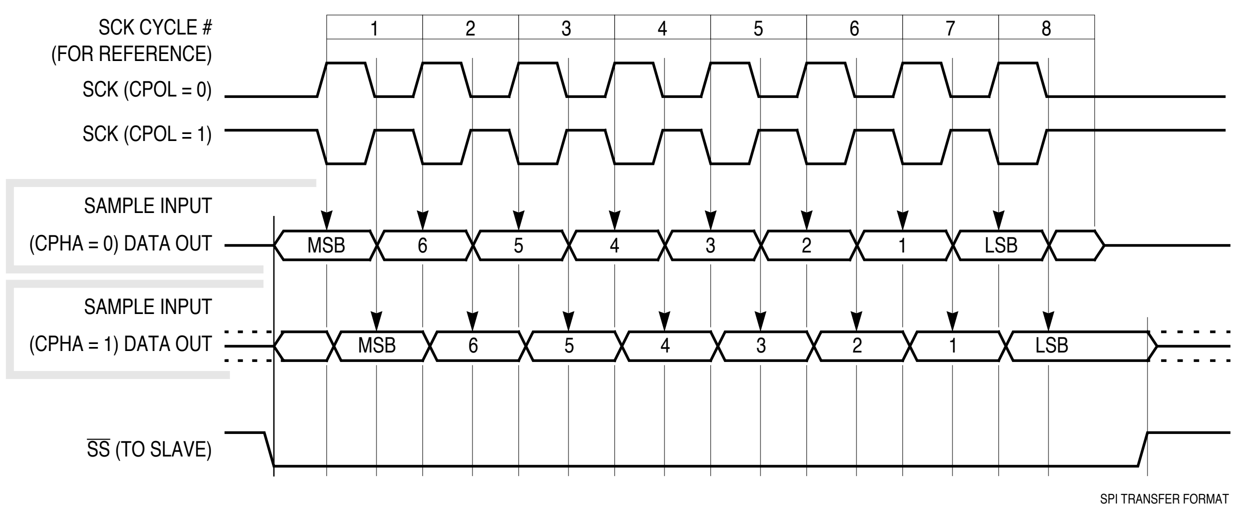


Figure 6-1 Data Clock Timing Diagram

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 MOSI 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.

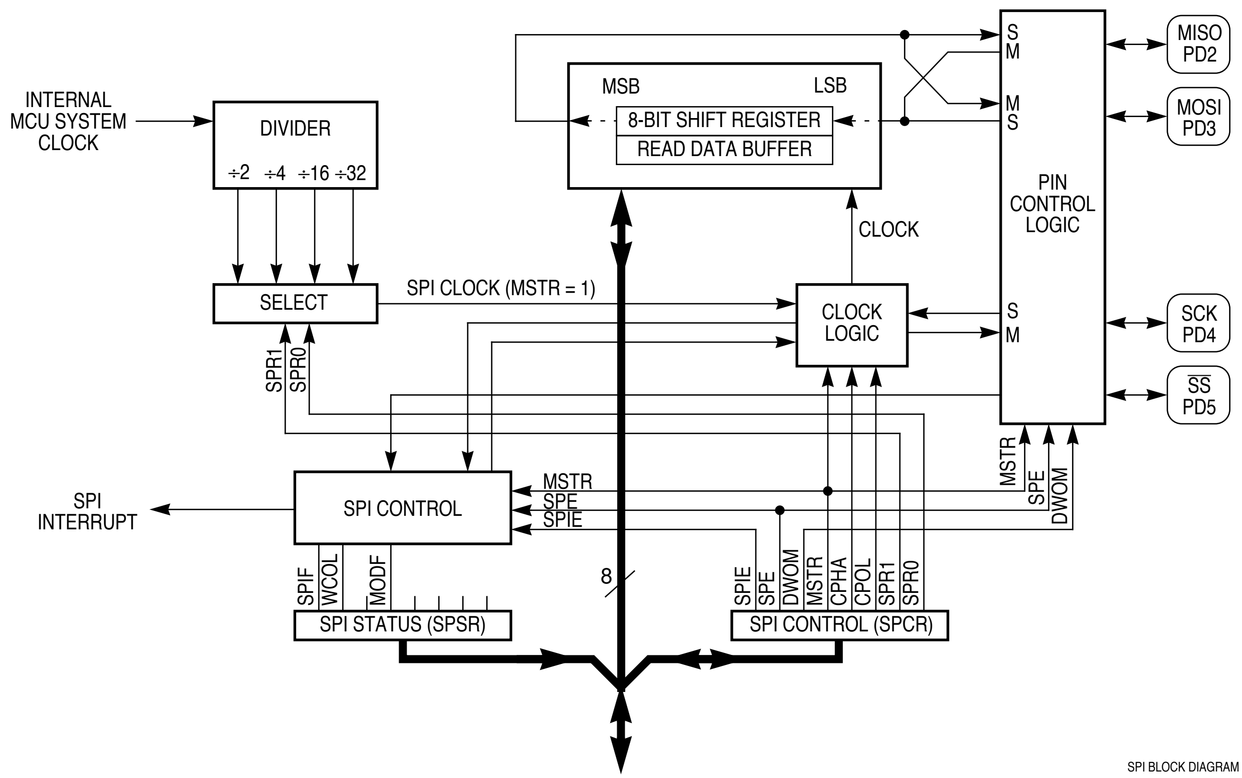


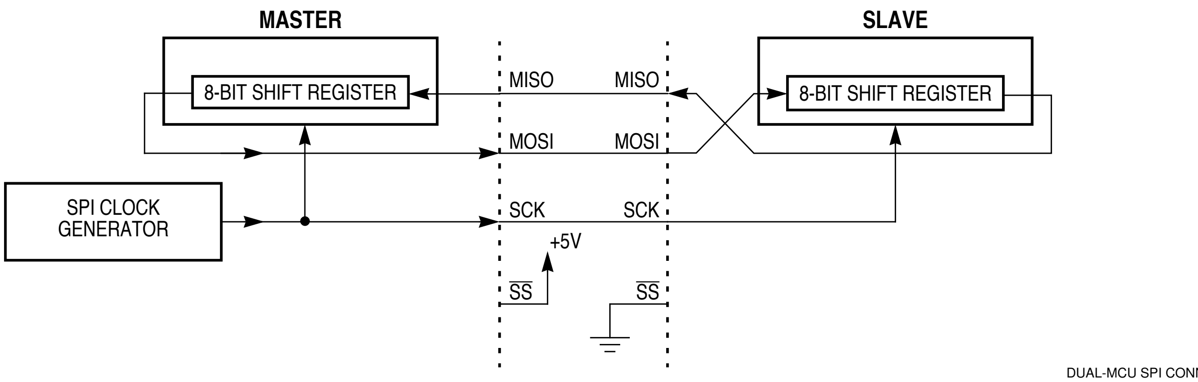
Figure 6-2 Serial Peripheral Interface Block Diagram

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 wire-OR mode (DWOM) option, the SPI system can be configured in a variety of ways. Systems with a single bidirectional data path rather than separate MISO and MOSI paths can be accommodated. Since MC68HC11A8 SPI slaves can selectively disable their MISO output, a broadcast message protocol is also possible.



6

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 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 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 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**.

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

SPR1 and SPR0—SPI Clock Rate Selects
These two serial peripheral rate bits 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.

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

WCOL — Write Collision

The write collision bit is set when an attempt is made to write to the serial peripheral data register while data transfer is taking place. If CPHA 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 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 (with WCOL set) followed by an access to SPDR.

Bit 5 — Not Implemented

This bit always reads zero.

MODF — Mode Fault

The mode fault 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 (with MODF set), followed by a write to the SPCR. 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 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 MC68HC11A8 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 ± 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} = 0 V and V_{RH} = 5 V ± 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 3 V. 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 5.0 V). 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

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.

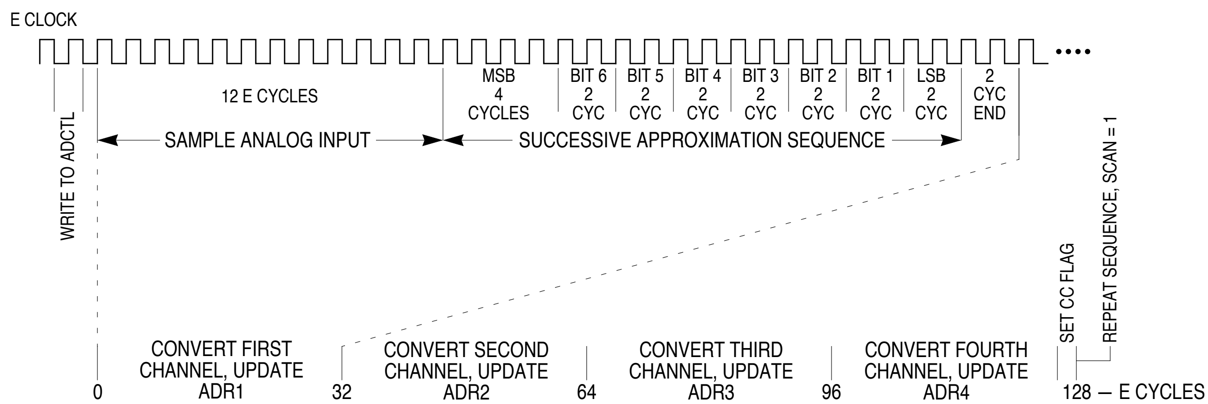


Figure 7-1 A/D Conversion Sequence

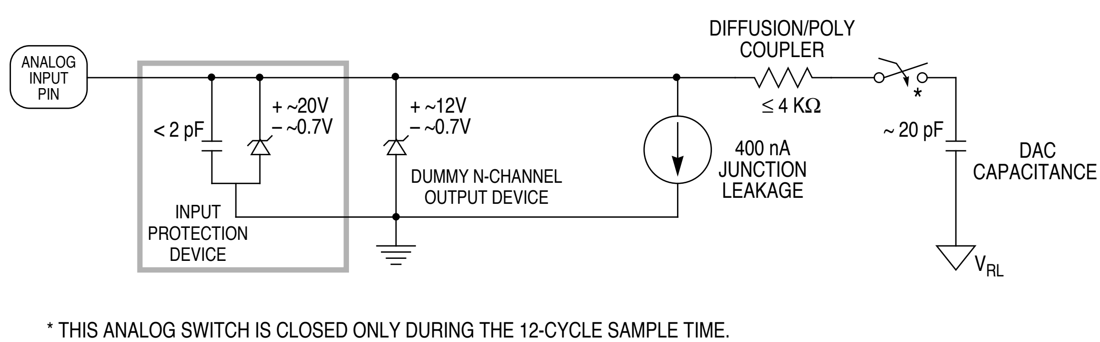


Figure 7-2 A/D Pin Model

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 entered, 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 read-only status indicator and bit 6 which always reads as a zero.

7

	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.

CAUTION

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

7

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 read-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 channel assignments 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 1.5 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 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.

7

8 PROGRAMMABLE TIMER, RTI, 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 from 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

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 $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 successful compare regardless of whether or not the OCxF flag was previously clear.

8

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

After a 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.

Writes 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 with 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 occurring immediately before or after the force may result in undesirable operation.

8

	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 bits 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 is 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	

8

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	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 Capture 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	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

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 OCxF 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 operate 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 conditions 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 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	
$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 Capture 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)

8

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 corresponding 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 PR0 — Timer Prescaler Selects

These two bits may be read at any time but may only be written during initialization. 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	Prescaler
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.

8

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, an 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

8

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 enabled.

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 configured as an 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.

8.3.1 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	0	PAI Falling Edge Increments the Counter
0	1	PAI Rising Edge Increments the Counter
1	0	A zero on PAI Inhibits Counting
1	1	A one 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 time 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.

8

Table 8-1 Real Time Interrupt Rate versus RTR1 and RTR0

RTR1	RTR0	Rate	XTAL = 12.0 MHz	XTAL = 2^{23}	XTAL = 8.0 MHz	XTAL = 4.9152 MHz	XTAL = 4.0 MHz	XTAL = 3.6864 MHz
0	0	$2^{13} \div E$	8.192 ms	3.91 ms	4.10 ms	6.67 ms	8.19 ms	8.89 ms
0	1	$2^{14} \div E$	16.384 ms	7.81 ms	8.19 ms	13.33 ms	16.38 ms	17.78 ms
1	0	$2^{15} \div E$	32.768 ms	15.62 ms	16.38 ms	26.67 ms	32.77 ms	35.56 ms
1	1	$2^{16} \div E$	65.536 ms	31.25 ms	32.77 ms	53.33 ms	65.54 ms	71.11 ms
		E =	**3.0 MHz**	**2.1 MHz**	**2.0 MHz**	**1.2288 MHz**	**1.0 MHz**	**921.6 kHz**

8

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

9

The $\overline{\text{RESET}}$ pin is used to reset the MCU and allow an orderly software start-up 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. **Figure 9-2** shows an example of reset circuits with LVI capabilities. The best circuit for a particular application may be different from the suggested circuit.

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

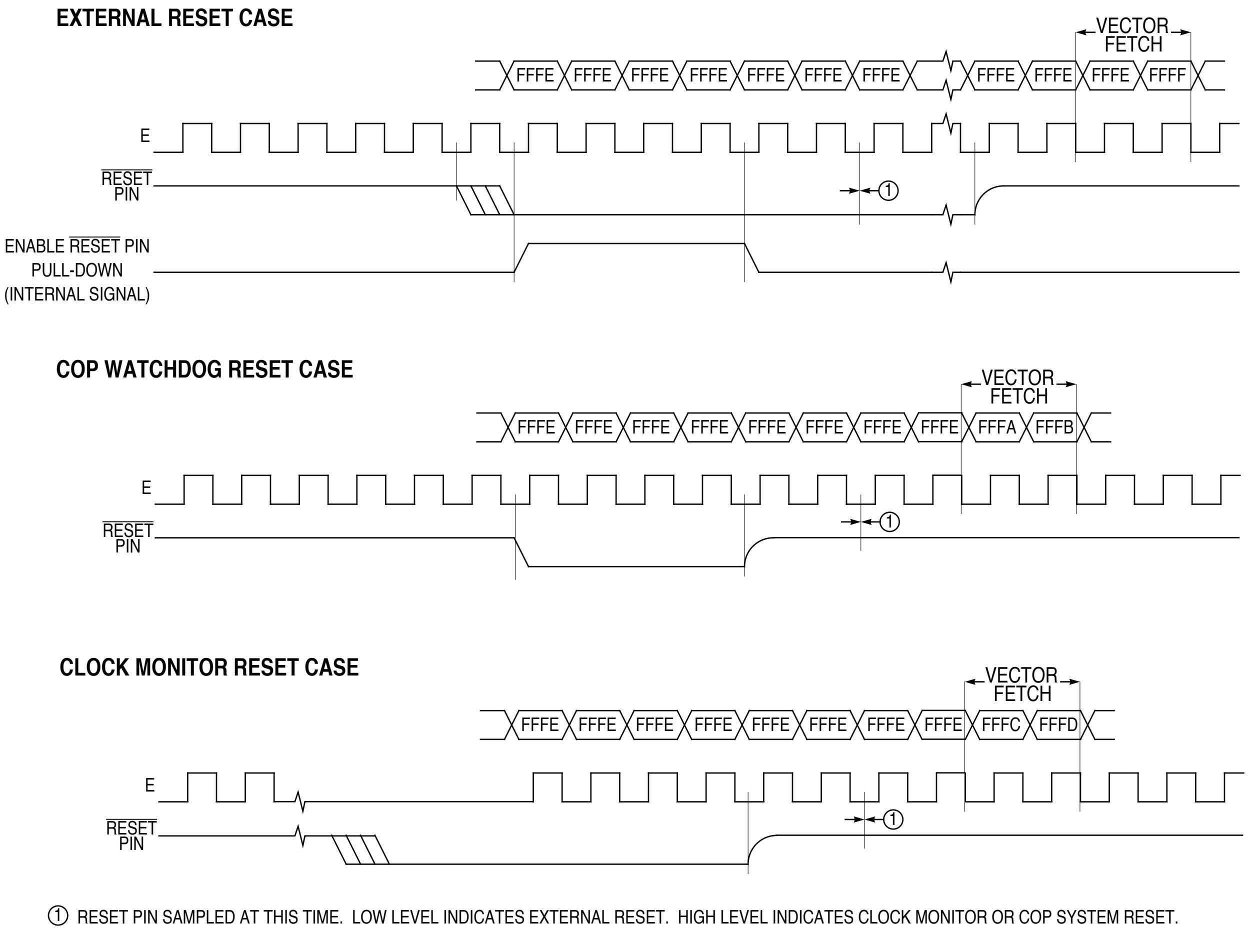


Figure 9-1 Reset Timing

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.

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.

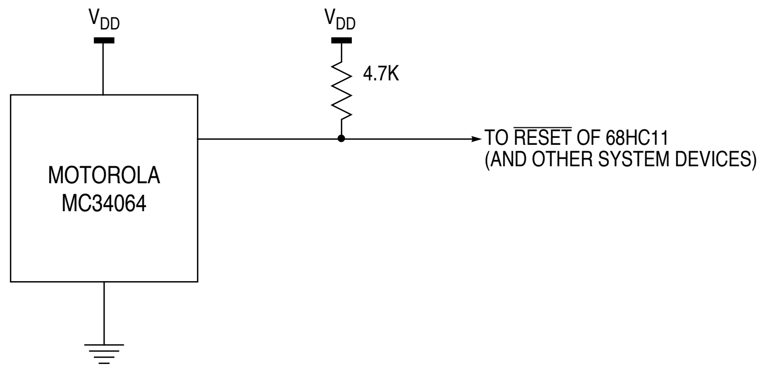


Figure 9-2 Simple LVI Reset Circuit

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-byte 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

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 CWOM 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 a general purpose output port with all bits cleared. 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 STRB 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 hardware 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 are 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 or 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 disabled 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 reset 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 PSEL[3:0] 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 watchdog 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 modes to prevent the COP system from causing a hardware reset.

9

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	Rate	XTAL = 12.0 MHz Timeout –0/+10.9 ms	XTAL = 2^{23} Timeout – 0/+15.6 ms	XTAL = 8.0 MHz Timeout – 0/+16.4 ms	XTAL = 4.9152 MHz Timeout – 0/+26.7 ms	XTAL = 4.0 MHz Timeout – 0/+32.8 ms	XTAL = 3.6864 MHz Timeout – 0/+35.6 ms
0	0	$2^{15} \div E$	10.923 ms	15.625 ms	16.384 ms	26.667 ms	32.768 ms	35.556 ms
0	1	$2^{17} \div E$	43.691 ms	62.5 ms	65.536 ms	106.67 ms	131.07 ms	142.22 ms
1	0	$2^{19} \div E$	174.76 ms	250 ms	262.14 ms	426.67 ms	524.29 ms	568.89 ms
1	1	$2^{21} \div E$	699.05 ms	1 s	1.049 s	1.707 s	2.1 s	2.276 s
		E =	**3.0 MHz**	**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) at $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 sequences 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.

9

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 accidental 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 Power-up

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 2 MHz. When running with an E clock below 1 MHz, CSEL must be high to program or erase EEPROM. When operating below 750 kHz 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 $\overline{\text{IRQ}}$ pin is configured for 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 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 hierarchy (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 MC68HC11A8 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.

Table 9-2 IRQ Vector Interrupts

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

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. **Table 9-2**, **Table 9-3**, and **Table 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 9-3** shows the interrupt stacking order.

Table 9-3 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 Capture 3	I Bit	OC3I
FFEC, ED	Timer Input Capture 2	I Bit	OC2I
FFEE, EF	Timer Input Capture 1	I Bit	OC1I
FFF0, F1	Real Time Interrupt	I Bit	RTII
FFF2, F3	$\overline{\text{IRQ}}$ (External Pin or Parallel I/O)	I Bit	See Table 9-4
FFF4, F5	$\overline{\text{XIRQ}}$ Pin (Pseudo 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	$\overline{\text{RESET}}$	None	None

Table 9-4 SCI Serial System Interrupts

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

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 an interrupt is pending. However, once an SWI instruction has begun, no interrupt can be honored until the SWI vector has been fetched.

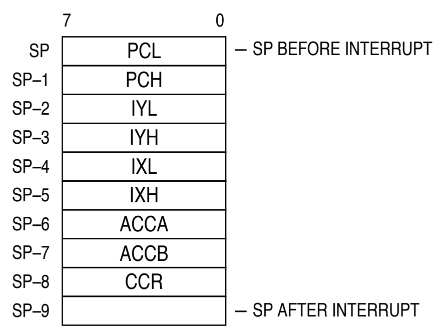


Figure 9-3 Interrupt Stacking Order

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 $\overline{XIRQ}$ pin remains effectively non-masked. In the interrupt priority logic, the $\overline{XIRQ}$ interrupt is a higher priority than any source that is maskable by the I bit. All I bit related 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 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 $\overline{XIRQ}$. (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.

Figure 9-4, **Figure 9-5**, and **Figure 9-6** illustrate the interrupt process as it relates to normal processing. **Figure 9-4** shows how the CPU begins from a reset and how interrupt detection relates to normal opcode fetches. **Figure 9-5** is an expansion of a block in **Figure 9-4** and shows how interrupt priority is resolved. **Figure 9-6** is an expansion of the SCI interrupt block in **Figure 9-5**. **Figure 9-6** 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 miscellaneous 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.

9

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 MDA 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.

9

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).

9

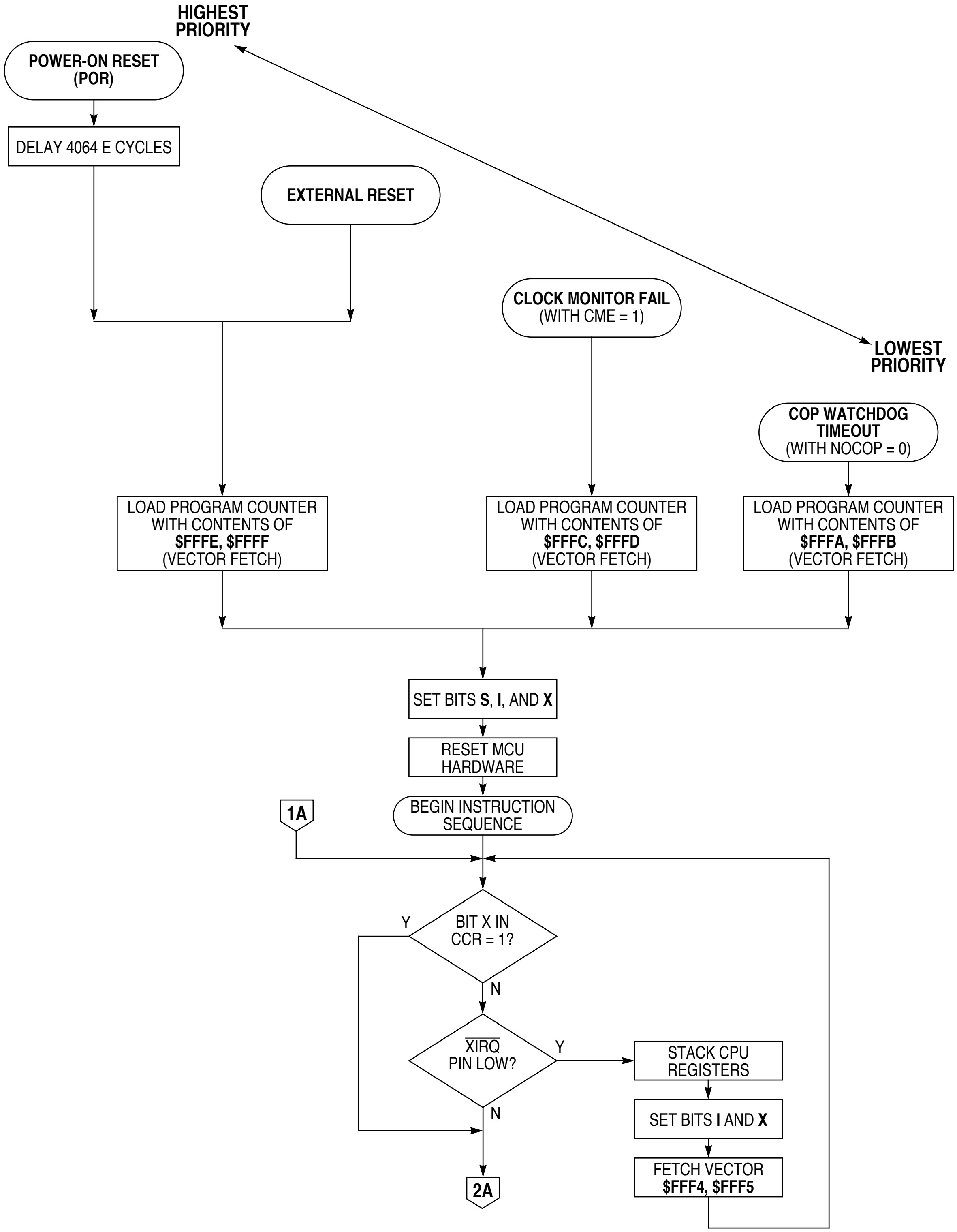


Figure 9-4 Processing Flow Out of Resets (Sheet 1 of 2)

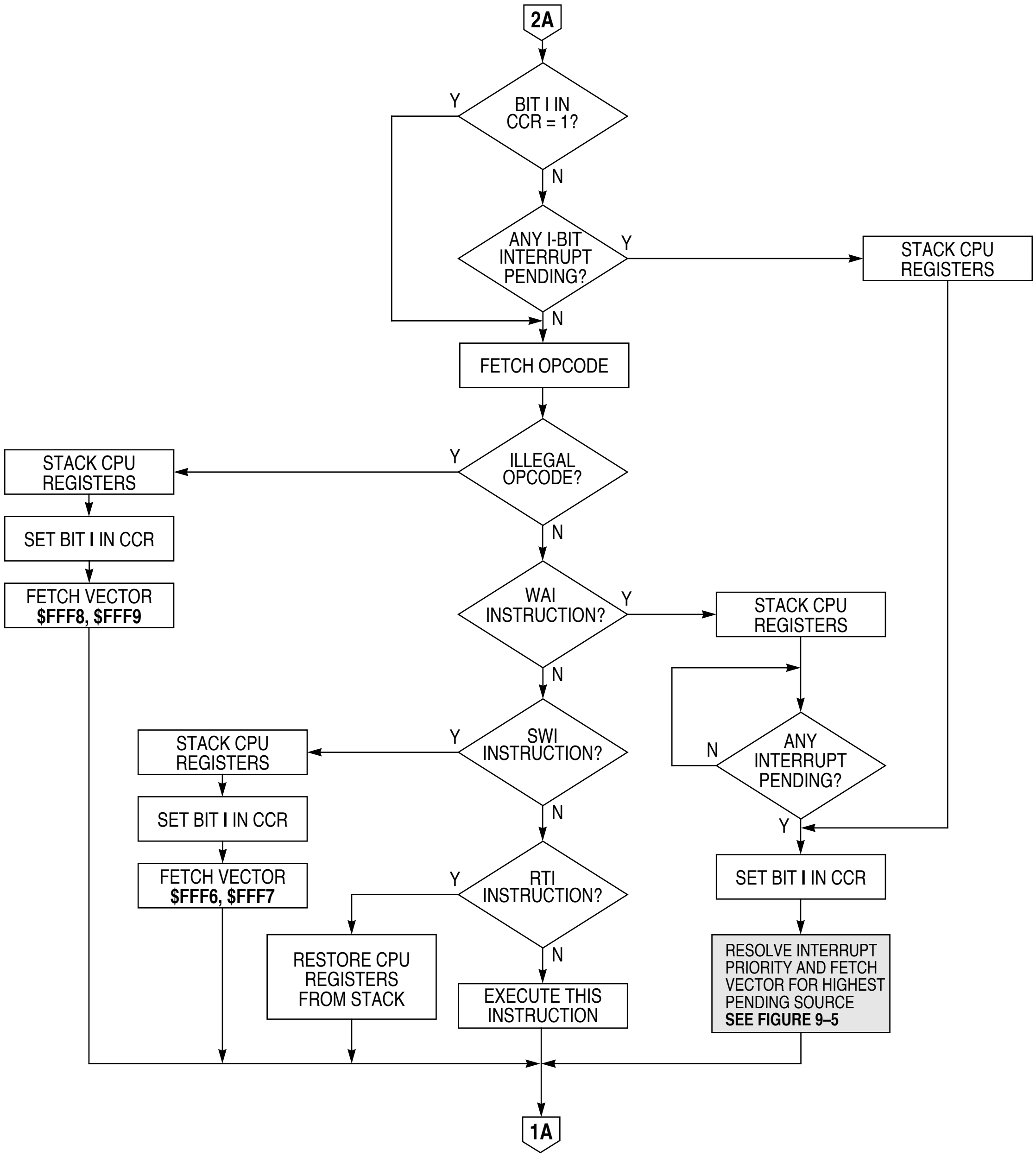


Figure 9-4 Processing Flow Out of Resets (Sheet 2 of 2)

9

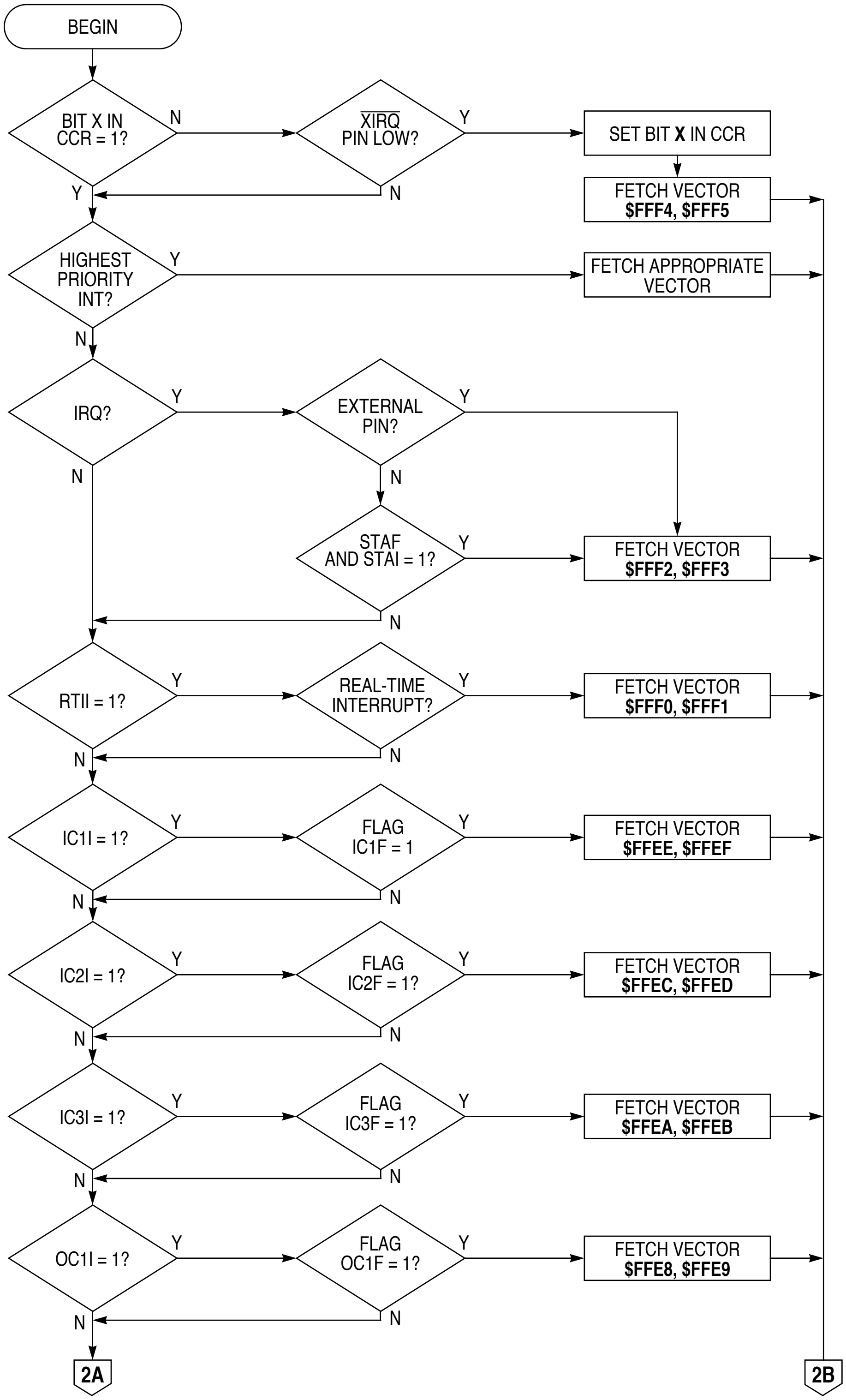
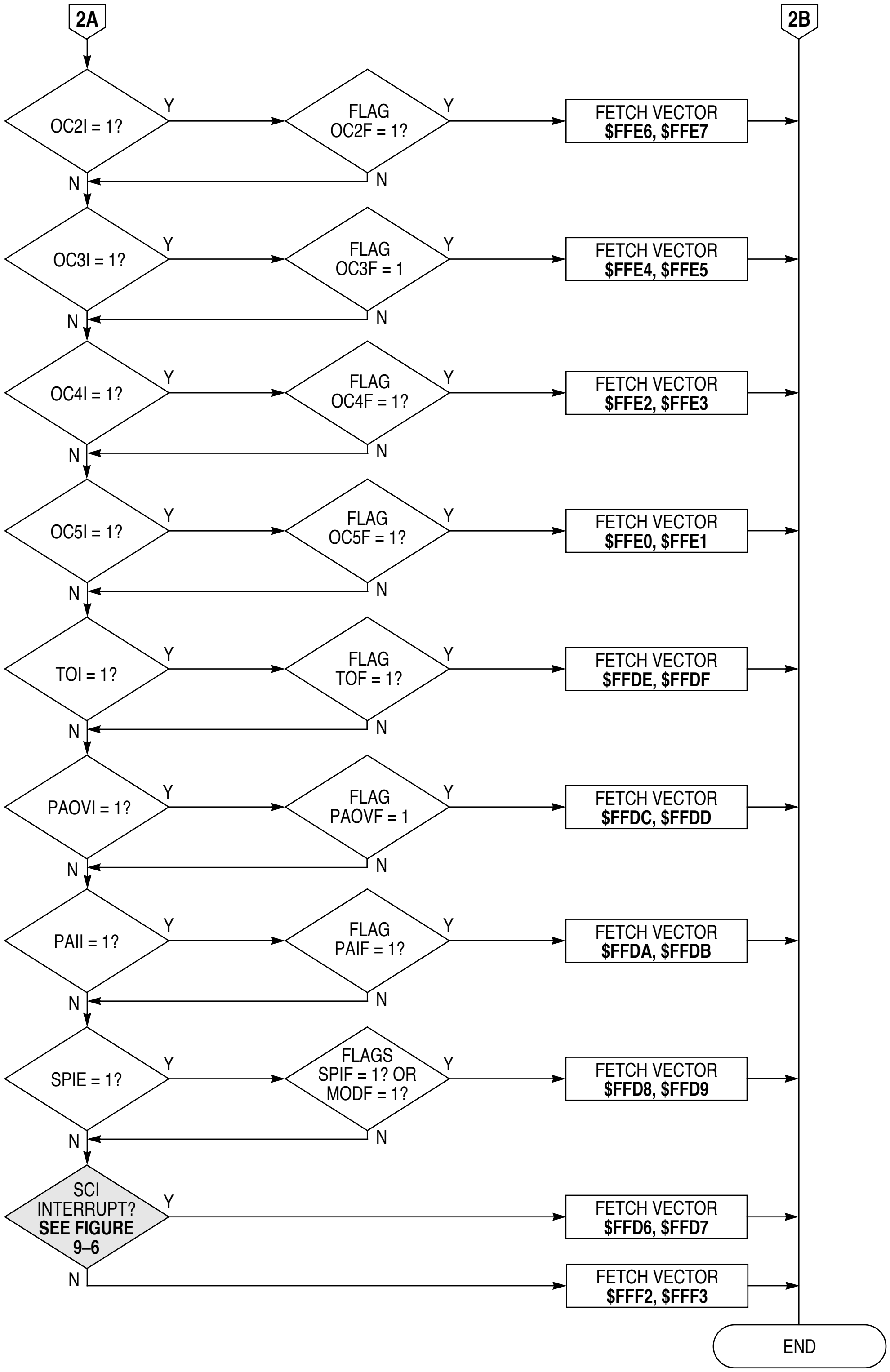


Figure 9-5 Interrupt Priority Resolution (Sheet 1 of 2)



9

Figure 9-5 Interrupt Priority Resolution (Sheet 2 of 2)

9

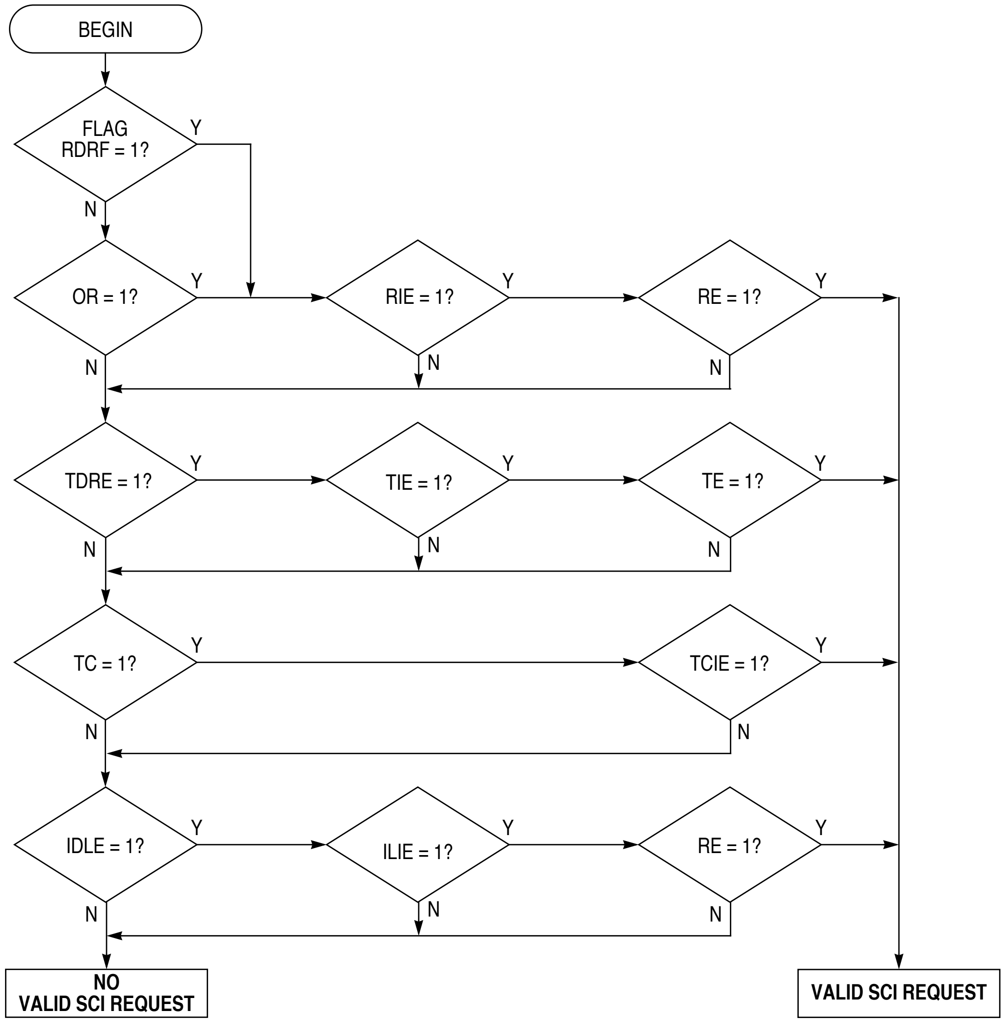


Figure 9-6 Interrupt Source Resolution Within SCI

Table 9-6 Highest Priority I Interrupt versus PSEL[3:0]

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 $\overline{\text{IRQ}}$)
0	1	1	0	$\overline{\text{IRQ}}$ (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 corresponds to “Reserved (default to $\overline{\text{IRQ}}$)” being the highest priority I-bit-related interrupt source.

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 $\overline{XIRQ}$ request. If the X bit is set, then processing will continue with the instruction immediately following the STOP instruction and no $\overline{XIRQ}$ interrupt service routine is requested. A 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			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	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	LO ADD
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.
LO ADD =Low 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 M68HC11 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 MC68HC11A8 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

10

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.

10

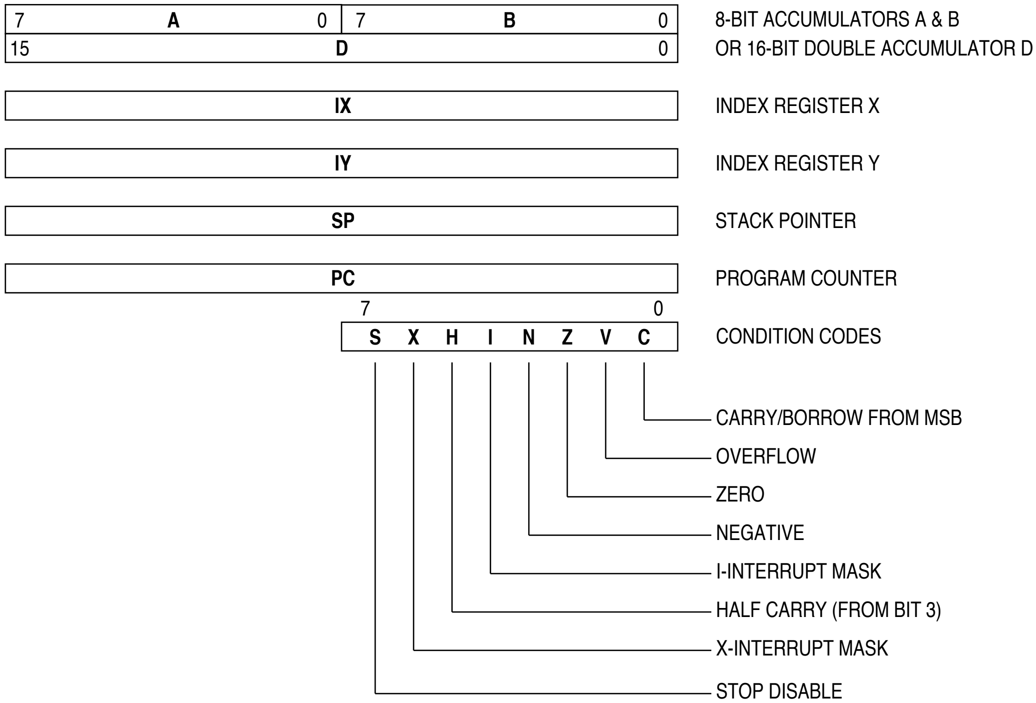


Figure 10-1 Programming Model

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 an 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 instruction), the SP is decrement; whereas, each time a byte is removed from the stack (a pull instruction) 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 an 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 MC68HC11A8, 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 Kbyte 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 MC68HC11A8, 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.

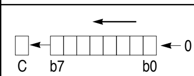
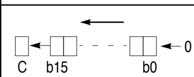
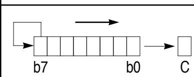
10.3 Instruction Set

The central processing unit (CPU) in the MC68HC11A8 is basically a proper extension of the MC6801 CPU. In addition to its ability to execute all M6800 and M6801 instructions, the MC68HC11A8 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 instructions, STOP and WAIT instructions, and bit manipulation instructions.

Table 10-1 shows all MC68HC11A8 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 **Table 10-8** provide 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 MC68HC11A8 Instructions, Addressing Modes, and Execution Times (Sheet 1 of 6)

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 A DIR A EXT A IND,X A IND,Y	89 99 B9 A9 18 A9	ii dd hh ll ff ff	2 2 3 2 3	2 3 4 4 5	3-1 4-1 5-2 6-2 7-2	- - ↕ - ↕ ↕ ↕ ↕
ADCB (opr)	Add with Carry to B	B + M + C → B	B IMM B DIR B EXT B IND,X B IND,Y	C9 D9 F9 E9 18 E9	ii dd hh ll ff ff	2 2 3 2 3	2 3 4 4 5	3-1 4-1 5-2 6-2 7-2	- - ↕ - ↕ ↕ ↕ ↕
ADDA (opr)	Add Memory to A	A + M → A	A IMM A DIR A EXT A IND,X A IND,Y	8B 9B BB AB 18 AB	ii dd hh ll ff ff	2 2 3 2 3	2 3 4 4 5	3-1 4-1 5-2 6-2 7-2	- - ↕ - ↕ ↕ ↕ ↕
ADDB (opr)	Add Memory to B	B + M → B	B IMM B DIR B EXT B IND,X B IND,Y	CB DB FB EB 18 EB	ii dd hh ll ff ff	2 2 3 2 3	2 3 4 4 5	3-1 4-1 5-2 6-2 7-2	- - ↕ - ↕ ↕ ↕ ↕
ADDD (opr)	Add 16-Bit to D	D + M:M + 1 → D	IMM DIR EXT IND,X IND,Y	C3 D3 F3 E3 18 E3	jj kk dd hh ll ff ff	3 2 3 2 3	4 5 6 6 7	3-3 4-7 5-10 6-10 7-8	- - - - ↕ ↕ ↕ ↕
ANDA (opr)	AND A with Memory	A•M → A	A IMM A DIR A EXT A IND,X A IND,Y	84 94 B4 A4 18 A4	ii dd hh ll ff ff	2 2 3 2 3	2 3 4 4 5	3-1 4-1 5-2 6-2 7-2	- - - - ↕ ↕ 0 -
ANDB (opr)	AND B with Memory	B•M → B	B IMM B DIR B EXT B IND,X B IND,Y	C4 D4 F4 E4 18 E4	ii dd hh ll ff ff	2 2 3 2 3	2 3 4 4 5	3-1 4-1 5-2 6-2 7-2	- - - - ↕ ↕ 0 -
ASL (opr) ASLA ASLB	Arithmetic Shift Left		EXT IND,X IND,Y A INH B INH	78 68 18 68 48 58	hh ll ff ff	3 2 3 1 1	6 6 7 2 2	5-8 6-3 7-3 2-1 2-1	- - - - ↕ ↕ ↕ ↕
ASLD	Arithmetic Shift Left Double		INH	05		1	3	2-2	- - - - ↕ ↕ ↕ ↕
ASR (opr) ASRA ASRB	Arithmetic Shift Right		EXT IND,X IND,Y A INH B INH	77 67 18 67 47 57	hh ll ff ff	3 2 3 1 1	6 6 7 2 2	5-8 6-3 7-3 2-1 2-1	- - - - ↕ ↕ ↕ ↕
BCC (rel)	Branch if Carry Clear	? C = 0	REL	24	rr	2	3	8-1	- - - - - - - -
BCLR (opr) (msk)	Clear Bit(s)	$M \bullet (\overline{mm}) \rightarrow M$	DIR IND,X IND,Y	15 1D 18 1D	dd mm ff mm ff mm	3 3 4	6 7 8	4-10 6-13 7-10	- - - - ↕ ↕ 0 -
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	- - - - - - - -
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	- - - - - - - -

*Cycle-by-cycle number provides a reference to Tables 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 MC68HC11A8 Instructions, Addressing Modes, and Execution Times (Sheet 2 of 6)

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
BITA (opr)	Bit(s) Test A with Memory	A•M	A IMM A DIR A EXT A IND,X A IND,Y	85 95 B5 A5 18 A5	ii dd hh ll ff ff	2 2 3 2 3	2 3 4 4 5	3-1 4-1 5-2 6-2 7-2	- - - - ↕ ↕ 0 -
BITB (opr)	Bit(s) Test B with Memory	B•M	B IMM B DIR B EXT B IND,X B IND,Y	C5 D5 F5 E5 18 E5	ii dd hh ll ff ff	2 2 3 2 3	2 3 4 4 5	3-1 4-1 5-2 6-2 7-2	- - - - ↕ ↕ 0 -
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 IND,X IND,Y	13 1F 18 1F	dd mm rr ff mm rr ff mm rr	4 4 5	6 7 8	4-11 6-14 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 IND,X IND,Y	12 1E 18 1E	dd mm rr ff mm rr ff mm rr	4 4 5	6 7 8	4-11 6-14 7- 11	- - - - - - - -
BSET(opr) (msk)	Set Bit(s)	M + mm → M	DIR IND,X IND,Y	14 1C 18 1C	dd mm ff mm ff mm	3 3 4	6 7 8	4-10 6-13 7-10	- - - - ↕ ↕ 0 -
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 IND,X IND,Y	7F 6F 18 6F	hh ll ff ff	3 2 3	6 6 7	5-8 6-3 7-3	- - - - 0 1 0 0
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 A DIR A EXT A IND,X A IND,Y	81 91 B1 A1 18 A1	ii dd hh ll ff ff	2 2 3 2 3	2 3 4 4 5	3-1 4-1 5-2 6-2 7-2	- - - - ↕ ↕ ↕ ↕
CMPB (opr)	Compare B to Memory	B – M	B IMM B DIR B EXT B IND,X B IND,Y	C1 D1 F1 E1 18 E1	ii dd hh ll ff ff	2 2 3 2 3	2 3 4 4 5	3-1 4-1 5-2 6-2 7-2	- - - - ↕ ↕ ↕ ↕
COM (opr)	1's Complement Memory Byte	$FF – M → M	EXT IND,X IND,Y	73 63 18 63	hh ll ff ff	3 2 3	6 6 7	5-8 6-3 7-3	- - - - ↕ ↕ 0 1
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 DIR EXT IND,X IND,Y	1A 83 1A 93 1A B3 1A A3 CD A3	jj kk dd hh ll ff ff	4 3 4 3 3	5 6 7 7 7	3-5 4-9 5-11 6-11 7-8	- - - - ↕ ↕ ↕ ↕

*Cycle-by-cycle number provides a reference to Tables 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.

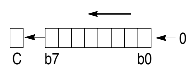
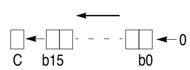
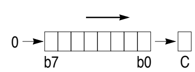
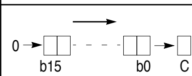
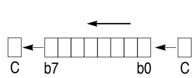
Table 10-1 MC68HC11A8 Instructions, Addressing Modes, and Execution Times (Sheet 3 of 6)

10

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
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	88	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	- - - - - - - -
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	

*Cycle-by-cycle number provides a reference to Tables 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.

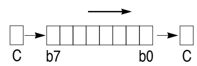
Table 10-1 MC68HC11A8 Instructions, Addressing Modes, and Execution Times (Sheet 4 of 6)

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
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	3-7	
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	- - - - ↕ ↕ ↕ ↕
			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	AxB → D	INH	3D		1	10	2-13	- - - - - - - ↕
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			A INH	49		1	2	2-1	
ROLB			B INH	59		1	2	2-1	

*Cycle-by-cycle number provides a reference to Tables 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.

10

Table 10-1 MC68HC11A8 Instructions, Addressing Modes, and Execution Times (Sheet 5 of 6)

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
ROR (opr) RORA RORB	Rotate Right		EXT IND,X IND,Y A INH B INH	76 66 18 66 46 56	hh ll ff ff	3 2 3 1 1	6 6 7 2 2	5-8 6-3 7-3 2-1 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 A DIR A EXT A IND,X A IND,Y	82 92 B2 A2 18 A2	ii dd hh ll ff ff	2 2 3 2 3	2 3 4 4 5	3-1 4-1 5-2 6-2 7-2	- - - - ↕ ↕ ↕ ↕
SBCB (opr)	Subtract with Carry from B	B – M – C → B	B IMM B DIR B EXT B IND,X B IND,Y	C2 D2 F2 E2 18 E2	ii dd hh ll ff ff	2 2 3 2 3	2 3 4 4 5	3-1 4-1 5-2 6-2 7-2	- - - - ↕ ↕ ↕ ↕
SEC	Set Carry	1 → C	INH	OD		1	2	2-1	- - - - - - - 1
SEI	Set Interrupt Mask	1 → I	INH	OF		1	2	2-1	- - - 1 - - - -
SEV	Set Overflow Flag	1 → V	INH	OB		1	2	2-1	- - - - - - 1 -
STAA (opr)	Store Accumulator A	A → M	A DIR A EXT A IND,X A IND,Y	97 B7 A7 18 A7	dd hh ll ff ff	2 3 2 3	3 4 4 5	4-2 5-3 6-5 7-5	- - - - ↕ ↕ 0 -
STAB (opr)	Store Accumulator B	B → M	B DIR B EXT B IND,X B IND,Y	D7 F7 E7 18 E7	dd hh ll ff ff	2 3 2 3	3 4 4 5	4-2 5-3 6-5 7-5	- - - - ↕ ↕ 0 -
STD (opr)	Store Accumulator D	A → M, B → M + 1	DIR EXT IND,X IND,Y	DD FD ED 18 ED	dd hh ll ff ff	2 3 2 3	4 5 5 6	4-4 5-5 6-8 7-7	- - - - ↕ ↕ 0 -
STOP	Stop Internal Clocks		INH	CF		1	2	2-1	- - - - - - - -
STS (opr)	Store Stack Pointer	SP → M:M + 1	DIR EXT IND,X IND,Y	9F BF AF 18 AF	dd hh ll ff ff	2 3 2 3	4 5 5 6	4-4 5-5 6-8 7-7	- - - - ↕ ↕ 0 -
STX (opr)	Store Index Register X	IX → M:M + 1	DIR EXT IND,X IND,Y	DF FF EF CD EF	dd hh ll ff ff	2 3 2 3	4 5 5 6	4-4 5-5 6-8 7-7	- - - - ↕ ↕ 0 -
STY (opr)	Store Index Register Y	IY → M:M + 1	DIR EXT IND,X IND,Y	18 DF 18 FF 1A EF 18 EF	dd hh ll ff ff	3 4 3 3	5 6 6 6	4-6 5-7 6-9 7-7	- - - - ↕ ↕ 0 -
SUBA (opr)	Subtract Memory from A	A – M → A	A IMM A DIR A EXT A IND,X A IND,Y	80 90 B0 A0 18 A0	ii dd hh ll ff ff	2 2 3 2 3	2 3 4 4 5	3-1 4-1 5-2 6-2 7-2	- - - - ↕ ↕ ↕ ↕
SUBB (opr)	Subtract Memory from B	B – M → B	B IMM B DIR B EXT B IND,X B IND,Y	C0 D0 F0 E0 18 E0	ii dd hh ll ff ff	2 2 3 2 3	2 3 4 4 5	3-1 4-1 5-2 6-2 7-2	- - - - ↕ ↕ ↕ ↕
SUBD (opr)	Subtract Memory from D	D – M:M + 1→ D	IMM DIR EXT IND,X IND,Y	83 93 B3 A3 18 A3	jj kk dd hh ll ff ff	3 2 3 2 3	4 5 6 6 7	3-3 4-7 5-10 6-10 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 -

*Cycle-by-cycle number provides a reference to Tables 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.

10

Table 10-1 MC68HC11A8 Instructions, Addressing Modes, and Execution Times (Sheet 6 of 6)

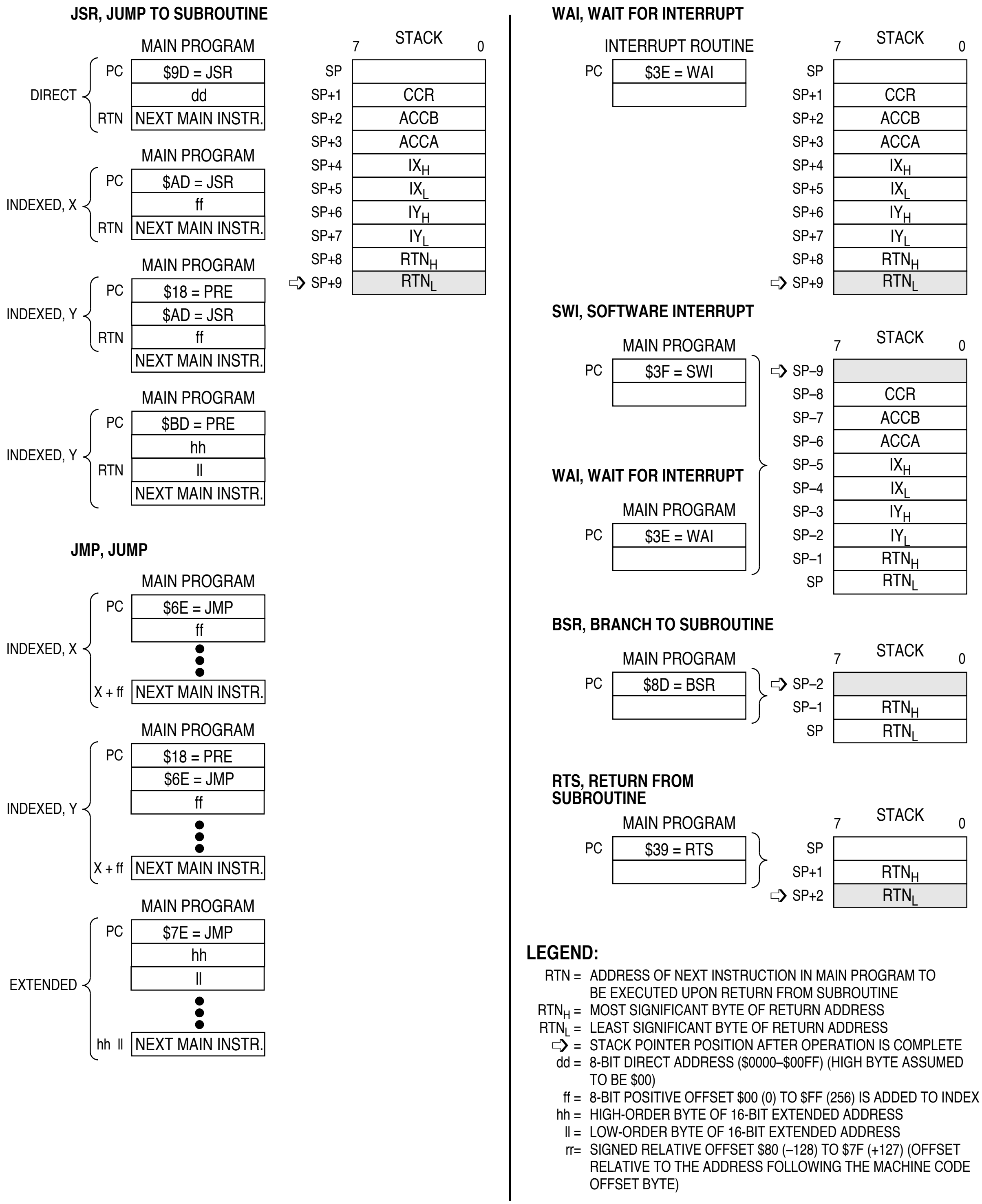
Source Form(s)	Operation	Boolean Expression	Addressing Mode for Operand	Machine Coding (Hexadecimal) Opcode	Operand(s)	Bytes	Cycle	Cycle by Cycle*	Condition Codes S X H I N Z V C
TEST	TEST (Only in Test Modes)	Address Bus Counts	INH	00		1	**	2-20	- - - - - - - -
TPA	Transfer CC Register to A	CCR → A	INH	07		1	2	2-1	- - - - - - - -
TST (opr)	Test for Zero or Minus	M – 0	EXT IND,X IND,Y	7D 6D 18 6D	hh ll ff ff	3 2 3	6 6 7	5-9 6-4 7-4	- - - - ↕ ↕ 0 0
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	- - - - - - - -
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 Tables 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 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 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)



10

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
2-9	PULA, PULB	4	1 2 3 4	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer + 1	1 1 1 1	Opcode Irrelevant Data Irrelevant Data Operand Data from Stack
2-10	PULX	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Stack Pointer Stack Pointer + 1 Stack Pointer + 2	1 1 1 1 1	Opcode ($38) Irrelevant Data Irrelevant Data IXH (High Byte) from Stack IXL (Low Byte) from Stack

* The reference number is given to provide a cross-reference to Table 10-1.

10

Table 10-2 Cycle-by-Cycle Operation — Inherent Mode (Sheet 2 of 4)

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
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	IYH (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 Address	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
			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 Address	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)

* 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
2-16	WAI	14 + n	1	Opcode Address	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
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)

* 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
2-20	TEST	Infinite	1 2 3 4 5 – n	Opcode Address Opcode Address + 1 Opcode Address + 1 Opcode Address + 2 Previous Address + 1	1 1 1 1 1	Opcode ($00) Irrelevant Data Irrelevant Data Irrelevant Data 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 2 3	Opcode Address Opcode Address + 1 Operand Address	1 1 1	Opcode Operand Address (Low Byte) (High Byte Assumed to be $00) Operand Data
4-2	STAA, STAB	3	1 2 3	Opcode Address Opcode Address + 1 Operand Address	1 1 0	Opcode Operand Address (Low Byte) (High Byte Assumed to be $00) Data from Accumulator

*The reference number is given to provide a cross-refrerence to Table 10-1.

10

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
4-3	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) ($18)
			2	Opcode Address + 1	1	Opcode (Second Byte) ($DF)
			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
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-refrerence 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 2 3	Opcode Address Opcode Address + 1 Opcode Address + 2	1 1 1	Opcode ($7E) Jump Address (High Byte) 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 2 3 4	Opcode Address Opcode Address + 1 Opcode Address + 2 Operand Address	1 1 1 1	Opcode Operand Address (High Byte) Operand Address (Low Byte) Operand Data
5-3	STAA, STAB	4	1 2 3 4	Opcode Address Opcode Address + 1 Opcode Address + 2 Operand Address	1 1 1 0	Opcode Operand Address (High Byte) Operand Address (Low Byte) Accumulator Data
5-4	LDD, LDS, LDX	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Opcode Address + 2 Operand Address Operand Address + 1	1 1 1 1 1	Opcode Operand Address (High Byte) Operand Address (Low Byte) Operand Data (High Byte) Operand Data (Low Byte)
5-5	STD, STS, STX	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Opcode Address + 2 Operand Address Operand Address + 1	1 1 1 0 0	Opcode Operand Address (High Byte) Operand Address (Low Byte) Register Data (High Byte) Register Data (Low Byte)
5-6	LDY	6	1 2 3 4 5 6	Opcode Address Opcode Address + 1 Opcode Address + 2 Opcode Address + 3 Operand Address Operand Address + 1	1 1 1 1 1 1	Opcode (Page Select Byte) ($18) Opcode (Second Byte) ($FE) Operand Address (High Byte) Operand Address (Low Byte) Operand Data (High Byte) Operand Data (Low Byte)
5-7	STY	6	1 2 3 4 5 6	Opcode Address Opcode Address + 1 Opcode Address + 2 Opcode Address + 3 Operand Address Operand Address + 1	1 1 1 1 0 0	Opcode (Page Select Byte) ($18) Opcode (Second Byte) ($FF) Operand Address (High Byte) Operand Address (Low Byte) Register Data (High Byte) Register Data (Low Byte)
5-8	ASL, ASR, CLR, COM, DEC, INC, LSL, LSR, NEG, ROL, ROR	6	1 2 3 4 5 6	Opcode Address Opcode Address + 1 Opcode Address + 2 Operand Address $FFFF Operand Address	1 1 1 1 1 0	Opcode Operand Address (High Byte) Operand Address (Low Byte) Original Operand Data Irrelevant Data Result Operand Data
5-9	TST	6	1 2 3 4 5 6	Opcode Address Opcode Address + 1 Opcode Address + 2 Operand Address $FFFF $FFFF	1 1 1 1 1 1	Opcode ($7D) Operand Address (High Byte) Operand Address (Low Byte) Original Operand Data Irrelevant Data Irrelevant Data
5-10	ADDD, CPX, SUBD	6	1 2 3 4 5 6	Opcode Address Opcode Address + 1 Opcode Address + 2 Operand Address Operand Address + 1 $FFFF	1 1 1 1 1 1	Opcode Operand Address (High Byte) Operand Address (Low 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 10-5 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
5-11	CPD, CPY	7	1 2 3 4 5 6 7	Opcode Address Opcode Address + 1 Opcode Address + 2 Opcode Address + 3 Operand Address Operand Address + 1 $FFFF	1 1 1 1 1 1 1	Opcode (Page Select Byte) Opcode (Second Byte) Operand Address (High Byte) Operand Address (Low Byte) Operand Data (High Byte) Operand Data (Low Byte) Irrelevant Data
5-12	JSR	6	1 2 3 4 5 6	Opcode Address Opcode Address + 1 Opcode Address + 2 Subroutine Address Stack Pointer Stack Pointer – 1	1 1 1 1 0 0	Opcode ($BD) Subroutine Address (High Byte) Subroutine Address (Low Byte) First Opcode in Subroutine Return Address (Low Byte) 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)

10

Reference Number*	Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
6-1	JMP	3	1 2 3	Opcode Address Opcode Address + 1 $FFFF	1 1 1	Opcode ($6E) Index Offset 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 2 3 4	Opcode Address Opcode Address + 1 $FFFF (IX) + Offset	1 1 1 1	Opcode Index Offset Irrelevant Data Operand Data
6-3	ASL, ASR, CLR, COM, DEC, INC, LSL, LSR, NEG, ROL, ROR	6	1 2 3 4 5 6	Opcode Address Opcode Address + 1 $FFFF (IX) + Offset $FFFF (IX) + Offset	1 1 1 1 1 0	Opcode Index Offset Irrelevant Data Original Operand Data Irrelevant Data Result Operand Data
6-4	TST	6	1 2 3 4 5 6	Opcode Address Opcode Address + 1 $FFFF (IX) + Offset $FFFF $FFFF	1 1 1 1 1 1	Opcode ($6D) Index Offset Irrelevant Data Original Operand Data Irrelevant Data Irrelevant Data
6-5	STAA, STAB	4	1 2 3 4	Opcode Address Opcode Address + 1 $FFFF (IX) + Offset	1 1 1 0	Opcode Index Offset Irrelevant Data Accumulator Data
6-6	LDD, LDS, LDX	5	1 2 3 4 5	Opcode Address Opcode Address + 1 $FFFF (IX) + Offset (IX) + Offset + 1	1 1 1 1 1	Opcode Index Offset Irrelevant Data Operand Data (High Byte) Operand Data (Low Byte)

*The reference number is given to provide a cross-reference to Table 10-1.

10

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
6-7	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
			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	Operand Data (High Byte)
			5	(IX) + Offset + 1	1	Operand 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
			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

* 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
7-9	JSR	7	1 2 3 4 5 6 7	Opcode Address Opcode Address + 1 Opcode Address + 2 $FFFF (IY) + Offset Stack Pointer Stack Pointer – 1	1 1 1 1 1 0 0	Opcode (Page Select Byte) ($18) Opcode (Second Byte) ($AD) Index Offset Irrelevant Data First Opcode in Subroutine Return Address (Low Byte) Return Address (High Byte)
7-10	BCLR, BSET	8	1 2 3 4 5 6 7 8	Opcode Address Opcode Address + 1 Opcode Address + 2 $FFFF (IY) + Offset Opcode Address + 3 $FFFF (IY) + Offset	1 1 1 1 1 1 1 0	Opcode (Page Select Byte) ($18) Opcode (Second Byte) Index Offset Irrelevant Data Original Operand Data Mask Byte Irrelevant Data Result Operand Data
7-11	BRCLR, BRSET	8	1 2 3 4 5 6 7 8	Opcode Address Opcode Address + 1 Opcode Address + 2 $FFFF (IY) + Offset Opcode Address + 3 Opcode Address + 4 $FFFF	1 1 1 1 1 1 1 1	Opcode (Page Select Byte) ($18) Opcode (Second Byte) Index Offset Irrelevant Data Original Operand Data Mask Byte Branch Offset Irrelevant Data

* The reference number is given to provide a cross-reference to Table 10-1.

10

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 2 3	Opcode Address Opcode Address + 1 $FFFF	1 1 1	Opcode Branch Offset Irrelevant Data
8-2	BSR	6	1 2 3 4 5 6	Opcode Address Opcode Address + 1 $FFFF Subroutine Address Stack Pointer Stack Pointer – 1	1 1 1 1 0 0	Opcode ($8D) Branch Offset Irrelevant Data Opcode of Next Instruction Return Address (Low Byte) Return Address (High Byte)

*The reference number is given to provide a cross-reference to Table 10-1.

A ELECTRICAL CHARACTERISTICS

Table A-1 Maximum Rating

Rating	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 MC68HC11A8 MC68HC11A8C MC68HC11A8V MC68HC11A8M MC68L11A8	T_A	T_L to T_H 0 to 70 – 40 to 85 – 40 to 105 – 40 to 125 – 20 to 70	°C
Storage Temperature Range	T_{stg}	– 55 to 150	°C
Current Drain per Pin* Excluding V_{DD}, V_{SS}, V_{RH}, and V_{RL}	I_D	25	mA

*One pin at a time, observing maximum power dissipation limits.

A

Internal circuitry protects the inputs against damage caused by high static voltages or electric fields; however, normal precautions are necessary to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. Extended operation at the maximum ratings can adversely affect device reliability. Tying unused inputs to an appropriate logic voltage level (either GND or V_{DD}) enhances reliability of operation.

Table A-2 Thermal Characteristics

Characteristic	Symbol	Value	Unit
Average Junction Temperature	T_J	$T_A + (P_D \times \Theta_{JA})$	°C
Ambient Temperature	T_A	User-determined	°C
Package Thermal Resistance (Junction-to-Ambient) 52-Pin Plastic Quad Pack (PLCC) 48-Pin Plastic Dual In-Line Package (DIP)	Θ_{JA}	 50 40	°C/W
Total Power Dissipation	P_D	$P_{INT} + P_{I/O}$ $K \div (T_J + 273°C)$ (Note 1)	W
Device Internal Power Dissipation	P_{INT}	$I_{DD} \times V_{DD}$	W
I/O Pin Power Dissipation	$P_{I/O}$ (Note 2)	User-determined	W
A Constant	K	$P_D \times (T_A + 273°C) +$ $\Theta_{JA} \times P_D2$ (Note 3)	W · °C

NOTES:
1. This is an approximate value, neglecting $P_{I/O}$.
2. For most applications $P_{I/O} \ll P_{INT}$ and can be neglected.
3. K is a constant pertaining to the device. Solve for K with a known T_A and a measured P_D (at equilibrium). Use this value of K to solve for P_D and T_J iteratively for any value of T_A.

Table A-3 DC Electrical Characteristics

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

Characteristics	Symbol	Min	Max	Unit
Output Voltage (Note 1) All Outputs except XTAL All Outputs Except XTAL, $\overline{RESET}$, and MODA I_{Load} =± 10.0 µA	V_{OL} V_{OH}	— V_{DD} – 01	0.1 —	V
Output High Voltage (Note 1) All Outputs Except XTAL, $\overline{RESET}$, and MODA I_{Load} = – 0.8 mA, V_{DD} = 4.5 V	V_{OH}	V_{DD} – 0.8	—	V
Output Low Voltage All Outputs Except XTAL I_{Load} = 1.6 mA	V_{OL}	—	0.4	V
Input High Voltage All Inputs Except $\overline{RESET}$ $\overline{RESET}$	V_{IH}	0.7 x V_{DD} 0.8 x V_{DD}	V_{DD} + 0.3 V_{DD} + 0.3	V
Input Low Voltage All Inputs	V_{IL}	V_{SS} – 0.3	0.2 x V_{DD}	V
I/O Ports, Three-State Leakage PA7, PC0-PC7, PD0-PD5, AS/STRA, MODA/$\overline{LIR}$, $\overline{RESET}$ V_{in} = V_{IH} or V_{IL}	I_{OZ}	—	±10	µA
Input Leakage Current (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 Power down	V_{SB}	4.0	V_{DD}	V
RAM Standby Current Power down	I_{SB}	—	10	µA
Total Supply Current (Note 3) RUN: Single-Chip Mode Expanded Multiplexed Mode WAIT: All Peripheral Functions Shut Down Single-Chip Mode Expanded Multiplexed Mode STOP: No Clocks, Single-Chip Mode	 I_{DD} dc – 2 MHz 3 MHz dc – 2 MHz 3 MHz W_{IDD} dc – 2 MHz 3 MHz dc – 2 MHz 3 MHz S_{IDD} dc – 2 MHz 3 MHz	 — — — — — — — — — —	 15 27 27 35 6 15 10 20 50 150	 mA mA µA
Input Capacitance PA0-PA2, 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 Single-Chip Mode Expanded Multiplexed Mode	P_D 2 MHz 3 MHz 	 — — — —	 85 150 150 195	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 wired-OR mode.
2. Refer to A/D specification for leakage current for port E.
3. EXTAL is driven with a square wave, and
   t_{cyc} = 500 ns for 2 MHz rating;
   t_{cyc} = 333 ns for 3 MHz rating.
   $V_{IL} \leq 0.2$ V
   $V_{IH} \geq V_{DD} - 0.2$ V
   No dc loads.

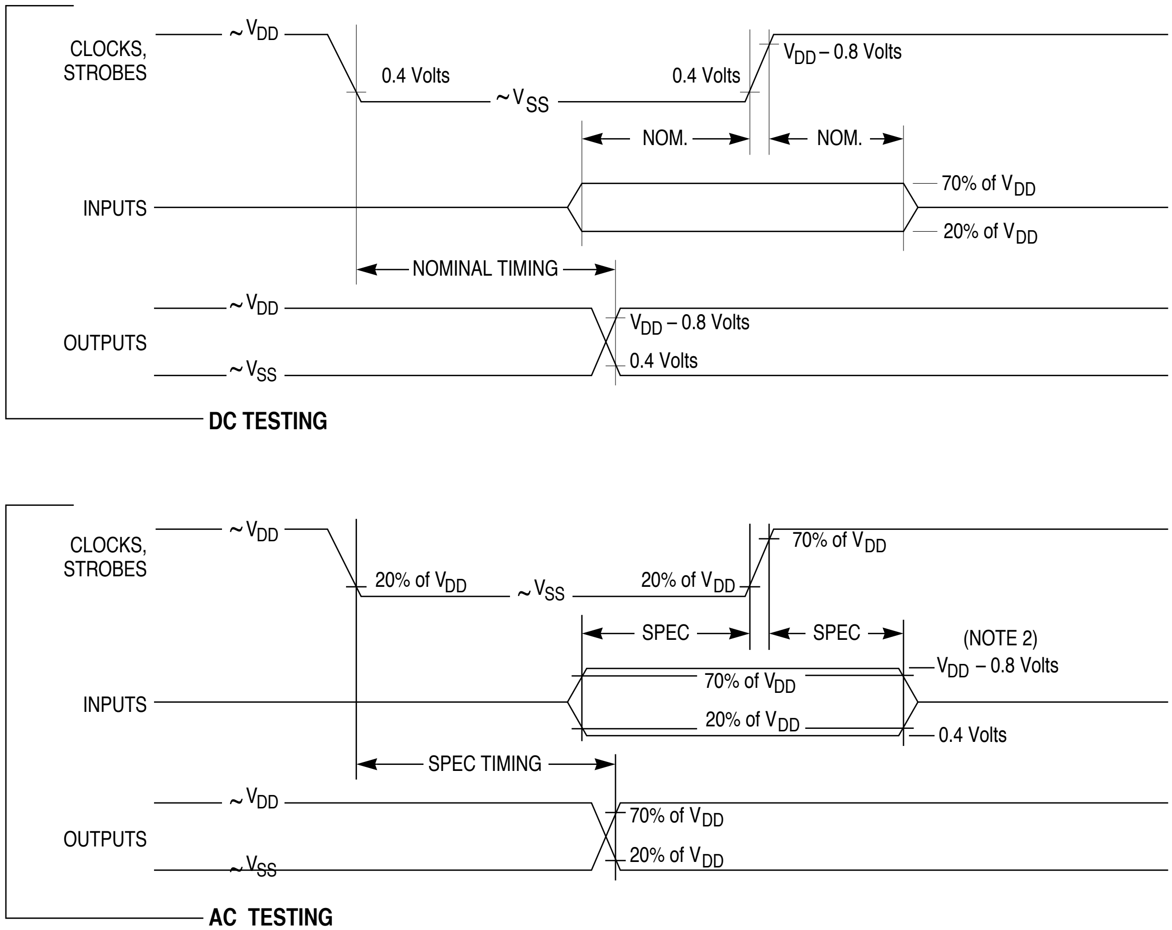
Table A-3a DC Electrical Characteristics (MC68L11A8)

V_{DD} = 3.0 Vdc to 5.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted

Characteristics	Symbol	Min	Max	Unit
Output Voltage (Note 1) All Outputs except XTAL All Outputs Except XTAL, $\overline{\text{RESET}}$, and MODA I_{Load} =± 10.0 µA	V_{OL} V_{OH}	— V_{DD} – 01	0.1 —	V
Output High Voltage (Note 1) All Outputs Except XTAL, $\overline{\text{RESET}}$, and MODA I_{Load} = – 0.8 mA, V_{DD} = 4.5 V	V_{OH}	V_{DD} – 0.8	3	V
Output Low Voltage All Outputs Except XTAL I_{Load} = 1.6 mA	V_{OL}	—	0.4	V
Input High Voltage All Inputs Except $\overline{\text{RESET}}$ $\overline{\text{RESET}}$	V_{IH}	0.7 x V_{DD} 0.8 x V_{DD}	V_{DD} + 0.3 V_{DD} + 0.3	V
Input Low VoltageAll Inputs	V_{IL}	V_{SS} – 0.3	0.2 x V_{DD}	V
I/O Ports, Three-State Leakage PA7, PC0-PC7, PD0-PD5, AS/STRA, MODA/$\overline{\text{LIR}}$, $\overline{\text{RESET}}$ $V_{in} = V_{IH}$ or V_{IL}	I_{OZ}	—	±10	µA
Input Leakage Current (Note 2) $V_{in} = V_{DD}$ or V_{SS} PA0-PA2, $\overline{\text{IRQ}}$, $\overline{\text{XIRQ}}$ $V_{in} = V_{DD}$ or V_{SS} MODB/V_{STBY}	I_{in}	 — —	 ±1 ±10	µA
RAM Standby Voltage Power down	V_{SB}	2.0	V_{DD}	V
RAM Standby Current Power down	I_{SB}	—	10	µA
Total Supply Current (Note 3) RUN: Single-Chip Mode Expanded Multiplexed Mode WAIT: All Peripheral Functions Shut Down Single-Chip Mode Expanded Multiplexed Mode STOP: No Clocks, Single-Chip Mode 	 I_{DD} dc – 1 MHz 2MHz dc – 1 MHz 2 MHz W_{IDD} dc – 1 MHz 2 MHz dc – 1 MHz 2 MHz S_{IDD} dc – 1 MHz 2 MHz	 — — — — — — — — — —	 4 8 7 14 3 6 2.5 5 25 25	 mA mA µA
Input Capacitance PA0-PA2, PE0-PE7, $\overline{\text{IRQ}}$, $\overline{\text{XIRQ}}$, EXTAL PA7, PC0-PC7, PD0-PD5, AS/STRA, MODA/$\overline{\text{LIR}}$, $\overline{\text{RESET}}$	C_{in}	— —	8 12	pF
Power Dissipation Single-Chip Mode Expanded Multiplexed Mode Single-Chip Mode Expanded Multiplexed Mode	P_D 1 MHz 2 MHz 	 — — — —	 12 21 24 42	mW

NOTES:
1. V_{OH} specification for $\overline{\text{RESET}}$ and MODA is not applicable because they are open-drain pins. V_{OH} specification not applicable to ports C and D in wired-OR mode.
2. Refer to A/D specification for leakage current for port E.
3. EXTAL is driven with a square wave, and
   t_{cyc} = 1000 ns for 1MHz rating;
   t_{cyc} = 500 ns for 2 MHz rating.
   $V_{IL} \leq 0.2$ V
   $V_{IH} \geq V_{DD} - 0.2$ V
   No dc loads.

A



NOTES:
1. Full test loads are applied during all DC electrical tests and AC timing measurements.
2. During AC timing measurements, inputs are driven to 0.4 volts and VDD – 0.8 volts while timing measurements are taken at the 20% and 70% of V_{DD} points.

TEST METHODS

Figure A-1 Test Methods

A

Table A-4 Control Timing

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

Characteristic	Symbol	1.0 MHz		2.0 MHz		3.0 MHz		Unit
		Min	Max	Min	Max	Min	Max	
Frequency of Operation	f_o	dc	1.0	dc	2.0	dc	3.0	MHz
E-Clock Period	t_{cyc}	1000	—	500	—	333	—	ns
Crystal Frequency	f_{XTAL}	—	4.0	—	8.0	—	12.0	MHz
External Oscillator Frequency	4 f_o	dc	4.0	dc	8.0	dc	12.0	MHz
Processor Control Setup Time $t_{PCSU} = 1/4\ t_{cyc} + 50$ ns	t_{PCSU}	300	—	175	—	133	—	ns
Reset Input Pulse Width (Note 1) (To Guarantee External Reset Vector) (Minimum Input Time; Can Be Preempted by Internal Reset)	PW_{RSTL}	8 1	— —	8 1	— —	8 1	— —	t_{cyc}
Mode Programming Setup Time	t_{MPS}	2	—	2	—	2	—	t_{cyc}
Mode Programming Hold Time	t_{MPH}	10	—	10	—	10	—	ns
Interrupt Pulse Width, $\overline{IRQ}$ Edge-Sensitive Mode $PW_{IRQ} = t_{cyc} + 20$ ns	PW_{IRQ}	1020	—	520	—	353	—	ns
Wait Recovery Start-up Time	t_{WRS}	—	4	—	4	—	4	t_{cyc}
Timer Pulse Width Input Capture, Pulse Accumulator Input $PW_{TIM} = t_{cyc} + 20$ ns	PW_{TIM}	1020	—	520	—	353	—	ns

NOTES:
1. $\overline{RESET}$ is recognized during the first clock cycle it is held low. Internal circuitry then drives the pin low for four clock cycles, releases the pin, and samples the pin level two cycles later to determine the source of the interrupt. Refer to **9 RESETS, INTERRUPTS, AND LOW POWER MODES** for further detail.
2. All timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted.

A

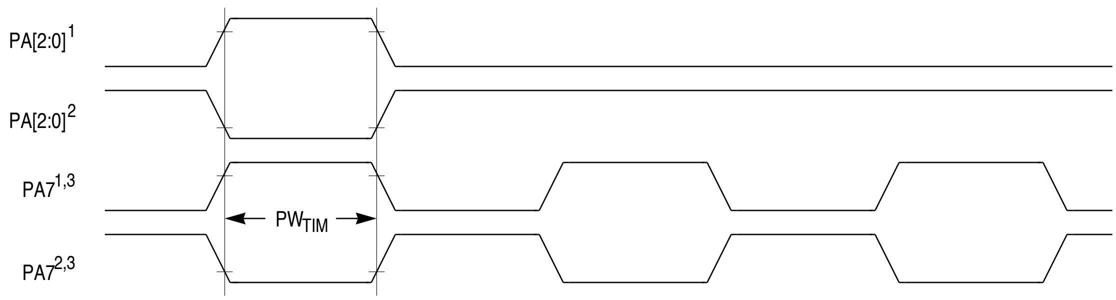
Table A-4a Control Timing (MC68L11A8)

V_{DD} = 3.0 Vdc to 5.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H

Characteristic		Symbol	1.0 MHz		2.0 MHz		Unit
			Min	Max	Min	Max	
Frequency of Operation		f_o	dc	1.0	dc	2.0	MHz
E-Clock Period		t_{cyc}	1000	—	500	—	ns
Crystal Frequency		f_{XTAL}	—	4.0	—	8.0	MHz
External Oscillator Frequency		4 f_o	dc	4.0	dc	8.0	MHz
Processor Control Setup Time	t_{PCSU} = 1/4 t_{cyc} + 50 ns	t_{PCSU}	325	—	200	—	ns
Reset Input Pulse Width (Note 1)	(To Guarantee External Reset Vector)	PW_{RSTL}	8	—	8	—	t_{cyc}
	(Minimum Input Time; Can Be Preempted by Internal Reset)		1	—	1	—	
Mode Programming Setup Time		t_{MPS}	2	—	2	—	t_{cyc}
Mode Programming Hold Time		t_{MPH}	10	—	10	—	ns
Interrupt Pulse Width, $\overline{IRQ}$ Edge-Sensitive Mode	$PW_{IRQ} = t_{cyc}$ + 20 ns	PW_{IRQ}	1020	—	520	—	ns
Wait Recovery Start-up Time		t_{WRS}	—	4	—	4	t_{cyc}
Timer Pulse Width Input Capture, Pulse Accumulator Input	$PW_{TIM} = t_{cyc}$ + 20 ns	PW_{TIM}	1020	—	520	—	ns

NOTES:
1. $\overline{RESET}$ is recognized during the first clock cycle it is held low. Internal circuitry then drives the pin low for four clock cycles, releases the pin, and samples the pin level two cycles later to determine the source of the interrupt. Refer to **9 RESETS, INTERRUPTS, AND LOW POWER MODES** for further detail.
2. All timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted.

A



NOTES:
1. Rising edge sensitive input
2. Falling edge sensitive input
3. Maximum pulse accumulator clocking rate is E-clock frequency divided by 2.

TIMER INPUTS TIM

Figure A-2 Timer Inputs

A

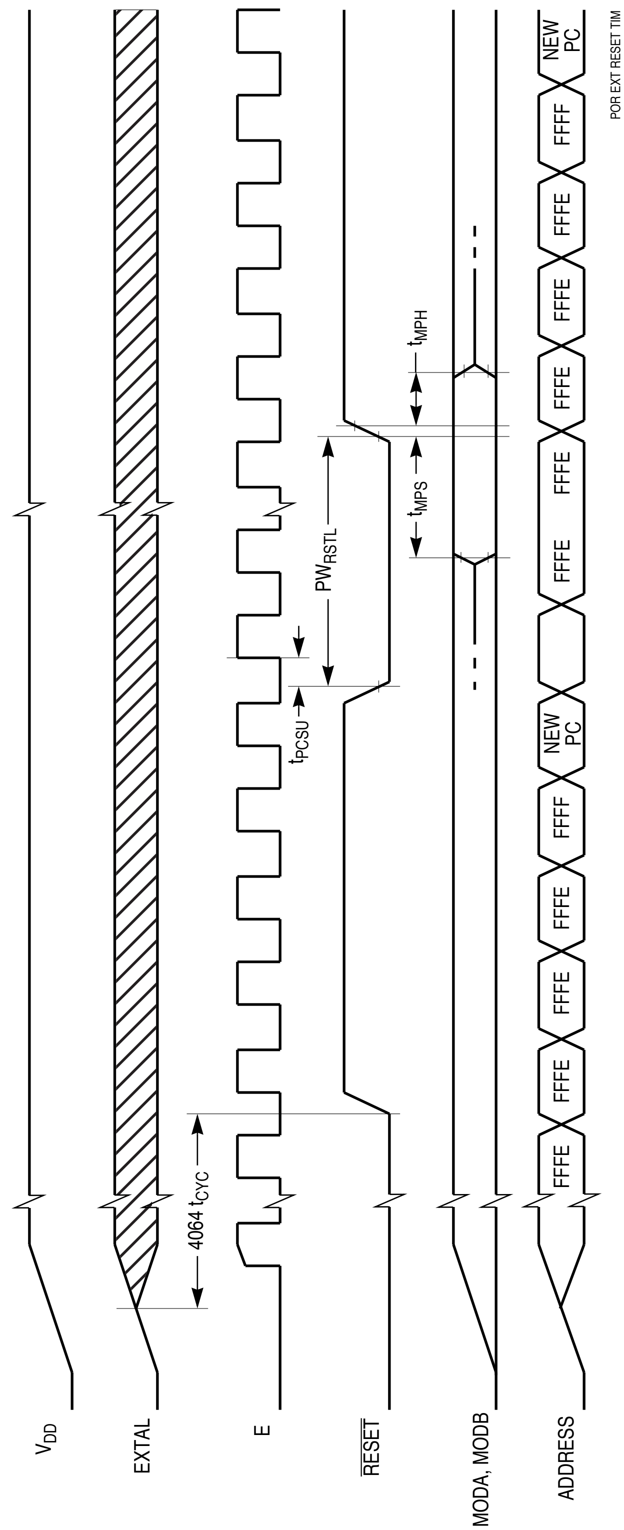
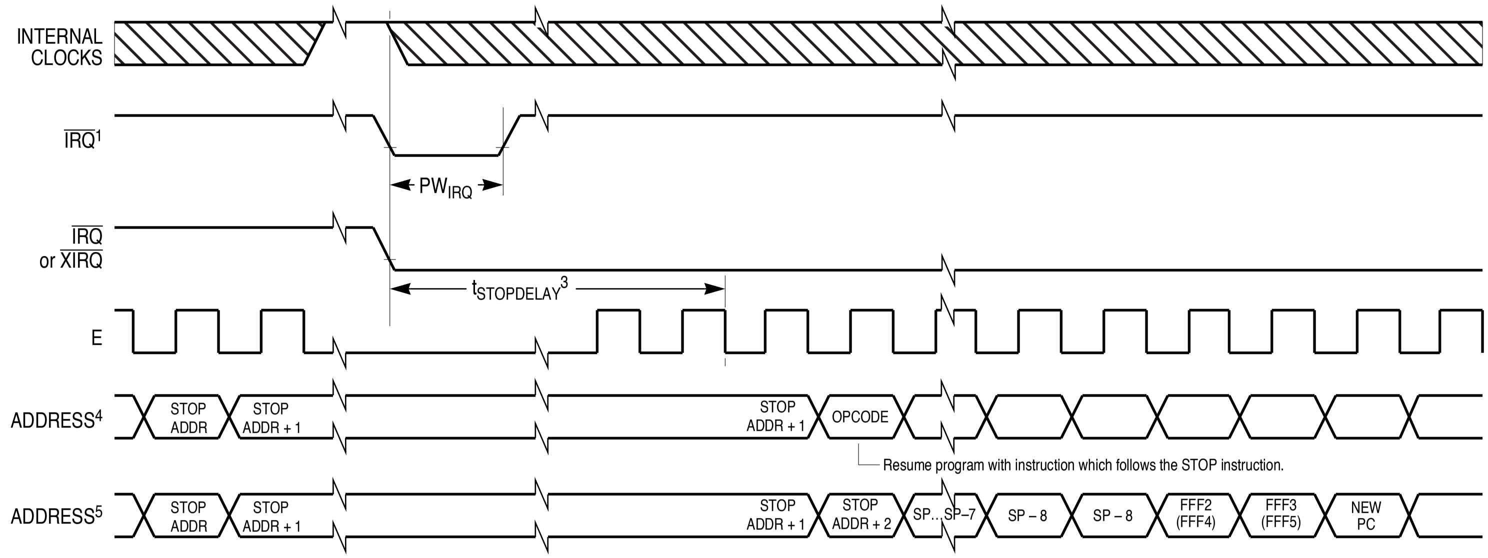


Figure A-3 POR and External Reset Timing Diagram



NOTES:
1. Edge Sensitive $\overline{IRQ}$ pin (IRQE bit = 1)
2. Level sensitive $\overline{IRQ}$ pin (IRQE bit = 0)
3. $t_{STOPDELAY}$ = 4064 t_{CYC} if DLY bit = 1 or 4 t_{CYC} if DLY = 0.
4. $\overline{XIRQ}$ with X bit in CCR = 1.
5. $\overline{IRQ}$ or ($\overline{XIRQ}$ with X bit in CCR = 0).

STOP RECOVERY TIM

Figure A-4 STOP Recovery Timing Diagram

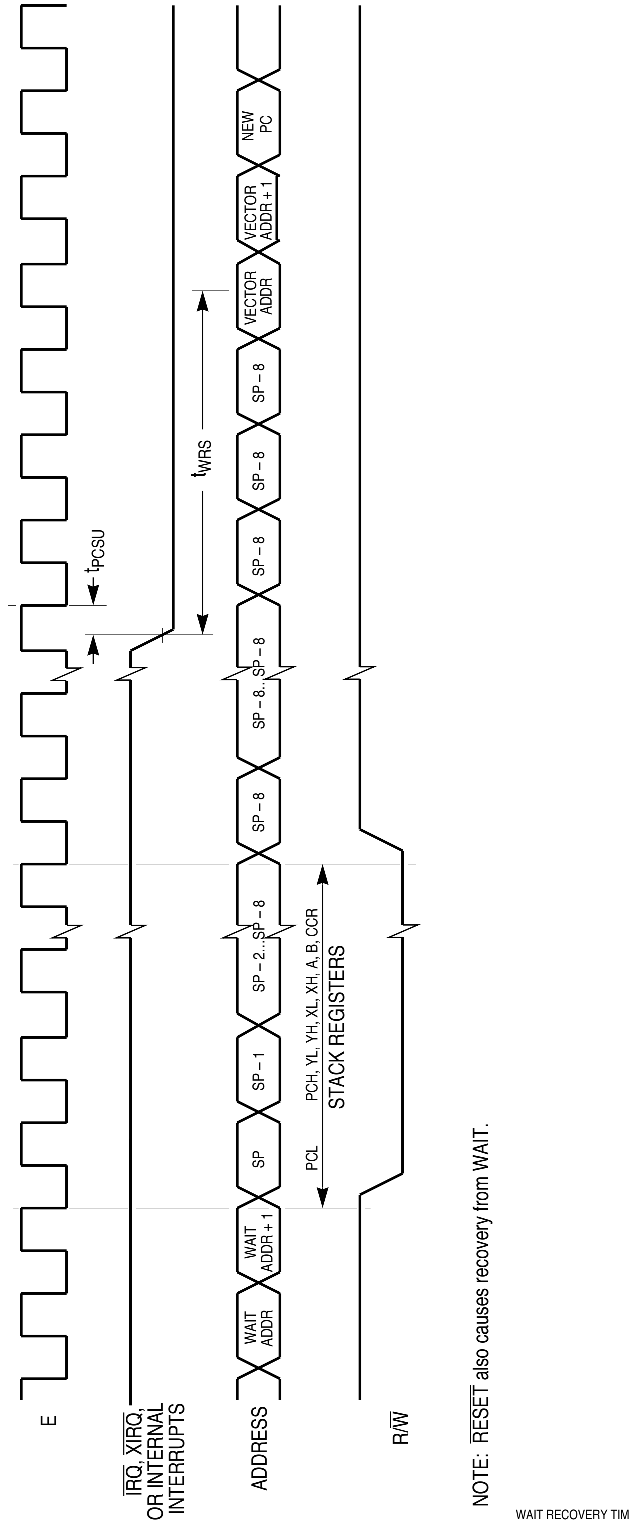


Figure A-5 WAIT Recovery Timing Diagram

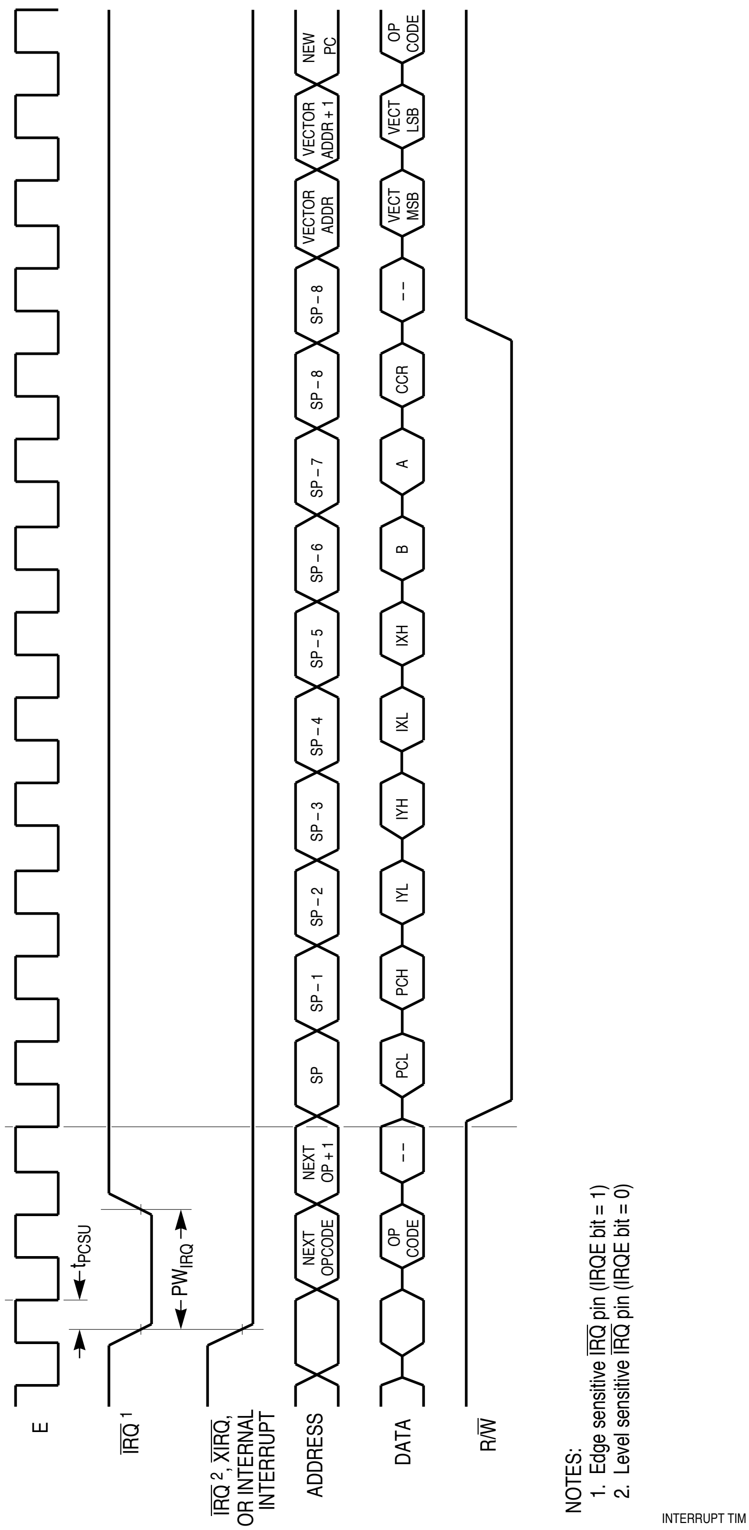


Figure A-6 Interrupt Timing Diagram

Table A-5 Peripheral Port Timing

V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H								
Characteristic	**Symbol**	**1.0 MHz**		**2.0 MHz**		**3.0 MHz**		**Unit**
		Min	**Max**	**Min**	**Max**	**Min**	**Max**	
Frequency of Operation (E-Clock Frequency)	f_o	dc	1.0	dc	2.0	dc	3.0	MHz
E-Clock Period	t_{cyc}	1000	—	500	—	333	—	ns
Peripheral Data Setup Time (MCU Read of Ports A, C, D, and E)	t_{PDSU}	100	—	100	—	100	—	ns
Peripheral Data Hold Time (MCU Read of Ports A, C, D, and E)	t_{PDH}	50	—	50	—	50	—	ns
Delay Time, Peripheral Data Write	t_{PWD}							ns
MCU Write to Port A		—	200	—	200	—	200	
MCU Writes to Ports B, C, and D $t_{PWD} = 1/4\ t_{cyc} + 100$ ns		—	350	—	225	—	183	
Input Data Setup Time (Port C)	t_{IS}	60	—	60	—	60	—	ns
Input Data Hold Time (Port C)	t_{IH}	100	—	100	—	100	—	ns
Delay Time, E Fall to STRB $t_{DEB} = 1/4\ t_{cyc} + 100$ ns	t_{DEB}	—	350	—	225	—	183	ns
Setup Time, STRA Asserted to E Fall (Note 1)	t_{AES}	0	—	0	—	0	—	ns
Delay Time, STRA Asserted to Port C Data Output Valid	t_{PCD}	—	100	—	100	—	100	ns
Hold Time, STRA Negated to Port C Data	t_{PCH}	10	—	10	—	10	—	ns
Three-State Hold Time	t_{PCZ}	—	150	—	150	—	150	ns

NOTES:
1. If this setup time is met, STRB acknowledges in the next cycle. If it is not met, the response may be delayed one more cycle.
2. Port C 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.

A

Table A-5a Peripheral Port Timing (MC68L11A8)

V_{DD} = 3.0 Vdc to 5.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H						
Characteristic	**Symbol**	**1.0 MHz**		**2.0 MHz**		**Unit**
		Min	**Max**	**Min**	**Max**	
Frequency of Operation (E-Clock Frequency)	f_o	dc	1.0	dc	2.0	MHz
E-Clock Period	t_{cyc}	1000	—	500	—	ns
Peripheral Data Setup Time (MCU Read of Ports A, C, D, and E)	t_{PDSU}	100	—	100	—	ns
Peripheral Data Hold Time (MCU Read of Ports A, C, D, and E)	t_{PDH}	50	—	50	—	ns
Delay Time, Peripheral Data Write	t_{PWD}					ns
MCU Write to Port A		—	250	—	250	
MCU Writes to Ports B, C, and D t_{PWD} = 1/4 t_{cyc} + 150 ns		—	400	—	275	
Input Data Setup Time (Port C)	t_{IS}	60	—	60	—	ns
Input Data Hold Time (Port C)	t_{IH}	100	—	100	—	ns
Delay Time, E Fall to STRB t_{DEB} = 1/4 t_{cyc} + 150 ns	t_{DEB}	—	400	—	275	ns
Setup Time, STRA Asserted to E Fall (Note 1)	t_{AES}	0	—	0	—	ns
Delay Time, STRA Asserted to Port C Data Output Valid	t_{PCD}	—	100	—	100	ns
Hold Time, STRA Negated to Port C Data	t_{PCH}	10	—	10	—	ns
Three-State Hold Time	t_{PCZ}	—	150	—	150	ns

NOTES:

1. If this setup time is met, STRB acknowledges in the next cycle. If it is not met, the response may be delayed one more cycle.
2. Port C 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.

A

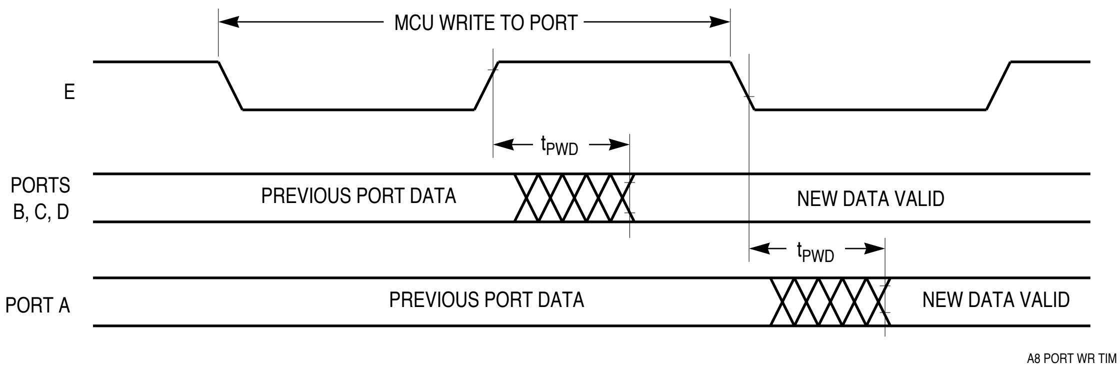


Figure A-7 Port Write Timing Diagram

A

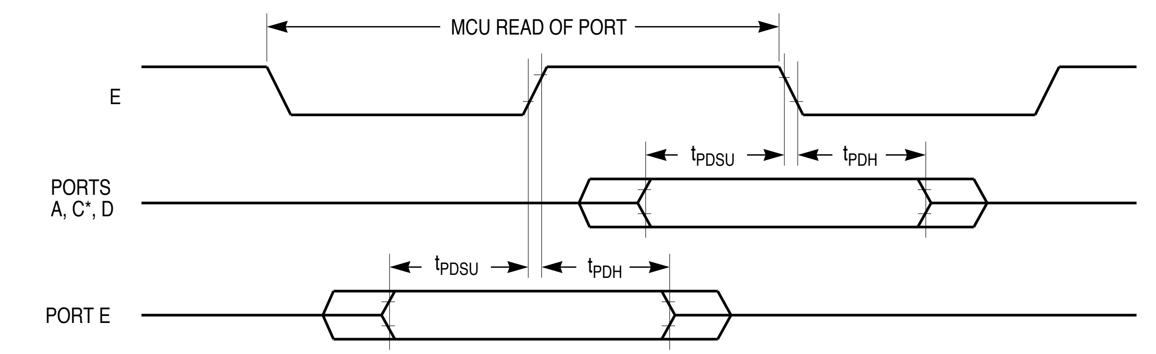


Figure A-8 Port Read Timing Diagram

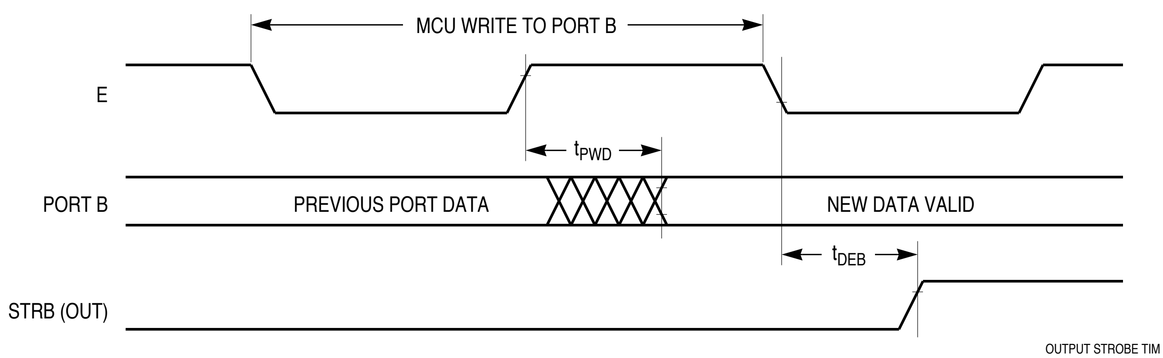


Figure A-9 Simple Output Strobe Timing Diagram

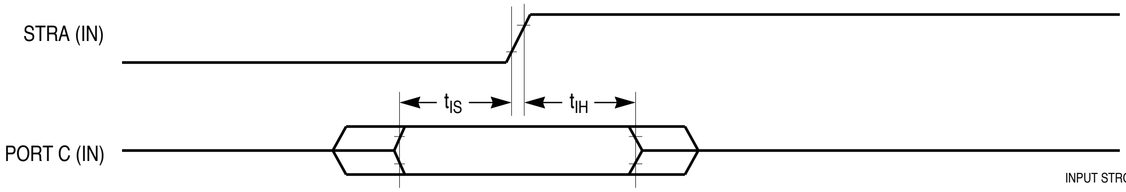
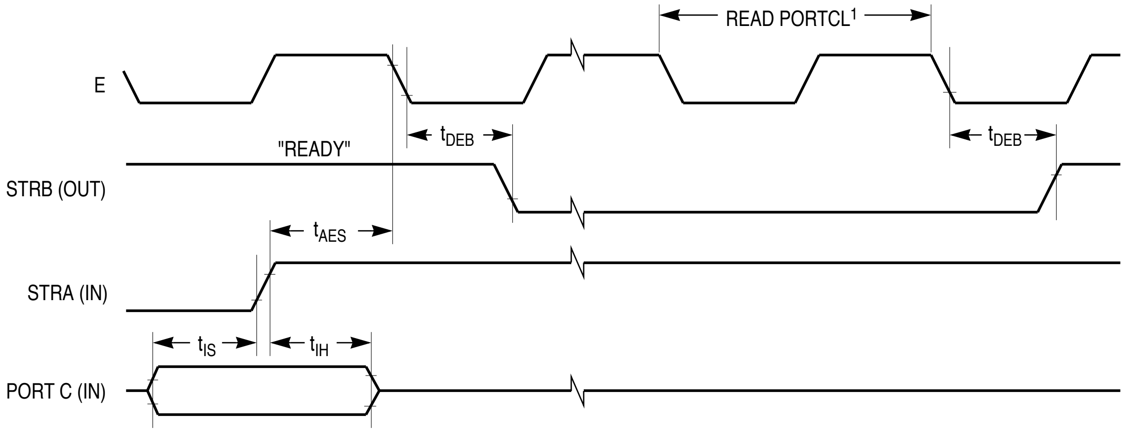


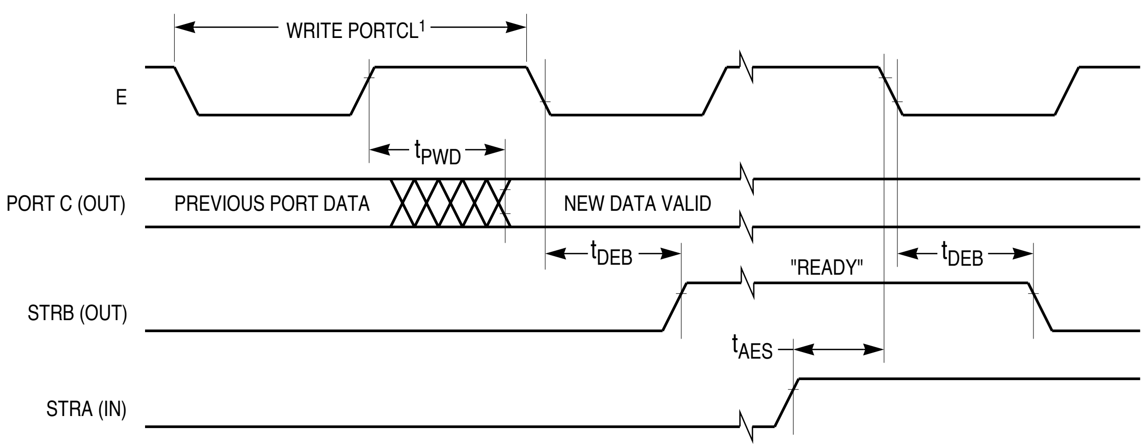
Figure A-10 Simple Input Strobe Timing Diagram



NOTES:
1. After reading PIOC with STAF set
2. Figure shows rising edge STRA (EGA = 1) and high true STRB (INVB = 1).

PORTC INPUT HNDSHK TIM

Figure A-11 Port C Input Handshake Timing Diagram

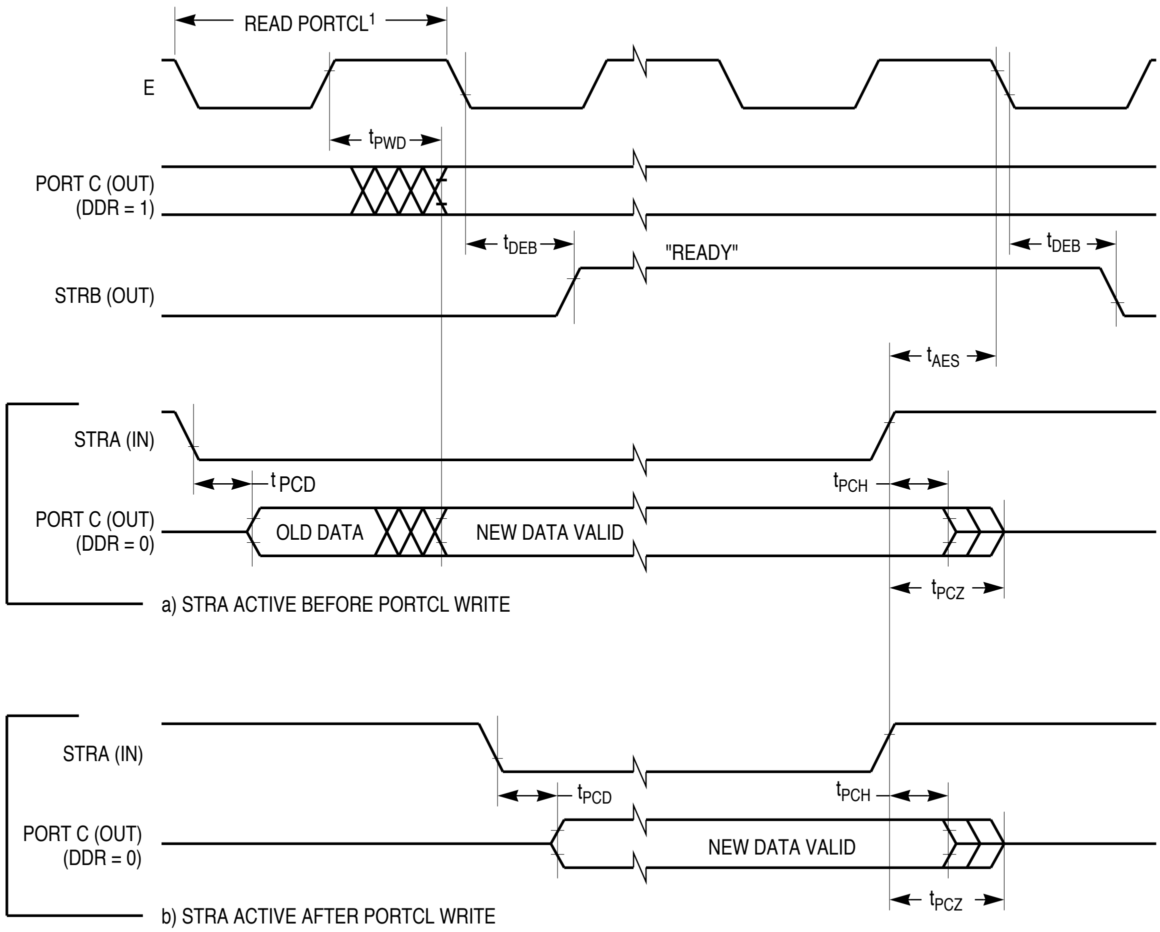


NOTES:
1. After reading PIOC with STAF set
2. Figure shows rising edge STRA (EGA = 1) and high true STRB (INVB = 1).

PORTC OUTPUT HNDSHK TIM

Figure A-12 Port C Output Handshake Timing Diagram

A



NOTES:
1. After reading PIOC with STAF set
2. Figure shows rising edge STRA (EGA = 1) and high true STRB (INVB = 1).

3STATE VAR HNDSHK

Figure A-13 Three-State Variation of Output Handshake Timing Diagram (STRA Enables Output Buffer)

A

Table A-6 Analog-To-Digital Converter Characteristics

V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, 750 kHz ≤ E ≤ 3.0 MHz, unless otherwise noted						
Characteristic	**Parameter**	**Min**	**Absolute**	**2.0 MHz**	**3.0 MHz**	**Unit**
				Max	**Max**	
Resolution	Number of Bits Resolved by A/D Converter	—	8	—	—	Bits
Non-Linearity	Maximum Deviation from the Ideal A/D Transfer Characteristics	—	—	± 1/2	± 1	LSB
Zero Error	Difference Between the Output of an Ideal and an Actual for Zero Input Voltage	—	—	± 1/2	± 1	LSB
Full Scale Error	Difference Between the Output of an Ideal and an Actual A/D for Full-Scale Input Voltage	—	—	± 1/2	± 1	LSB
Total Unadjusted Error	Maximum Sum of Non-Linearity, Zero Error, and Full-Scale Error	—	—	± 1/2	± 1 1/2	LSB
Quantization Error	Uncertainty Because of Converter Resolution	—	—	± 1/2	± 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	± 2	LSB
Conversion Range	Analog Input Voltage Range	V_{RL}	—	V_{RH}	V_{RH}	V
V_{RH}	Maximum Analog Reference Voltage (Note 2)	V_{RL}	—	V_{DD} + 0.1	V_{DD} + 0.1	V
V_{RL}	Minimum Analog Reference Voltage (Note 2)	V_{SS} – 0.1	—	V_{RH}	V_{RH}	V
ΔV_R	Minimum Difference between V_{RH} and V_{RL} (Note 2)	3	—	—	—	V
Conversion Time	Total Time to Perform a Single Analog-to-Digital Conversion: a. E Clock b. Internal RC Oscillator	 — —	 32 —	 — t_{cyc} + 32	 — t_{cyc} + 32	 t_{cyc} 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	FF	Hex
Sample Acquisition Time	Analog Input Acquisition Sampling Time: a. E Clock b. Internal RC Oscillator	 — —	 12 —	 — 12	 — 12	 t_{cyc} µs
Sample/Hold Capacitance	Input Capacitance during Sample PE0-PE7	—	20 (Typ)	—	—	pF
Input Leakage	Input Leakage on A/D PinsPE0-PE7 V_{RL}, V_{RH}	— —	— —	400 1.0	400 1.0	nA µA

NOTES:
1. Source impedances greater than 10 kΩ affect accuracy adversely because of input leakage.
2. Performance verified down to 2.5 V ΔV_R, but accuracy is tested and guaranteed at ΔV_R = 5 V ± 10%.

A

A

Table A-6a Analog-To-Digital Converter Characteristics (MC68L11A8)

V_{DD} = 3.0 Vdc to 5.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, 750 kHz ≤ E ≤ 2.0 MHz, unless otherwise noted					
Characteristic	**Parameter**	**Min**	**Absolute**	**Max**	**Unit**
Resolution	Number of Bits Resolved by A/D Converter	—	8	—	Bits
Non-Linearity	Maximum Deviation from the Ideal A/D Transfer Characteristics	—	—	± 1	LSB
Zero Error	Difference Between the Output of an Ideal and an Actual for Zero Input Voltage	—	—	± 1	LSB
Full Scale Error	Difference Between the Output of an Ideal and an Actual A/D for Full-Scale Input Voltage	—	—	± 1	LSB
Total Unadjusted Error	Maximum Sum of Non-Linearity, Zero Error, and Full-Scale Error	—	—	±1 1/2	LSB
Quantization Error	Uncertainty Because of 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	—	—	± 2	LSB
Conversion Range	Analog Input Voltage Range	V_{RL}	—	V_{RH}	V
V_{RH}	Maximum Analog Reference Voltage(Note 2)	V_{RL}	—	V_{DD} + 0.1	V
V_{RL}	Minimum Analog Reference Voltage (Note 2)	V_{SS} – 0.1	—	V_{RH}	V
ΔV_R	Minimum Difference between V_{RH} and V_{RL} (Note 2)	3	—	—	V
Conversion Time	Total Time to Perform a Single Analog-to-Digital Conversion: a. E Clock b. Internal RC Oscillator	 — —	 32 —	 — t_{cyc} + 32	 t_{cyc} 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	 t_{cyc} µs
Sample/Hold Capacitance	Input Capacitance during Sample PE0-PE7	—	20 (Typ)	—	pF
Input Leakage	Input Leakage on A/D PinsPE0-PE7 V_{RL}, V_{RH}	— —	— —	400 1.0	nA µA

NOTES:
1. Source impedances greater than 10 kΩ affect accuracy adversely because of input leakage.

Table A-7 Expansion Bus Timing

V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = T_L to T_H									
Num	**Characteristic**	**Symbol**	**1.0 MHz**		**2.0 MHz**		**3.0 MHz**		**Unit**
			Min	**Max**	**Min**	**Max**	**Min**	**Max**	
	Frequency of Operation (E-Clock Frequency)	f_o	dc	1.0	dc	2.0	dc	3.0	MHz
1	Cycle Time	t_{cyc}	1000	—	500	—	333	—	ns
2	Pulse Width, E Low PW_{EL} = 1/2 t_{cyc} – 23 ns(Note 1)	PW_{EL}	477	—	227	—	146	—	ns
3	Pulse Width, E High PW_{EH} = 1/2 t_{cyc} – 28 ns(Note 1)	PW_{EH}	472	—	222	—	141	—	ns
4a, b	E and AS Rise and Fall Time	t_r t_f	—	20 20	—	20 20	—	20 15	ns
9	Address Hold Time t_{AH} = 1/8 t_{cyc} – 29.5 ns (Note 1, 2a)	t_{AH}	95.5	—	33	—	26	—	ns
12	Non-Muxed Address Valid Time to E Rise t_{AV} = PW_{EL} – (t_{ASD} + 80 ns)(Note 1, 2a)	t_{AV}	281.5	—	94	—	54	—	ns
17	Read Data Setup Time	t_{DSR}	30	—	30	—	30	—	ns
18	Read Data Hold Time (Max = t_{MAD})	t_{DHR}	0	145.5	0	83	0	51	ns
19	Write Data Delay Time t_{DDW} = 1/8 t_{cyc} + 65.5 ns (Note 1, 2a)	t_{DDW}	—	190.5	—	128	—	71	ns
21	Write Data Hold Time t_{DHW} = 1/8 t_{cyc} – 29.5 ns(Note 1, 2a)	t_{DHW}	95.5	—	33	—	26	—	ns
22	Muxed Address Valid Time to E Rise t_{AVM} = PW_{EL} – (t_{ASD} + 90 ns)(Note 1, 2a)	t_{AVM}	271.5	—	84	—	54	—	ns
24	Muxed Address Valid Time to AS Fall t_{ASL} = PW_{ASH} – 70 ns(Note 1)	t_{ASL}	151	—	26	—	13	—	ns
25	Muxed Address Hold Time t_{AHL} = 1/8 t_{cyc} – 29.5 ns(Note 1, 2b)	t_{AHL}	95.5	—	33	—	31	—	ns
26	Delay Time, E to AS Rise t_{ASD} = 1/8 t_{cyc} – 9.5 ns (Note 1, 2a)	t_{ASD}	115.5	—	53	—	31	—	ns
27	Pulse Width, AS High PW_{ASH} = 1/4 t_{cyc} – 29 ns(Note 1)	PW_{ASH}	221	—	96	—	63	—	ns
28	Delay Time, AS to E Rise t_{ASED} = 1/8 t_{cyc} – 9.5 ns(Note 1, 2b)	t_{ASED}	115.5	—	53	—	31	—	ns
29	MPU Address Access Time(Note 2a) t_{ACCA} = t_{cyc} – (PW_{EL} – t_{AVM}) – t_{DSR} – t_f	t_{ACCA}	744.5	—	307	—	196	—	ns
35	MPU Access Time t_{ACCE} = PW_{EH} – t_{DSR}	t_{ACCE}	—	442	—	192	—	111	ns
36	Muxed Address Delay (Previous Cycle MPU Read) t_{MAD} = t_{ASD} + 30 ns(Note 1, 2a)	t_{MAD}	145.5	—	83	—	51	—	ns

NOTES:
1. Formula only for dc to 2 MHz.
2. Input clocks with duty cycles other than 50% 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 following expressions in place of 1/8 t_{cyc} in the above formulas, where applicable:
   (a) (1-DC) x 1/4 t_{cyc}
   (b) DC x 1/4 t_{cyc}
   Where:
   DC is the decimal value of duty cycle percentage (high time).
3. All timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted.

A

Table A-7a Expansion Bus Timing (MC68L11A8)

V_{DD} = 3.0 Vdc to 5.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H							
Num	**Characteristic**	**Symbol**	**1.0 MHz**		**2.0 MHz**		**Unit**
			Min	**Max**	**Min**	**Max**	
	Frequency of Operation (E-Clock Frequency)	f_o	dc	1.0	dc	2.0	MHz
1	Cycle Time	t_{cyc}	1000	—	500	—	ns
2	Pulse Width, E Low $PW_{EL} = 1/2\ t_{cyc} - 23$ ns (Note 1)	PW_{EL}	475	—	225	—	ns
3	Pulse Width, E High $PW_{EH} = 1/2\ t_{cyc} - 28$ ns (Note 1)	PW_{EH}	470	—	220	—	ns
4a, b	E and AS Rise and Fall Time	t_r t_f	—	25 25	—	25 25	ns
9	Address Hold Time $t_{AH} = 1/8\ t_{cyc} - 29.5$ ns (Note 1, 2a)	t_{AH}	95	—	33	—	ns
12	Non-Muxed Address Valid Time to E Rise $t_{AV} = PW_{EL} - (t_{ASD} + 80$ ns) (Note 1, 2a)	t_{AV}	275	—	88	—	ns
17	Read Data Setup Time	t_{DSR}	30	—	30	—	ns
18	Read Data Hold Time (Max = t_{MAD})	t_{DHR}	0	150	0	88	ns
19	Write Data Delay Time $t_{DDW} = 1/8\ t_{cyc} + 65.5$ ns (Note 1, 2a)	t_{DDW}	—	195	—	133	ns
21	Write Data Hold Time $t_{DHW} = 1/8\ t_{cyc} - 29.5$ ns (Note 1, 2a)	t_{DHW}	95	—	33	—	ns
22	Muxed Address Valid Time to E Rise $t_{AVM} = PW_{EL} - (t_{ASD} + 90$ ns) (Note 1, 2a)	t_{AVM}	265	—	78	—	ns
24	Muxed Address Valid Time to AS Fall $t_{ASL} = PW_{ASH} - 70$ ns (Note 1)	t_{ASL}	150	—	25	—	ns
25	Muxed Address Hold Time $t_{AHL} = 1/8\ t_{cyc} - 29.5$ ns (Note 1, 2b)	t_{AHL}	95	—	33	—	ns
26	Delay Time, E to AS Rise $t_{ASD} = 1/8\ t_{cyc} - 9.5$ ns (Note 1, 2a)	t_{ASD}	120	—	58	—	ns
27	Pulse Width, AS High $PW_{ASH} = 1/4\ t_{cyc} - 29$ ns (Note 1)	PW_{ASH}	220	—	95	—	ns
28	Delay Time, AS to E Rise $t_{ASED} = 1/8\ t_{cyc} - 9.5$ ns (Note 1, 2b)	t_{ASED}	120	—	58	—	ns
29	MPU Address Access Time (Note 2a) $t_{ACCA} = t_{cyc} - (PW_{EL} - t_{AVM}) - t_{DSR} - t_f$	t_{ACCA}	735	—	298	—	ns
35	MPU Access Time $t_{ACCE} = PW_{EH} - t_{DSR}$	t_{ACCE}	—	440	—	190	ns
36	Muxed Address Delay (Previous Cycle MPU Read) $t_{MAD} = t_{ASD} + 30$ ns (Note 1, 2a)	t_{MAD}	150	—	88	—	ns

NOTES:
1. Formula only for dc to 2 MHz.
2. Input clocks with duty cycles other than 50% 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 following expressions in place of 1/8 t_{cyc} in the above formulas, where applicable:
   (a) (1-DC) x 1/4 t_{cyc}
   (b) DC x 1/4 t_{cyc}
   Where:
   DC is the decimal value of duty cycle percentage (high time).
3. All timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted.

A

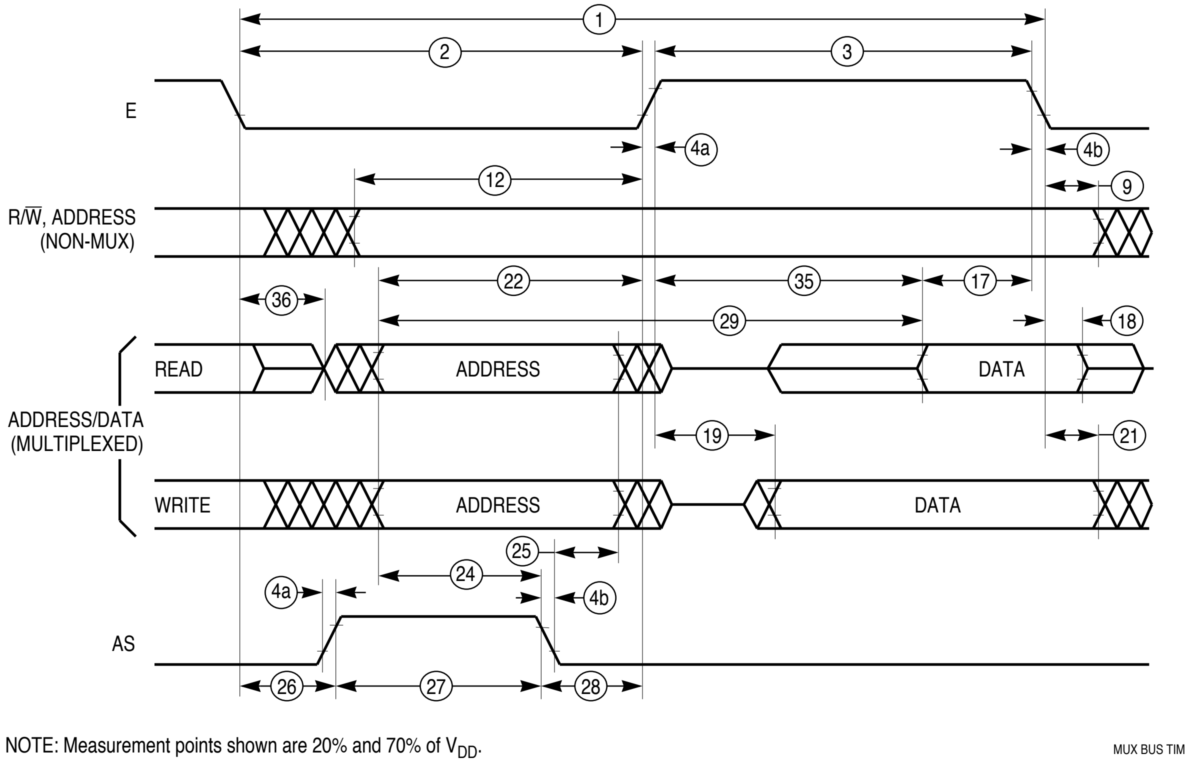


Figure A-14 Multiplexed Expansion Bus Timing Diagram

A

Table A-8 Serial Peripheral Interface (SPI) Timing

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

Num	Characteristic	Symbol	2.0 MHz		3.0 MHz		Unit
			Min	Max	Min	Max	
	Operating Frequency Master Slave	 $f_{op(m)}$ $f_{op(s)}$	 dc dc	 0.5 2.0	 dc dc	 0.5 3.0	 f_{op} MHz
1	Cycle Time Master Slave	 $t_{cyc(m)}$ $t_{cyc(s)}$	 2.0 500	 — —	 2.0 333	 — —	 t_{cyc} ns
2	Enable Lead Time Master (Note 2) Slave	 $t_{lead(m)}$ $t_{lead(s)}$	 — 250	 — —	 — 240	 — —	 ns ns
3	Enable Lag Time Master (Note 2) Slave	 $t_{lag(m)}$ $t_{lag(s)}$	 — 250	 — —	 — 240	 — —	 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	120	ns
9	Disable Time (Hold Time to High-Impedance State) Slave	t_{dis}	—	240	—	167	ns
10	Data Valid (After Enable Edge)(Note 3)	$t_{v(s)}$	—	240	—	167	ns
11	Data Hold Time (Outputs) (After Enable Edge)	t_{ho}	0	—	0	—	ns
12	Rise Time (20% V_{DD} to 70% V_{DD}, CL = 200 pF) 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 = 200 pF) 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

NOTES:
1. All timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted.
2. Signal production depends on software.
3. Assumes 200 pF load on all SPI pins.

A

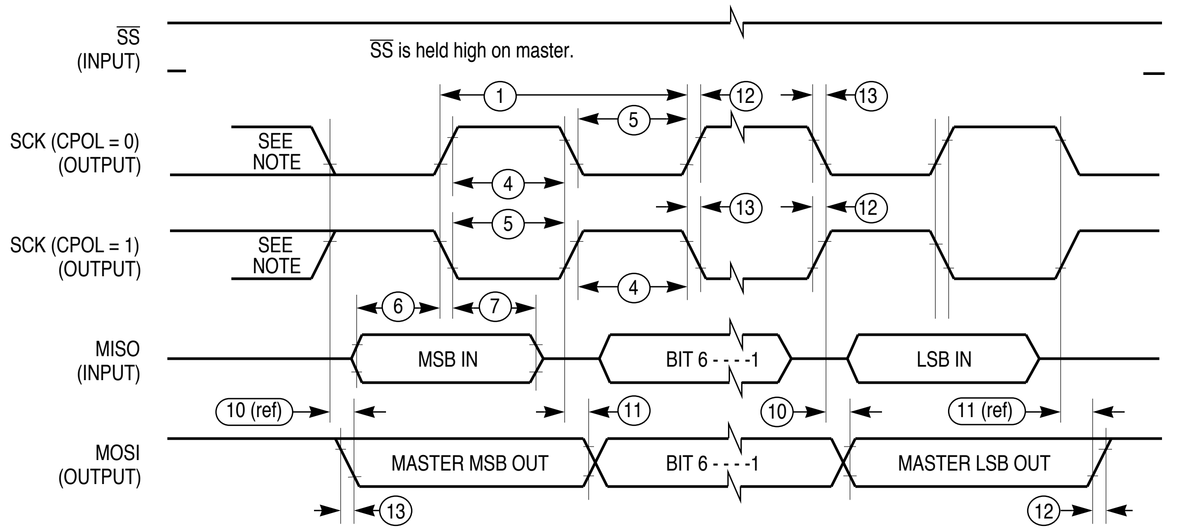
Table A-8a Serial Peripheral Interface (SPI) Timing (MC68L11A8)

V_{DD} = 3.0 Vdc to 5.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H

Num	Characteristic	Symbol	1.0 MHz		2.0 MHz		Unit
			Min	Max	Min	Max	
	Operating Frequency Master Slave	$f_{op(m)}$ $f_{op(s)}$	dc dc	0.5 1.0	dc dc	0.5 2.0	f_{op} MHz
1	Cycle Time Master Slave	$t_{cyc(m)}$ $t_{cyc(s)}$	2.0 1000	— —	2.0 500	— —	t_{cyc} ns
2	Enable Lead Time Master (Note 2) Slave	$t_{lead(m)}$ $t_{lead(s)}$	— 500	— —	— 250	— —	ns ns
3	Enable Lag Time Master (Note 2) Slave	$t_{lag(m)}$ $t_{lag(s)}$	— 500	— —	— 250	— —	ns ns
4	Clock (SCK) High Time Master Slave	$t_{w(SCKH)m}$ $t_{w(SCKH)s}$	680 380	— —	340 190	— —	ns ns
5	Clock (SCK) Low Time Master Slave	$t_{w(SCKL)m}$ $t_{w(SCKL)s}$	680 380	— —	340 190	— —	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	120	ns
9	Disable Time (Hold Time to High-Impedance State) Slave	t_{dis}	—	240	—	240	ns
10	Data Valid (After Enable Edge)(Note 3)	$t_{v(s)}$	—	240	—	240	ns
11	Data Hold Time (Outputs) (After Enable Edge)	t_{ho}	0	—	0	—	ns
12	Rise Time (20% V_{DD} to 70% V_{DD}, CL = 200 pF) 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 = 200 pF) 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

NOTES:
1. All timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted.
2. Signal production depends on software.
3. Assumes 200 pF load on all SPI pins.

A

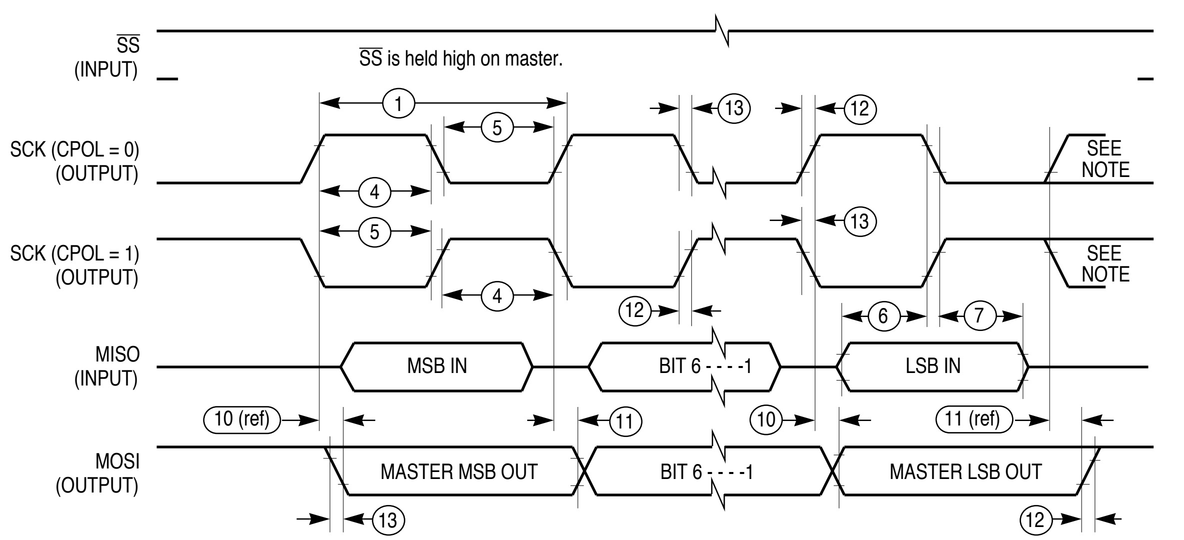


NOTE: This first clock edge is generated internally but is not seen at the SCK pin.

SPI MASTER CPHA0 TIM

a) SPI Master Timing (CPHA = 0)

A

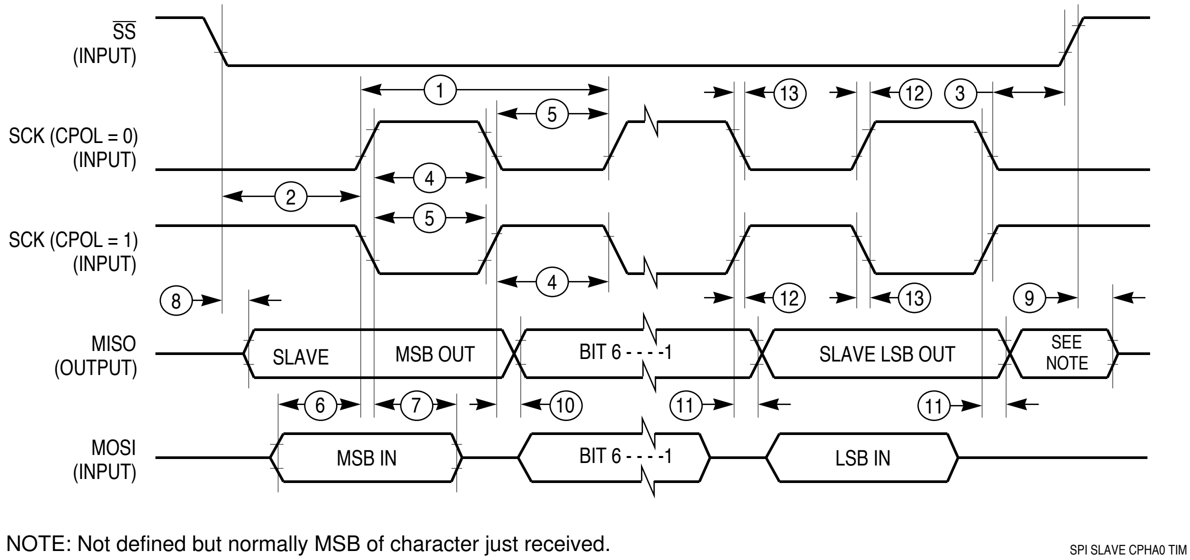


NOTE: This last clock edge is generated internally but is not seen at the SCK pin.

SPI MASTER CPHA1 TIM

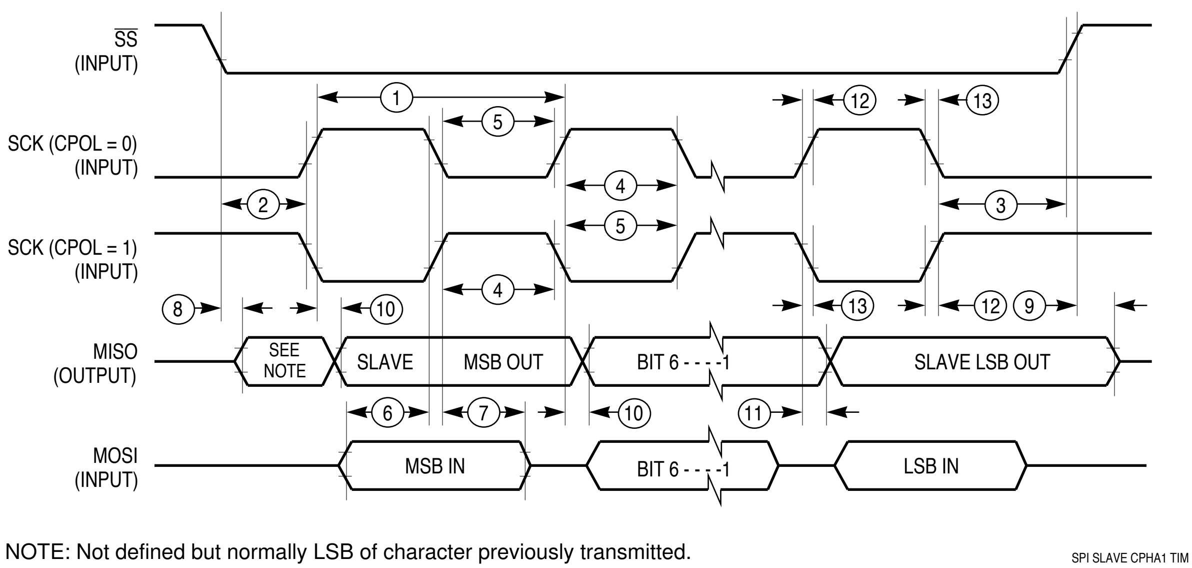
b) SPI Master Timing (CPHA = 1)

Figure A-15 SPI Timing Diagram (1 of 2)



c) SPI Slave Timing (CPHA = 0)

A



d) SPI Slave Timing (CPHA = 1)

Figure A-15 SPI Timing Diagrams (2 of 2)

Table A-9 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 (Note 1)	<1.0 MHz, RCO Enabled	10	15	20	ms
	1.0 to 2.0 MHz, RCO Enabled	20	Must use RCO	Must use RCO	
	≥ 2.0 MHz (or Anytime RCO Enabled)	10	15	20	
Erase Time (Note 1)	Byte, Row and Bulk	10	10	10	ms
Write/Erase Endurance (Note 2)		10,000	10,000	10,000	Cycles
Data Retention (Note 2)		10	10	10	Years

NOTES:
1. The RC oscillator (RCO) 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. Refer to Reliability Monitor Report (current quarterly issue) for current failure rate information.

Table A-9a EEPROM Characteristics (MC68L11A8)

A

V_{DD} = 3.0 Vdc to 5.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H

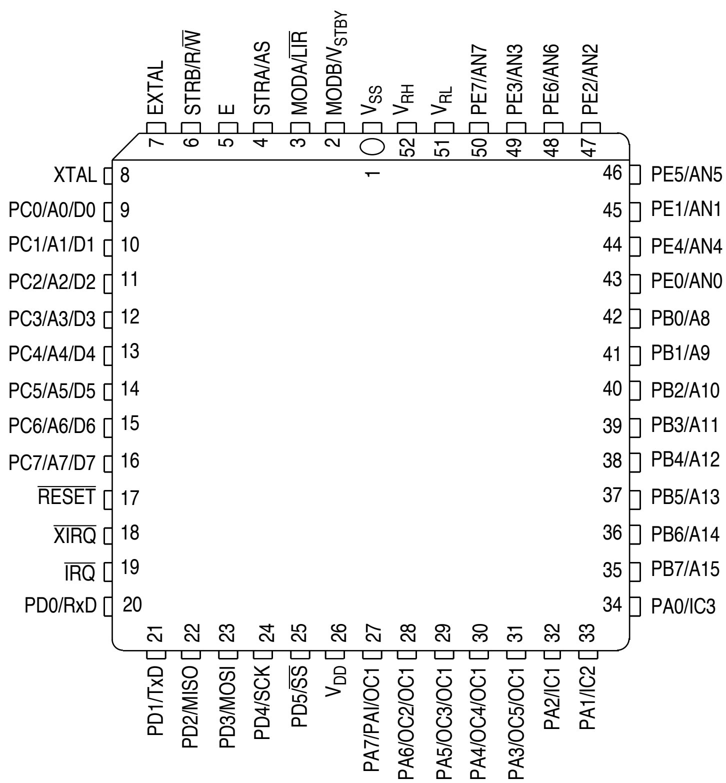
Characteristic		Temperature Range	Unit
		– 20 to 70° C	
Programming Time (Note 1)	3 V, E ≤ 2.0 MHz, RCO Enabled	25	ms
	5 V, E ≤ 2.0 MHz, RCO Enabled	10	
Erase Time (Byte, Row and Bulk) (Note 1)	3 V, E ≤ 2.0 MHz, RCO Enabled	25	ms
	5 V, E ≤ 2.0 MHz, RCO Enabled	10	
Write/Erase Endurance (Note 2)		10,000	Cycles
Data Retention (Note 2)		10	Years

NOTES:
1. The RC oscillator (RCO) must be enabled (by setting the CSEL bit in the OPTION register) for EEPROM programming and erasure.
2. Refer to Reliability Monitor Report (current quarterly issue) for current failure rate information.

B MECHANICAL DATA AND ORDERING INFORMATION

B.1 Pin Assignments

The MC68HC11A8 is available in the 52-pin plastic leaded chip carrier (PLCC), the 48-pin dual in-line package (DIP), or the 64-pin quad flat pack (QFP).



A8 52-PIN PLCC

Figure B-1 52-Pin PLCC

B

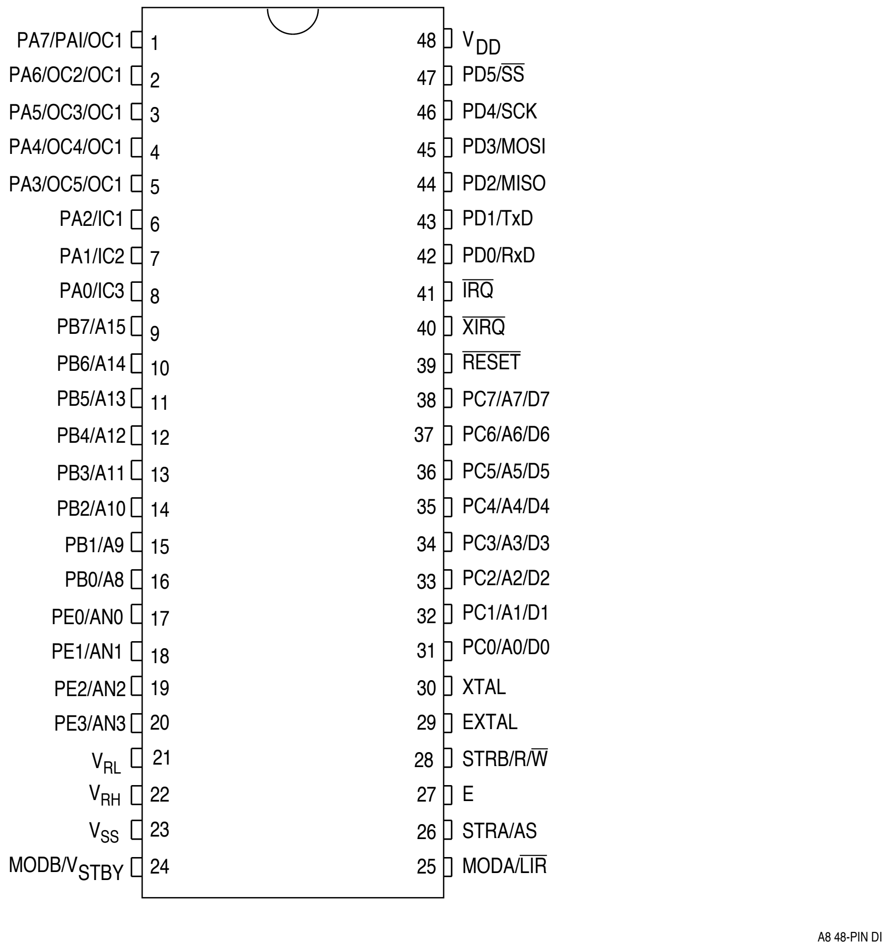


Figure B-2 48-Pin DIP

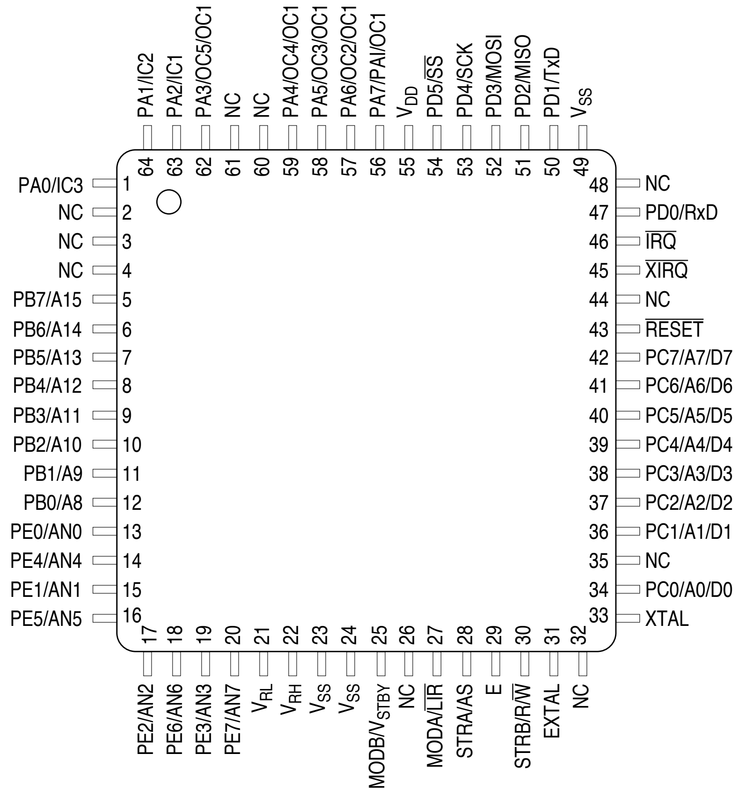


Figure B-3 64-Pin QFP

B.2 Package Dimensions

The latest versions of case outlines 778-02, 767-02, and 840B-01 can be obtained from our website at http://design-net.com.

Table B-1 Ordering Information

Package	Temperature	CONFIG	Description	MC Order Number
52-Pin PLCC	– 40° to + 85°C	$0C	No ROM, No EEPROM	MC68HC11A0CFN2
	– 40° to + 85°C	$0C	No ROM, No EEPROM, 3 MHz	MC68HC11A0CFN3
	– 40° to + 85°C	$0D	No ROM	MC68HC11A1CFN2
	– 40° to + 85°C	$0D	No ROM, 3 MHz	MC68HC11A1CFN3
	– 40° to + 105°C	$0D	No ROM	MC68HC11A1VFN2
	– 40° to + 125°C	$0D	No ROM	MC68HC11A1MFN2
	– 40° to + 85°C	$09	No ROM, COP On	MC68HCP11A1CFN2
	– 40° to + 85°C	$09	No ROM, COP On, 3 MHz	MC68HCP11A1CFN3
	– 40° to + 105°C	$09	No ROM, COP On	MC68HCP11A1VFN2
	– 40° to + 125°C	$09	No ROM, COP On	MC68HCP11A1MFN2
	– 40° to + 85°C	$0F	BUFFALO ROM	MC68HC11A8BCFN2
	– 40° to + 105°C	$0F	BUFFALO ROM	MC68HC11A8VCFN2
	– 40° to + 125°C	$0F	BUFFALO ROM	MC68HC11A8MCFN2
48-Pin DIP	– 40° to + 85°C	$0C	No ROM, No EEPROM	MC68HC11A0CP2
	– 40° to + 85°C	$0C	No ROM, No EEPROM, 3 MHz	MC68HC11A0CP3
	– 40° to + 85°C	$0D	No ROM	MC68HC11A1CP2
	– 40° to + 85°C	$0D	No ROM, 3 MHz	MC68HC11A1CP3
	– 40° to + 105°C	$0D	No ROM	MC68HC11A1VP2
	– 40° to + 125°C	$0D	No ROM	MC68HC11A1MP2
	– 40° to + 85°C	$09	No ROM, COP On	MC68HCP11A1CP2
	– 40° to + 85°C	$09	No ROM, COP On, 3 MHz	MC68HCP11A1CP3
	– 40° to + 105°C	$09	No ROM, COP On	MC68HCP11A1VP2
	– 40° to + 125°C	$09	No ROM, COP On	MC68HCP11A1MP2
	– 40° to + 85°C	$0F	BUFFALO ROM	MC68HC11A8BCP2
	– 40° to + 105°C	$0F	BUFFALO ROM	MC68HC11A8BVP2
	– 40° to + 125°C	$0F	BUFFALO ROM	MC68HC11A8BMP2
64-Pin QFP	– 40° to + 85°C	$0C	No ROM, No EEPROM	MC68HC11A0CFU2
	– 40° to + 85°C	$0C	No ROM, No EEPROM, 3 MHz	MC68HC11A0CFU3
	– 40° to + 85°C	$0D	No ROM	MC68HC11A1CFU2
	– 40° to + 85°C	$0D	No ROM, 3 MHz	MC68HC11A1CFU3
	– 40° to + 105°C	$0D	No ROM	MC68HC11 A1VFU2
	– 40° to + 125°C	$0D	No ROM	MC68HC11A1MFU2
	– 40° to + 85°C	$0F	BUFFALO ROM	MC68HC11A8BCFU2
	– 40° to + 105°C	$0F	BUFFALO ROM	MC68HC11A8VCFU2

MC 68 HC P 7 11XX B C FN 3 R2

QUALIFICATION LEVEL (MC)
- MC — FULLY SPECIFIED AND QUALIFIED
- XC — PILOT PRODUCTION DEVICE
- PC — ENGINEERING SAMPLE

NUMERIC DESIGNATOR (OPTIONAL) (68)

OPERATING VOLTAGE RANGE (HC)
- HC — HCMOS (V_{DD} = 5.0 VDC ±10%)
- L — HCMOS (V_{DD} = 3.0 VDC TO 5.5 VDC)

COP OPTION (ONLY ON A-SERIES DEVICES) (P)
- NONE — COP DISABLED
- P — COP ENABLED

MEMORY TYPE (7)
- BLANK — MASKED ROM OR NO ROM
- 7 — EPROM/OTPROM
- 8 — EEPROM

BASE PART NUMBER (11XX)
- 11A8, 11D3, 11E9, 11K4, ETC.

MONITOR MASK (B)
- NONE — BLANK
- B — BUFFALO

TEMPERATURE RANGE (C)
- NONE — 0°C TO 70°C
- C — – 40°C TO 85°C
- V — – 40°C TO 105°C
- M — – 40°C TO 125°C

PACKAGE TYPE (FN)
- FN — 44/52/68/84-PIN PLCC
- FS — 44/52/68/84-PIN CLCC
- FU — 64/80-PIN QFP
- FB — 44-PIN QFP
- PV — 112-PIN TQFP
- PU — 80/100-PIN TQFP
- PB — 52-PIN TQFP
- P — 40/48-PIN DIP
- S — 48-PIN SDIP

MAXIMUM SPECIFIED CLOCK SPEED (3)
- 2 — 2.0 MHz
- 3 — 3.0 MHz
- 4 — 4.0 MHz

TAPE AND REEL OPTION (R2)
- NONE — STANDARD PACKAGING
- R2 — TAPE AND REEL PACKAGING

HC11 PART NUMBERING

Figure B-4 M68HC11 P/N Options

B

C DEVELOPMENT SUPPORT

C.1 M68HC11EVB — Evaluation Board

Refer to the *M68HC11EVB Evaluation Board User's Manual* (M68HC11EVB/D1). The Evaluation Board (EVB) is a low cost tool for debugging and evaluating M68HC11 MCU-based target system equipment. The EVB is designed to operate in either the debugging or evaluation mode of operation.

C.1.1 EVB Features

- A low cost means of debugging user-assembled code and evaluating target systems incorporating M68HC11 MCUs
- One-line assembler/disassembler
- Host computer downloading capability
- M68HC11 MCU-based debugging/evaluating circuitry
- MC68HC24 Port Replacement Unit (PRU)-based MCU I/O expansion circuitry
- MC6850 Asynchronous Communications Interface Adaptor (ACIA)-based terminal I/O port circuitry
- RS-232C compatible terminal/host computer I/O ports

C

C.2 M68HC11EVBU — Universal Evaluation Board

Refer to *M68HC11EVBU Universal Evaluation Board User's Manual* (M68HC11EVBU/AD1). The EVBU is a low cost development tool for debugging and evaluation of MC68HC11A8, E9, 7E9, and 8E2 MCUs. The EVBU was designed along with a monitor/debugging program called BUFFALO (Bit User Fast Friendly Aid to Logical Operations). This monitor program is contained in MCU ROM. The debugging/evaluation operation allows the user to debug user code under control of the BUFFALO monitor program.

C.2.1 EVBU Features

- A low-cost means of debugging user assembled code and evaluating MC68HC11A8, E9, 7E9, and 8E2 MCU devices
- BUFFALO monitor in ROM or PCBUG11 software working through boot mode
- One-line assembler/disassembler
- Program downloading capability
- M68HC11-based debugging/evaluating circuitry
- RS-232C compatible terminal I/O port
- Wire-wrap area for custom interfacing
- Single (+5 Vdc) input power source requirements

C.3 M68HC11EVM — Evaluation Module

Refer to *M68HC11EVM Evaluation Module User's Manual* (M68HC11EVM/AD7). The EVM is a tool for designing, debugging, and evaluation of M68HC11 MCU-based target system equipment. By providing MCU timing and I/O circuitry, the EVM simplifies user evaluation of the prototype hardware/software product. The EVM requires a user-supplied power supply and an RS-232C compatible terminal for operation. The EVM evaluates both single-chip and expanded-multiplexed modes of operation. Generating, executing, and debugging target system MCU code is accomplished in either mode.

C.3.1 EVM Features

- Low cost means of evaluating target systems with M68HC11 A- and E-series devices
- Monitor/debugger firmware
- One-line assembler/disassembler
- Host computer download capability
- Dual 64 Kbyte memory maps:
  - — 16 Kbyte monitor EPROM
  - — 8 Kbyte/16 Kbyte user pseudo-ROM
- EEPROM MCU programmer
- MCU (single-chip/expanded-multiplexed mode) extension I/O ports
- RS-232C terminal and host computer I/O ports

C

C.4 MMDS11 — Modular Development System

The M68MMDS11 Motorola Modular Development System (MMDS11) is an emulator system for developing embedded systems based on an M68HC11 microcontroller unit (MCU). The MMDS11 provides a bus state analyzer and real-time memory windows. The unit's integrated development environment includes an editor, an assembler, user interface, and source-level debugging capability. These features significantly reduce the time necessary to develop and debug an embedded control system. The unit's compact size requires a minimum of bench space. The MMDS11 is one component of Motorola's modular approach to MCU-based product development. This modular approach allows easy configuration of the MMDS11 to fit a wide range of requirements. It also reduces development system cost by allowing the user to purchase only the modular components necessary to support the particular MCU device family.

The station module is a metal enclosure that contains a printed circuit board (the control board), a test emulator module (TEM), and an internal power supply. A power cable, an RS-232 serial cable, two logic clip cables (with clips), and a 9- to 25-pin RS-232 adapter come with the MMDS11. The system requires an IBM-PC (or compatible) host computer. Connection to a target system is via a separately purchased M68HC11 emulator module (EM) and target cable. The EM completes MMDS11 functionality for a particular MCU derivative or MCU family. The many EMs available let the system emulate a variety of different M68HC11 MCUs. Connection to a target system is made with a separately purchased cable (with the appropriate connector) which connects the EM to the target system. MMDS11 features include:

C.4.1 MMDS11Features

- Real-time, non-intrusive, in-circuit emulation at the MCU's operating frequency
- Real-time bus state analyzer
  - — 8K x 64 real-time trace buffer
  - — Display of real-time trace data as raw data, disassembled instructions, raw data and disassembled instructions, or assembly-language source code
  - — Four hardware triggers for commencing trace and to provide breakpoints
  - — Nine triggering modes
  - — As many as 8190 pre- or post-trigger points for trace data
  - — 16 general-purpose logic clips, four of which can be used to trigger the bus state analyzer sequencer
  - — 16-bit time tag, or an optional 24-bit time tag that reduces the logic clips traced from sixteen to eight
- Four data breakpoints (hardware breakpoints)
- Hardware instruction breakpoints over either the 64 Kbyte M68HC11 memory map, or over a 1MB bank-switched memory map
- Thirty-two real-time variables, nine of which can be displayed in the variables window. These variables may be read or written while the MCU is running
- Thirty-two bytes of real-time memory can be displayed in the memory window. This memory may be read or written while the MCU is running
- 64 Kbytes of fast emulation memory (SRAM)
- Target input/output connections are current limited
- Six software-selectable oscillator clock sources: five internally generated frequencies and an external frequency via a bus analyzer logic clip
- Command and response logging to MS DOS disk files to save session history
- SCRIPT command for automatic execution of a sequence of MMDS11 commands
- Assembly or C-language source-level debugging with global variable viewing
- Host/emulator communications speeds as high as 57600 baud for quick program loading
- Extensive on-line MCU information via the CHIPINFO command. View memory-map, vectors, register, and pin-out information pertaining to the device being emulated
- Host software supports
  - — An editor
  - — An assembler, user interface, and source-level debugging capability
  - — Bus state analysis
  - — IBM mouse

C

C

SUMMARY OF CHANGES

This is a revision with complete reprint. All known errors in the publication have been corrected. The following summary lists significant changes. References to parts no longer manufactured have been removed throughout the document and are not noted individually. Typographical errors which do not effect content have not been noted.

Section 5 Serial Communications Interface (SCI)

Page 5-10	12 MHz references added to table 5-3 (E = 3 MHz).
Page 5-11	4800 Baud references added to table 5-4.

Section 8 Programmable Timer, Real Time Interrupt, and Pulse Accumulator

Page 8-8	12 MHz references added to table 8-1.

S

Section 9 Resets, Interrupts, And Low Power Modes

Page 9-5	12 MHz references added to table 9-1 (E = 3 MHz).

Appendix A Electrical Characteristics

Page A-1 to A-26	Appendix A updated to include low-voltage characteristics.

Appendix B Mechanical Data and Ordering Information

Page B-4 to B-6	Case Outlines removed.
Page B-8	Figure B-7 has been replaced.

S