HC11

M68HC11 N SERIES

TECHNICAL
DATA

M68HC11 N SERIES
HCMOS MICROCONTROLLER UNITS

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

Section 4
Operating Modes and On-Chip Memory

TABLE OF CONTENTS
(Continued)

Section 5
Resets and Interrupts

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Continued)

Section 10
Analog-to-Digital Converter

TABLE OF CONTENTS
(Continued)

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS
(Continued)

LIST OF TABLES

LIST OF TABLES
(Continued)

SECTION 1
INTRODUCTION

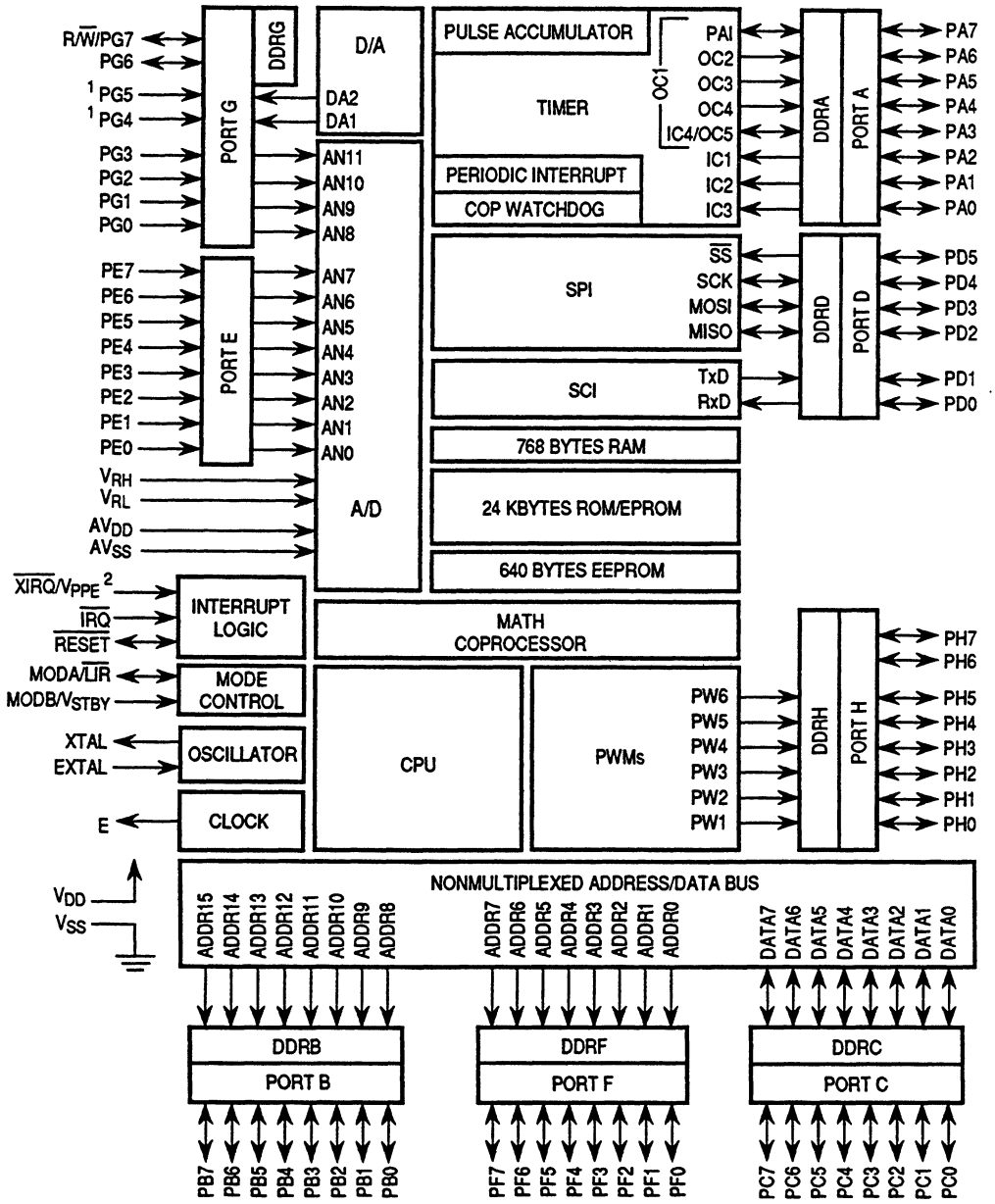
The M68HC11 N-series high-performance microcontroller units (MCUs) are enhanced derivatives of the MC68HC11F1 and include many additional features. These MCUs, with a nonmultiplexed expanded bus, are characterized by high speed and low power consumption. The fully static design allows operation at frequencies from 4 MHz to dc.

This technical data contains information concerning standard and custom-ROM devices. Standard devices have EPROM replacing ROM (MC68HC711N4). Custom-ROM devices have a ROM array that is programmed at the factory to customer specifications. Refer to **APPENDIX B MECHANICAL DATA AND ORDERING INFORMATION** for complete part numbers.

In this technical data, ROM/EPROM refers to ROM for ROM-based devices and refers to EPROM for EPROM-based devices.

1.1 Features

- M68HC11 Central Processing Unit (CPU)
- On-Chip 16-Bit Math Coprocessor
- Power Saving STOP and WAIT Modes
- 640 Bytes Electrically Erasable Programmable ROM (EEPROM)
- 768 Bytes RAM (All Saved During Standby)
- 24 Kbytes of ROM/EPROM
- PROG Mode Allows Use of Standard EPROM Programmer (27256 Footprint)
- Nonmultiplexed Address and Data Buses
- Enhanced 16-Bit Timer
  - — Three Input Capture (IC) Channels
  - — Four Output Compare (OC) Channels
  - — One Additional Channel, Selectable as Fourth IC or Fifth OC
- 8-Bit Pulse Accumulator
- Four 8-Bit Pulse-Width Modulation (PWM) Timer Channels Configurable as Two 16-Bit Channels
- Two 12-Bit PWM Timer Channels
- Real-Time Interrupt Circuit
- Computer Operating Properly (COP) Watchdog
- Enhanced Asynchronous Nonreturn to Zero (NRZ) Serial Communications Interface (SCI)
- Enhanced Synchronous Serial Peripheral Interface (SPI)
- Twelve-Channel 8-Bit Analog-to-Digital (A/D) Converter
- Two-Channel 8-Bit Digital-to-Analog (D/A) Converter
- Available in 80-Pin Plastic QFP (ROM/OTPROM) and 80-Pin Windowed Ceramic QFP (EPROM)



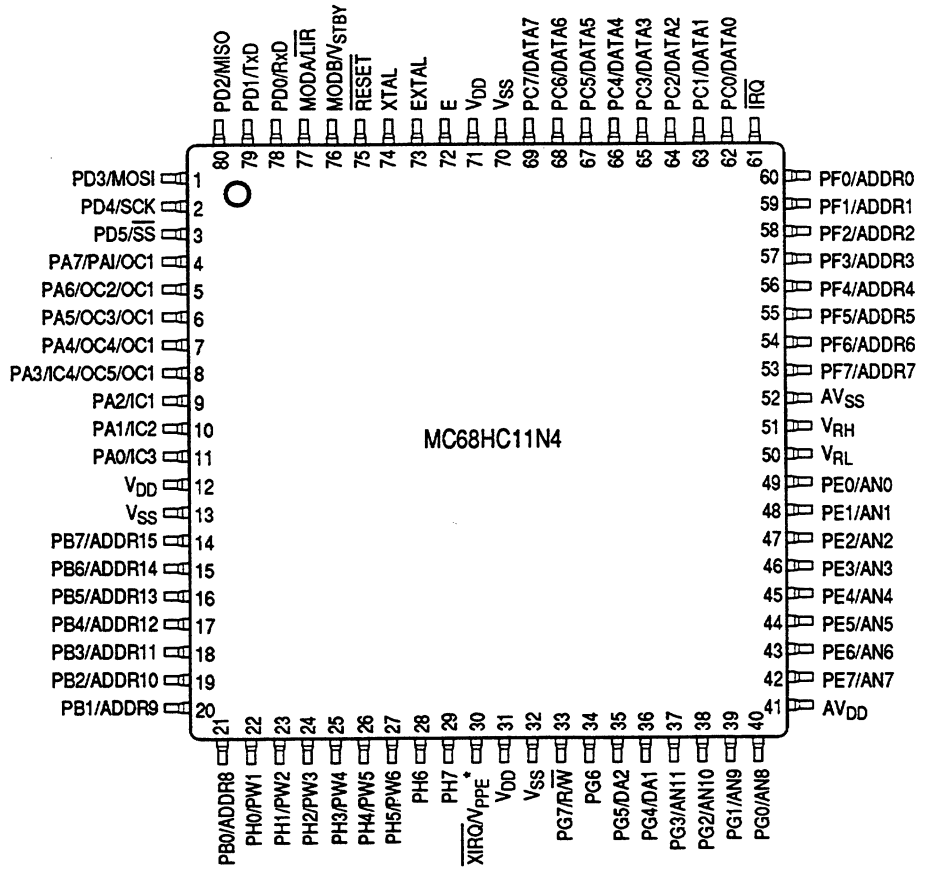
NOTES:
1. PG[5:4] ARE INPUT ONLY UNLESS AN ASSOCIATED D/A CHANNEL IS ENABLED, IN WHICH CASE THE ASSOCIATED PIN IS FORCED TO BE A D/A CONVERTER OUTPUT.
2. V_{PPE} APPLIES ONLY TO DEVICES WITH EPROM/OTPROM.

N4 BLOCK

Figure 1–1. MC68HC11N4 Block Diagram

SECTION 2
PIN DESCRIPTIONS

The MC68HC11N4 MCU is available in an 80-pin plastic quad flat pack (QFP). The MC68HC711N4 MCU is available in an 80-pin windowed ceramic QFP (EPROM) and an 80-pin plastic QFP (OTPROM). Most pins on these MCUs serve two or more functions, as described in the following paragraphs. Figure 2–1 shows the pin assignments for the plastic/ceramic QFP.



* V_{PPE} APPLIES ONLY TO DEVICES WITH EPROM/OTPROM.

N4 80-PIN QFP

Figure 2–1. Pin Assignments for M68HC11 N-Series 80-Pin QFP

2.1 V_{DD} and V_{SS}

Power is supplied to the MCU through V_{DD} and V_{SS}. V_{DD} is the power supply, and V_{SS} is ground. The MCU operates from a single 5-volt (nominal) power supply. Very fast signal transitions occur on the MCU pins. The short rise and fall times place high, short duration current demands on the power supply. To prevent noise problems, provide good power-supply bypassing at the MCU. Also, use bypass capacitors that have good high-frequency characteristics and situate them as close to the MCU as possible. Bypass requirements vary, depending on how heavily the MCU pins are loaded.

The M68HC11 N-series MCUs have four V_{DD} pins and four V_{SS} pins. One set of these pins (AV_{DD} and AV_{SS}) supplies power to the A/D and D/A converters.

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

An active low bidirectional control signal, $\overline{\text{RESET}}$, acts as an input to initialize the MCU to a known startup state. It also acts as an open-drain output to indicate that an internal failure has been detected in either the clock monitor or COP watchdog circuit. The CPU distinguishes between internal and external reset conditions by sensing whether the reset pin rises to a logic one in less than two E-clock cycles after a reset has occurred. It is not advisable to connect an external resistor-capacitor (RC) power-up delay circuit to the reset pin of M68HC11 devices because the circuit charge time constant can cause the device to misinterpret the type of reset that occurred. Refer to **SECTION 5 RESETS AND INTERRUPTS** for further information.

Figure 2–2 illustrates a reset circuit that uses an external switch. Other circuits can be used, however, it is important to incorporate a low voltage interrupt (LVI) circuit to prevent power transitions or corruption of RAM or EEPROM.

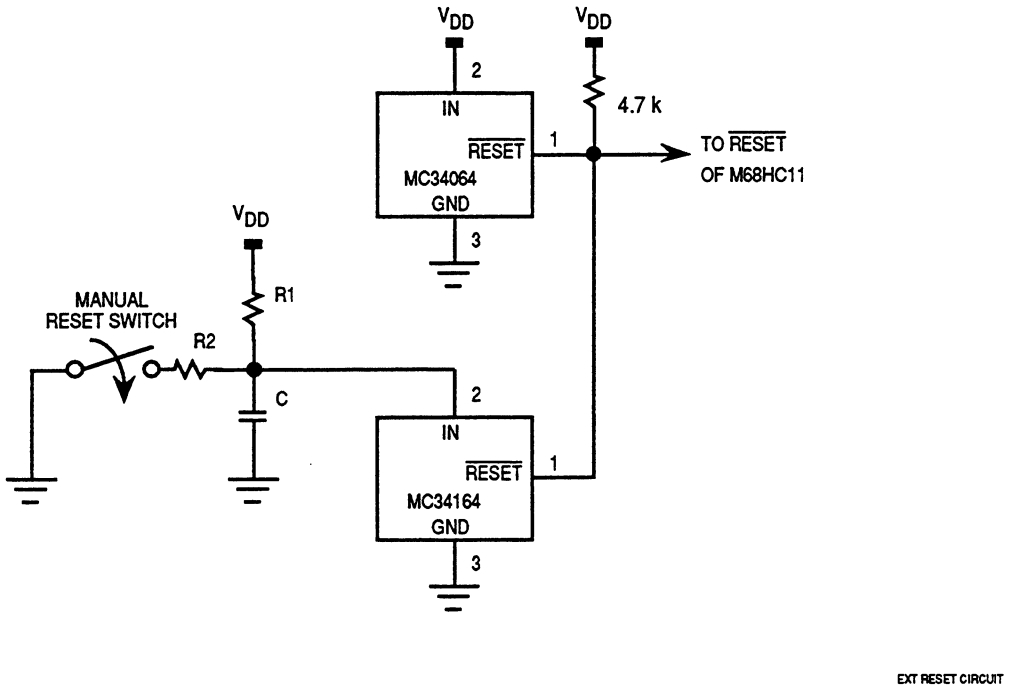


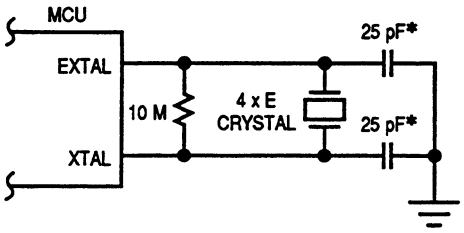
Figure 2–2. External Reset Circuit

2.3 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 is four times higher than the desired E-clock rate.

The XTAL pin is normally left unterminated when an external CMOS compatible clock input is connected to the EXTAL pin. However, a 10 kΩ to 100 kΩ load resistor connected from XTAL to ground can be used to reduce RFI noise emission. The XTAL output is normally intended to drive only a crystal. The XTAL output can be buffered with a high-impedance buffer, or it can be used to drive the EXTAL input of another M68HC11 device.

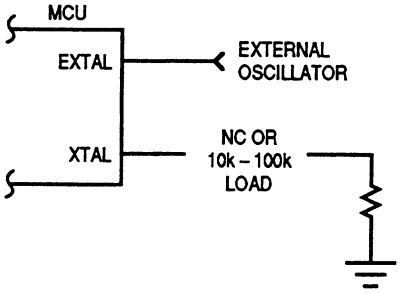
In all cases, use caution when designing circuitry associated with the oscillator pins. Load capacitances shown in the oscillator circuits include all stray layout capacitances. Refer to Figures 2–3, 2–4, and 2–5.



* VALUES INCLUDE ALL STRAY CAPACITANCES.

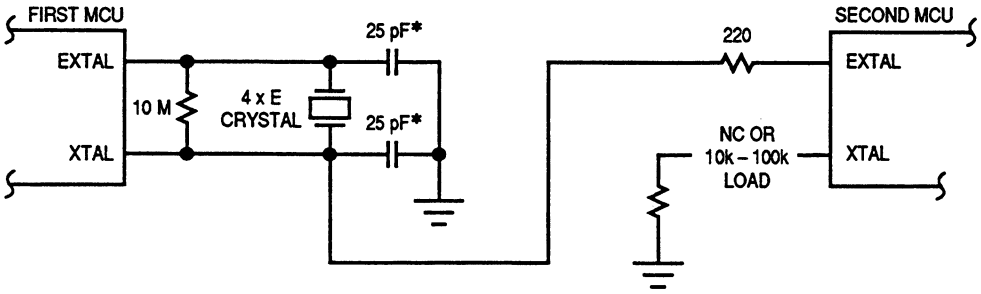
COMMON XTAL CONN

Figure 2–3. Common Crystal Connections



EXT OSC XTAL CONN

Figure 2–4. External Oscillator Connections



* VALUES INCLUDE ALL STRAY CAPACITANCES.

DUAL-MCU XTAL CONN

Figure 2–5. One Crystal Driving Two MCUs

2.4 E-Clock Output (E)

E is the output connection for the internally generated E clock. The signal from E is used as a timing reference. The frequency of the E-clock output is one fourth that of the input frequency at the XTAL and EXTAL pins. When E-clock output is low, an internal process is taking place. When it is high, data is being accessed. All clocks, including the E clock, are halted when the MCU is in STOP mode. The E clock can be turned off in single-chip modes to reduce the effects of radio frequency interference (RFI). Refer to **SECTION 9 TIMING SYSTEM**.

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

The $\overline{IRQ}$ input provides a means of applying asynchronous interrupt requests to the MCU. Either negative edge-sensitive triggering or level-sensitive triggering is program selectable (OPTION register). $\overline{IRQ}$ is always configured to level-sensitive triggering at reset. Connect an external pull-up resistor, typically 4.7 kΩ, to V_{DD} when $\overline{IRQ}$ is used in a level-sensitive wired-OR configuration. Refer to **SECTION 5 RESETS AND INTERRUPTS**.

2.6 Nonmaskable Interrupt ($\overline{XIRQ}$)/V_{PPE}

The $\overline{XIRQ}$ input provides a means of requesting a nonmaskable interrupt after reset initialization. During reset, the X bit in the condition code register (CCR) is set and any interrupt is masked until MCU software enables it. Because the $\overline{XIRQ}$ input is level sensitive, it can be connected to a multiple-source wired-OR network with an external pull-up resistor to V_{DD}. $\overline{XIRQ}$ is often used as a power loss detect interrupt.

Whenever $\overline{XIRQ}$ or $\overline{IRQ}$ are used with multiple interrupt sources ($\overline{IRQ}$ must be configured for level-sensitive operation if there is more than one source of $\overline{IRQ}$ interrupt), each source must drive the interrupt input with an open-drain type of driver to avoid contention between outputs. There should be a single pull-up resistor near the MCU interrupt input pin (typically 4.7 kΩ). There must also be an interlock mechanism at each interrupt source so that the source holds the interrupt line low until the MCU recognizes and acknowledges the interrupt request. If one or more interrupt sources are still pending after the MCU services a request, the interrupt line will still be held low and the MCU will be interrupted again as soon as the interrupt mask bit in the MCU is cleared (normally upon return from an interrupt). Refer to **SECTION 5 RESETS AND INTERRUPTS**.

On devices with EPROM, the V_{PPE} pin is used to input an external programming voltage (12.25 V typically) that must be present during EPROM programming.

2.7 MODA and MODB (MODA/$\overline{LIR}$ and MODB/V_{STBY})

During reset, MODA and MODB select one of the four operating modes. Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY**.

After the operating mode has been selected, the $\overline{LIR}$ pin provides an open-drain output to indicate that execution of an instruction has begun. The LIR pin is normally configured for wired-OR operation (only pulls low). In order to detect consecutive instructions in a high-speed application, this signal can be made to drive high for a short time to prevent false triggering. A series of E-clock cycles occurs during execution of each instruction. The $\overline{LIR}$ signal goes low during the first E-clock cycle of each instruction (opcode fetch). This output is provided for assistance in program debugging and its operation is controlled by the LIRDV bit in the OPT2 register.

The V_{STBY} pin is used to input RAM standby power. The MCU is powered from the V_{DD} signal unless the difference between the level of V_{STBY} and V_{DD} is greater than one MOS threshold (about 0.7 volts). When these voltages differ by more than 0.7 volts, the internal 768-byte RAM and part of the reset logic are powered from V_{STBY} rather than V_{DD}. 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.

2.8 AV_{DD} and AV_{SS}

The AV_{DD} and AV_{SS} pins provide the operating voltage for the A/D and D/A converter. They allow the supply voltage to the A/D and D/A converters to be bypassed independently.

2.9 V_{RH} and V_{RL}

These pins provide the reference voltage for the analog-to-digital converter. Bypass capacitors should be used to minimize noise on these signals. Any noise on V_{RH} and V_{RL} will directly affect A/D accuracy.

2.10 PG7/R/$\overline{W}$

This pin provides two separate functions, depending on the operating mode. In single-chip and bootstrap modes, PG7/R/$\overline{W}$ acts as input/output port G bit 7. Refer to **SECTION 6 PARALLEL INPUT/OUTPUT** for further information.

In expanded and test modes, PG7/R/$\overline{W}$ performs the read/write function. PG7/R/$\overline{W}$ controls the direction of transfers on the external data bus. A high on this pin indicates that a read cycle is in progress.

2.11 Port Signals

For the M68HC11 N-series MCUs, 62 pins are arranged into seven 8-bit ports: A, B, C, E, F, G, and H, and one 6-bit port (D). Each of these eight ports serves a purpose other than I/O, depending on the operating mode or peripheral functions selected. Note that ports B, C, F, and one bit of port G are available for I/O functions only in single-chip and bootstrap modes. The lines of ports A, B, C, D, F, and H are fully bidirectional. Only two lines on port G are bidirectional. Refer to Table 2–1 for details about the 62 port signals' functions within different operating modes.

Table 2–1. Port Signal Functions

Port/Bit	Single-Chip and Bootstrap Mode	Expanded and Special Test Mode
PA0	PA0/IC3	
PA1	PA1/IC2	
PA2	PA2/IC1	
PA3	PA3/OC5/IC4/OC1	
PA4	PA4/OC4/OC1	
PA5	PA5/OC3/OC1	
PA6	PA6/OC2/OC1	
PA7	PA7/PAI/OC1	
PB[7:0]	PB[7:0]	ADDR[15:8]
PC[7:0]	PC[7:0]	DATA[7:0]
PD0	PD0/RxD	
PD1	PD1/TxD	
PD2	PD2/MISO	
PD3	PD3/MOSI	
PD4	PD4/SCK	
PD5	PD5/$\overline{SS}$	
PE[7:0]	PE[7:0]/AN[7:0]	
PF[7:0]	PF[7:0]	ADDR[7:0]
PG0	PG0/AN8	
PG1	PG1/AN9	
PG2	PG2/AN10	
PG3	PG3/AN11	
PG4	PG4/DA1	
PG5	PG5/DA2	
PG6	PG6	
PG7	PG7	PG7/R/$\overline{W}$
PH0	PH0/PW1	
PH1	PH1/PW2	
PH2	PH2/PW3	
PH3	PH3/PW4	
PH4	PH4/PW5	
PH5	PH5/PW6	
PH6	PH6	
PH7	PH7	

2.11.1 Port A

Port A is an 8-bit general-purpose I/O port with a data register (PORTA) and a data direction register (DDRA). Port A pins share functions with the 16-bit timer system. PORTA can be read at any time. Inputs return the pin level; outputs return the pin driver input level. If written, PORTA stores the data in internal latches. It drives the pins only if they are configured as outputs. Writes to PORTA do not change the pin state when the pins are configured for timer output compares.

Out of reset, port A pins [7:0] are general-purpose high-impedance inputs. When the timer functions associated with these pins are disabled, the bits in DDRA govern the I/O state of the associated pin. For further information, refer to **SECTION 6 PARALLEL INPUT/OUTPUT**.

NOTE

When using the information about port functions, do not confuse pin function with the electrical state of the pin at reset. All general-purpose I/O pins configured as inputs at reset are in a high-impedance state. Port data registers reflect the functional state of the port at reset. The pin function is mode dependent.

2.11.2 Port B

Port B is an 8-bit general-purpose I/O port with a data register (PORTB) and a data direction register (DDRB). In single-chip mode, port B pins are general-purpose I/O pins (PB[7:0]). In expanded mode, port B pins act as the high-order address lines (ADDR[15:8]) of the address bus.

PORTB can be read at any time. Inputs return the pin level; outputs return the pin driver input level. If PORTB is written, the data is stored in internal latches. It drives the pins only if they are configured as outputs in single-chip or bootstrap mode.

Port B pins include on-chip pull-up devices. BPPUE bit in PPAR register enables or disables port B pull-up devices. In expanded and test modes, pull-up devices have no effect on port B pins because port B is the high-order address bus in these modes.

Refer to **SECTION 6 PARALLEL INPUT/OUTPUT**.

2.11.3 Port C

Port C is an 8-bit general-purpose I/O port with a data register (PORTC) and a data direction register (DDRC). In single-chip mode, port C pins are general-purpose I/O pins (PC[7:0]). In the expanded mode, port C pins are configured as data bus pins (DATA[7:0]).

PORTC can be read at any time. Inputs return the pin level; outputs return the pin driver input level. If PORTC is written, the data is stored in internal latches. It drives the pins only if they are configured as outputs in single-chip or bootstrap mode. Port C pins are general-purpose inputs out of reset in single-chip and bootstrap modes. In expanded and test modes, these pins are data bus lines out of reset.

The CWOM control bit in the OPT2 register disables port C's P-channel output drivers. Because the N-channel driver is not affected by CWOM, setting CWOM causes port C to become an open-drain-type output port suitable for wired-OR operation. In wired-OR mode, (PORTC bits are at logic level zero), pins are actively driven low by the N-channel driver. When a port C bit is at logic level one, the associated pin is in a high-impedance state, as neither the N-channel nor the P-channel devices are active. It is customary to have an external pull-up resistor on lines that are driven by open-drain devices. Port C can only be configured for wired-OR operation when the MCU is in single-chip mode. Refer to **SECTION 6 PARALLEL INPUT/OUTPUT** for additional information about port C functions.

2.11.4 Port D

Port D, a 6-bit general-purpose I/O port, has a data register (PORTD) and a data direction register (DDRD). The six port D lines (D[5:0]) can be used for general-purpose I/O, for the serial communications interface (SCI) and serial peripheral interface (SPI) subsystems.

PORTD can be read at any time. Inputs return the pin level; outputs return the pin driver input level. If PORTD is written, the data is stored in internal latches and can be driven only if port D is configured for general-purpose output.

Refer to **SECTION 6 PARALLEL INPUT/OUTPUT, SECTION 7 SERIAL COMMUNICATIONS INTERFACE**, and **SECTION 8 SERIAL PERIPHERAL INTERFACE** for further information on port D.

2.11.5 Port E

Port E, PE/AN[7:0], is an input-only port that is also used as the analog input port for the analog-to-digital converter.

Refer to **SECTION 10 ANALOG-TO-DIGITAL CONVERTER**.

2.11.6 Port F

Port F is an 8-bit general-purpose I/O port with a data register (PORTF) and a data direction register (DDRF). In single-chip mode, port F pins are general-purpose I/O pins (PF[7:0]). In expanded mode, port F pins act as the low-order address lines (ADDR[7:0]) of the address bus.

PORTF can be read at any time. Inputs return the pin level; outputs return the pin driver input level. If PORTF is written, the data is stored in internal latches. It drives the pins only if they are configured as outputs in single-chip or bootstrap mode.

Port F pins include on-chip pull-up devices. FPPUE bit in PPAR register enables or disables port F pull-up devices. In expanded and test modes, pull-up devices have no effect on port F pins because port F is the low-order address bus in these modes.

Refer to **SECTION 6 PARALLEL INPUT/OUTPUT**.

2.11.7 Port G

Port G bits [3:0] are input only or analog inputs (AN[11:8]) to the A/D converter. Port G bits [5:4] are input only unless an associated D/A channel is enabled, in which case bits [5:4] are analog outputs from the D/A converter. Port G bits [7:6] are bidirectional and have corresponding DDR bits (DDRG[7:6]). Bit 6 is always general-purpose input/output. Port G bit 7 is the R/$\overline{W}$ signal in expanded and test modes.

PORTG can be read at any time. Inputs return the pin level; outputs return the pin driver input level. If PORTG is written, the data is stored in internal latches. It drives the pins only if they are configured as outputs.

In single-chip and bootstrap modes, port G pins reset to high-impedance inputs. In expanded and test modes, port G pins reset to high-impedance inputs except bit 7 which is an output performing the R/$\overline{W}$ function.

Port G pins 6 and 7 include on-chip pull-up devices. GPPUE bit in PPAR register enables or disables port G pull-up devices.

Refer to **SECTION 6 PARALLEL INPUT/OUTPUT**.

2.11.8 Port H

Port H is an 8-bit general-purpose I/O port with a data register (PORTH) and a data direction register (DDRH). Port H pins support either input/output or pulse-width modulation functions. Bits [7:6] are input/output and bits [5:0] are available for input/output or serve as the pulse-width modulation timer outputs.

PORTH can be read at any time. Inputs return the pin level; outputs return the pin driver input level. If PORTH is written, the data is stored in internal latches. It drives the pins only if they are configured as outputs in single-chip or bootstrap mode.

Port H pins include on-chip pull-up devices. HPPUE bit in PPAR register enables or disables port H pull-up devices.

Refer to **SECTION 6 PARALLEL INPUT/OUTPUT**.

SECTION 3
CENTRAL PROCESSING UNIT

Section 3 presents information on M68HC11 central processing unit (CPU) architecture. Data types, addressing modes, the instruction set, and the extended addressing range required to support this MCU's memory expansion feature are also included, as are special operations such as subroutine calls and interrupts.

The CPU is designed to treat all peripheral, I/O, and memory locations identically as addresses in the 64 Kbyte memory map. This is referred to as memory-mapped I/O. There are no special instructions for I/O that are separate from those used for memory. This architecture also allows accessing an operand from an external memory location with no execution-time penalty.

3.1 CPU Registers

M68HC11 CPU registers are an integral part of the CPU and are not addressed as if they were memory locations. The seven registers, discussed in the following paragraphs, are shown in Figure 3–1.

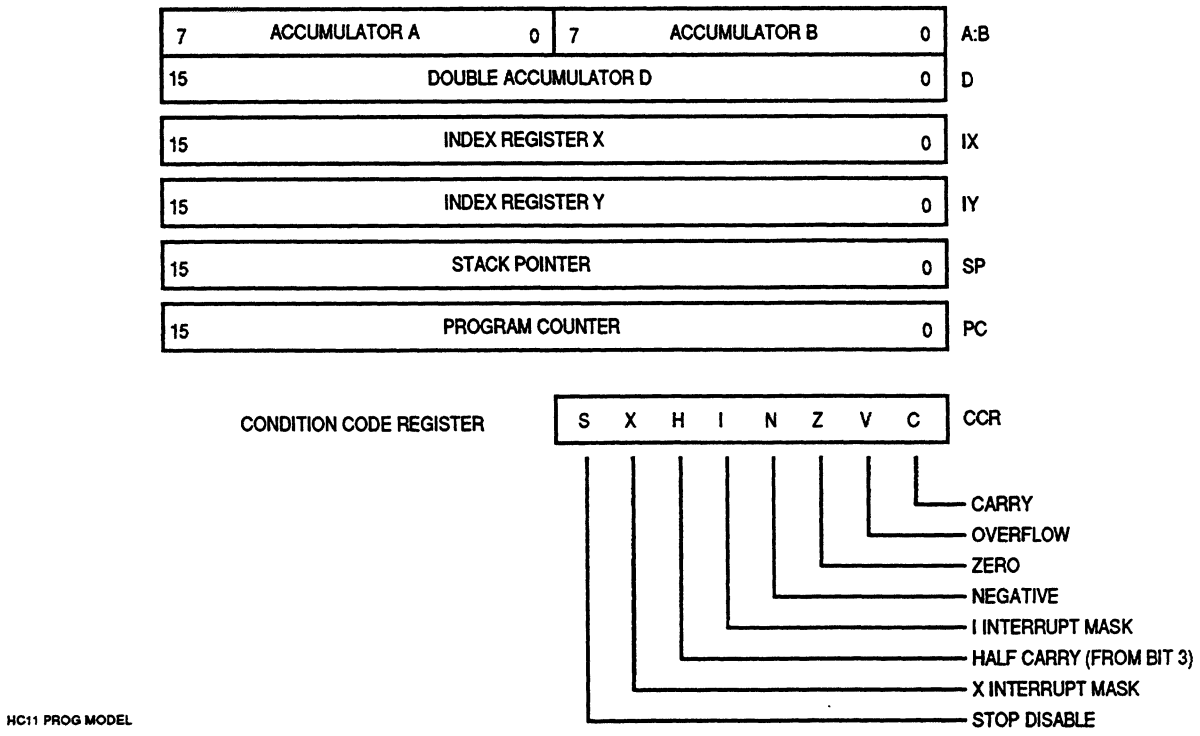


Figure 3–1. Programming Model

3.1.1 Accumulators A, B, and D

Accumulators A and B are general-purpose 8-bit registers that hold operands and results of arithmetic calculations or data manipulations. For some instructions, these two accumulators are treated as a single double-byte (16-bit) accumulator called accumulator D. Although most operations can use accumulators A or B interchangeably, the following exceptions apply:

The ABX and ABY instructions add the contents of 8-bit accumulator B to the contents of 16-bit register X or Y, but there are no equivalent instructions that use A instead of B.

The TAP and TPA instructions transfer data from accumulator A to the condition code register, or from the condition code register to accumulator A, however, there are no equivalent instructions that use B rather than A.

The decimal adjust accumulator A (DAA) instruction is used after binary-coded decimal (BCD) arithmetic operations, but there is no equivalent BCD instruction to adjust accumulator B.

The add, subtract, and compare instructions associated with both A and B (ABA, SBA, and CBA) only operate in one direction, making it important to plan ahead to ensure that the correct operand is in the correct accumulator.

3.1.2 Index Register X (IX)

The IX register provides a 16-bit indexing value that can be added to the 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.

3.1.3 Index Register Y (IY)

The 16-bit IY register performs an indexed mode function similar to that of the IX register. However, most instructions using the IY register require an extra byte of machine code and an extra cycle of execution time because of the way the opcode map is implemented. Refer to **3.3 Opcodes and Operands** for further information.

3.1.4 Stack Pointer (SP)

The M68HC11 CPU has an automatic program stack. This stack can be located anywhere in the address space and can be any size up to the amount of memory available in the system. Normally the SP is initialized by one of the first instructions in an application program. The stack is configured as a data structure that grows downward from high memory to low memory. Each time a new byte is pushed onto the stack, the SP is decremented. Each time a byte is pulled from the stack, the SP is incremented. At any given time, the SP holds the 16-bit address of the next free location in the stack. Figure 3–2 is a summary of SP operations.

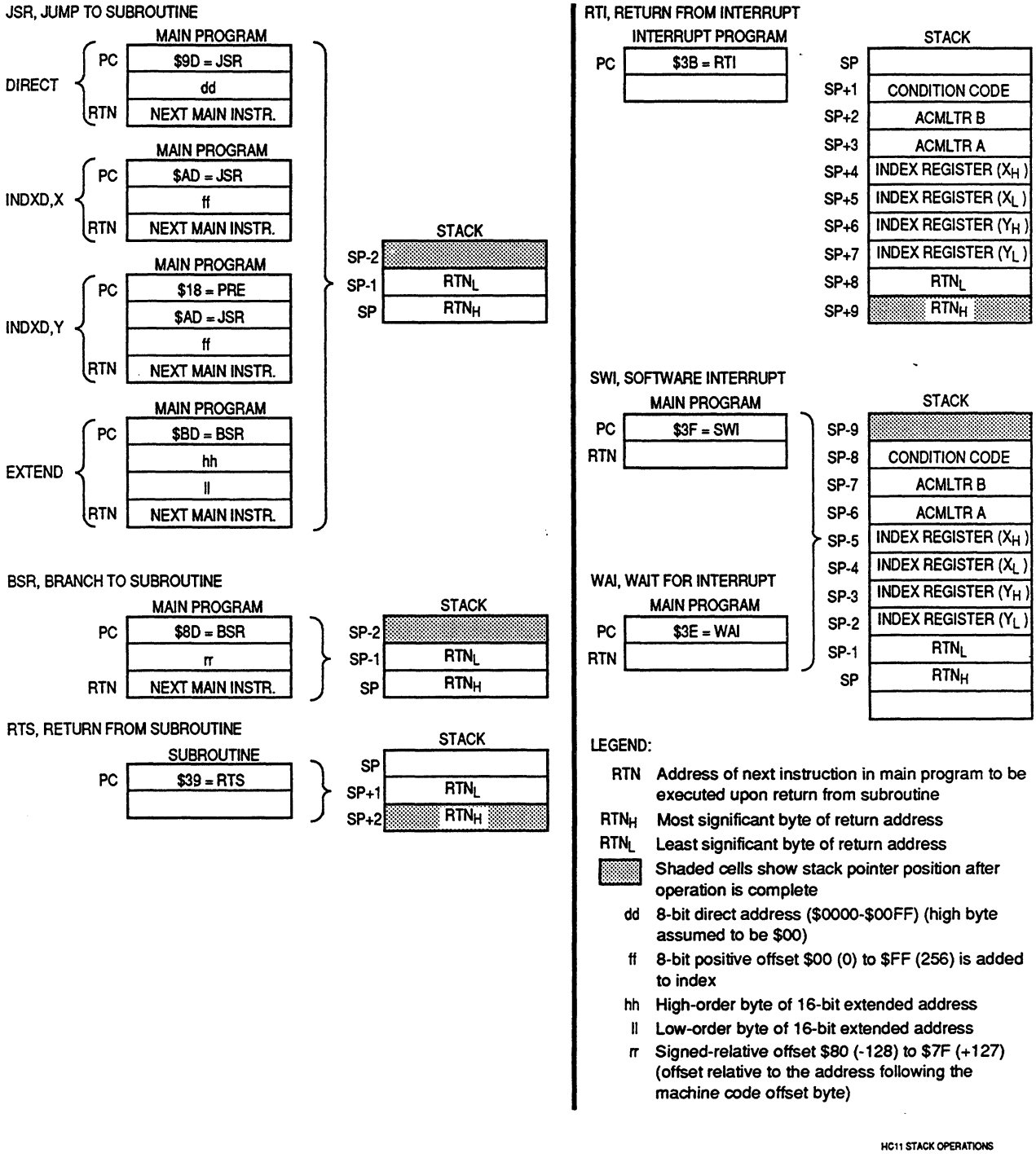


Figure 3–2. Stacking Operations

When a subroutine is called by a jump to subroutine (JSR) or branch to subroutine (BSR) instruction, the address of the instruction after the JSR or BSR is automatically pushed onto the stack, least significant byte first. When the subroutine is finished, a return from subroutine (RTS) instruction is executed. The RTS pulls the previously stacked return address from the stack, and loads it into the program counter. Execution then continues at this recovered return address.

When an interrupt is recognized, the current instruction finishes normally, the return address (the current value in the program counter) is pushed onto the stack, all of the CPU registers are pushed onto the stack, and execution continues at the address specified by the vector for the interrupt. At the end of the interrupt service routine, an RTI instruction is executed. The RTI instruction causes the saved registers to be pulled off the stack in reverse order. Program execution resumes at the return address.

There are instructions that push and pull the A and B accumulators and the X and Y index registers. These instructions are often used to preserve program context. For example, pushing accumulator A onto the stack when entering a subroutine that uses accumulator A, and then pulling accumulator A off the stack just before leaving the subroutine, ensures that the contents of a register will be the same after returning from the subroutine as it was before starting the subroutine.

3.1.5 Program Counter (PC)

The program counter, a 16-bit register, contains the address of the next instruction to be executed. After reset, the program counter is initialized from one of six possible vectors, depending on operating mode and the cause of reset.

Table 3–1. Reset Vector Comparison

	POR or $\overline{\text{RESET}}$ Pin	Clock Monitor	COP Watchdog
Normal	$FFFE, F	$FFFC, D	$FFFA, B
Test or Boot	$BFFE, F	$BFFC, D	$BFFA, B

3.1.6 Condition Code Register (CCR)

This 8-bit register contains five condition code indicators (C, V, Z, N, and H), two interrupt masking bits, (IRQ and XIRQ) and a stop disable bit (S). In the M68HC11 CPU, condition codes are automatically updated by most instructions. For example, load accumulator A (LDAA) and store accumulator A (STAA) instructions automatically set or clear the N, Z, and V condition code

flags. Pushes, pulls, add B to X (ABX), add B to Y (ABY), and transfer/exchange instructions do not affect the condition codes. Refer to Table 3–2, which shows what condition codes are affected by a particular instruction.

3.1.6.1 Carry/Borrow (C)

The C bit is set if the arithmetic logic unit (ALU) performs a carry or borrow during an arithmetic operation. The C bit also acts as an error flag for multiply and divide operations. Shift and rotate instructions operate with and through the carry bit to facilitate multiple-word shift operations.

3.1.6.2 Overflow (V)

The overflow bit is set if an operation causes an arithmetic overflow. Otherwise, the V bit is cleared.

3.1.6.3 Zero (Z)

The Z bit is set if the result of an arithmetic, logic, or data manipulation operation is zero. Otherwise, the Z bit is cleared. Compare instructions do an internal implied subtraction and the condition codes, including Z, reflect the results of that subtraction. A few operations (INX, DEX, INY, and DEY) affect the Z bit and no other condition flags. For these operations, only = and ≠ conditions can be determined.

3.1.6.4 Negative (N)

The N bit is set if the result of an arithmetic, logic, or data manipulation operation is negative (MSB = 1). Otherwise, the N bit is cleared. A result is said to be negative if its most significant bit (MSB) is a one. A quick way to test whether the contents of a memory location has the MSB set is to load it into an accumulator and then check the status of the N bit.

3.1.6.5 Interrupt Mask (I)

The interrupt request (IRQ) mask (I bit) is a global mask that disables all maskable interrupt sources. While the I bit is set, interrupts can become pending, but the operation of the CPU continues uninterrupted until the I bit is cleared. After any reset, the I bit is set by default and can only be cleared by a software instruction. When an interrupt is recognized, the I bit is set after the registers are stacked, but before the interrupt vector is fetched. After the interrupt has been serviced, a return from interrupt instruction is normally executed, restoring the registers to the values that were present before the interrupt occurred. Normally, the I bit is zero after a return from interrupt is executed. Although the I bit can be cleared within an interrupt service routine, "nesting" interrupts in this way should only be done when there is a clear

understanding of latency and of the arbitration mechanism. Refer to **SECTION 5 RESETS AND INTERRUPTS.**

3.1.6.6 Half Carry (H)

The H bit is set 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. Half carry is used during BCD operations.

3.1.6.7 X Interrupt Mask (X)

The XIRQ mask (X) bit disables interrupts from the $\overline{\text{XIRQ}}$ pin. After any reset, X is set by default and must be cleared by a software instruction. When an $\overline{\text{XIRQ}}$ interrupt is recognized, the X and I bits are set after the registers are stacked, but before the interrupt vector is fetched. After the interrupt has been serviced, an RTI instruction is normally executed, causing the registers to be restored to the values that were present before the interrupt occurred. The X interrupt mask bit is set only by hardware ($\overline{\text{RESET}}$ or $\overline{\text{XIRQ}}$ acknowledge). X is cleared only by program instruction (TAP, where the associated bit of A is zero; or RTI, where bit 6 of the value loaded into the CCR from the stack has been cleared). There is no hardware action for clearing X.

3.1.6.8 Stop Disable (S)

Setting the STOP disable (S) bit prevents the STOP instruction from putting the M68HC11 into a low-power stop condition. If the STOP instruction is encountered by the CPU while the S bit is set, it is treated as a no-operation (NOP) instruction, and processing continues to the next instruction. S is set by reset — STOP disabled by default.

3.2 Data Types

The M68HC11 CPU supports the following data types:

- Bit data
- 8-bit and 16-bit signed and unsigned integers
- 16-bit unsigned fractions
- 16-bit addresses

A byte is eight bits wide and can be accessed at any byte location. A word is composed of two consecutive bytes with the most significant byte at the lower value address. Because the M68HC11 is an 8-bit CPU, there are no special requirements for alignment of instructions or operands.

3.3 Opcodes and Operands

The M68HC11 family of microcontrollers uses 8-bit opcodes. Each opcode identifies a particular instruction and associated addressing mode to the CPU. Several opcodes are required to provide each instruction with a range of addressing capabilities. Only 256 opcodes would be available if the range of values were restricted to the number able to be expressed in 8-bit binary numbers.

A four-page opcode map has been implemented to expand the number of instructions. An additional byte, called a prebyte, directs the processor from page 0 of the opcode map to one of the other three pages. As its name implies, the additional byte precedes the opcode.

A complete instruction consists of a prebyte, if any, an opcode, and zero, one, two, or three operands. The operands contain information the CPU needs for executing the instruction. Complete instructions can be from one to five bytes long.

3.4 Addressing Modes

Six addressing modes can be used to access memory: immediate, direct, extended, indexed, inherent, and relative. These modes are detailed in the following paragraphs. All modes except inherent mode use an effective address. The effective address is the memory address from which the argument is fetched or stored, or the address from which execution is to proceed. The effective address can be specified within an instruction, or it can be calculated.

3.4.1 Immediate

In the immediate addressing mode an argument is contained in the byte(s) immediately following the opcode. The number of bytes following the opcode matches the size of the register or memory location being operated on. There are two-, three-, and four- (if prebyte is required) byte immediate instructions. The effective address is the address of the byte following the instruction.

3.4.2 Direct

In the direct addressing mode, the low-order byte of the operand address is contained in a single byte following the opcode, and the high-order byte of the address is assumed to be $00. Addresses $00–$FF are thus accessed directly, using two-byte instructions. Execution time is reduced by eliminating the additional memory access required for the high-order address byte. In most applications, this 256-byte area is reserved for frequently referenced data. In M68HC11 MCUs, the memory map can be configured for combinations of internal registers, RAM, or external memory to occupy these addresses.

3.4.3 Extended

In the extended addressing mode, the effective address of the argument is contained in two bytes following the opcode byte. These are three-byte instructions (or four-byte instructions if a prebyte is required). One or two bytes are needed for the opcode and two for the effective address.

3.4.4 Indexed

In the indexed addressing mode, an 8-bit unsigned offset contained in the instruction is added to the value contained in an index register (IX or IY). The sum is the effective address. This addressing mode allows referencing any memory location in the 64 Kbyte address space. These are two- to five-byte instructions, depending on whether or not a prebyte is required.

3.4.5 Inherent

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations that use only the index registers or accumulators, as well as control instructions with no arguments, are included in this addressing mode. These are one- or two-byte instructions.

3.4.6 Relative

The relative addressing mode is used only for branch instructions. If the branch condition is true, an 8-bit signed offset included in the instruction 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.

3.5 Instruction Set

Refer to Table 3–2, which shows all the M68HC11 instructions in all possible addressing modes. For each instruction, the table shows the operand construction, the number of machine code bytes, and execution time in CPU E clock cycles.

Table 3–2. Instruction Set (1 of 7)

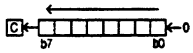
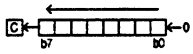
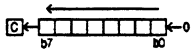
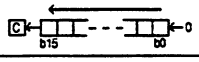
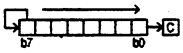
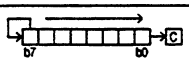
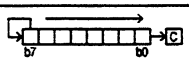
Mnemonic	Operation	Description	Addressing Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	X	H	I	N	Z	V	C
ABA	Add Accumulators	A + B ⇒ A	INH	1B	—	2	—	—	Δ	—	Δ	Δ	Δ	Δ
ABX	Add B to X	IX + (00 : B) ⇒ IX	INH	3A	—	3	—	—	—	—	—	—	—	—
ABY	Add B to Y	IY + (00 : B) ⇒ IY	INH	18 3A	—	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 3 4 4 5	—	—	Δ	—	Δ	Δ	Δ	Δ
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 3 4 4 5	—	—	Δ	—	Δ	Δ	Δ	Δ
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 3 4 4 5	—	—	Δ	—	Δ	Δ	Δ	Δ
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 3 4 4 5	—	—	Δ	—	Δ	Δ	Δ	Δ
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	4 5 6 6 7	—	—	—	—	Δ	Δ	Δ	Δ
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 3 4 4 5	—	—	—	—	Δ	Δ	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 3 4 4 5	—	—	—	—	Δ	Δ	0	—
ASL (opr)	Arithmetic Shift Left		EXT IND,X IND,Y	78 68 18 68	hh ll ff ff	6 6 7	—	—	—	—	Δ	Δ	Δ	Δ
ASLA	Arithmetic Shift Left A		A INH	48	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ASLB	Arithmetic Shift Left B		B INH	58	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ASLD	Arithmetic Shift Left D		INH	05	—	3	—	—	—	—	Δ	Δ	Δ	Δ
ASR	Arithmetic Shift Right		EXT IND,X IND,Y	77 67 18 67	hh ll ff ff	6 6 7	—	—	—	—	Δ	Δ	Δ	Δ
ASRA	Arithmetic Shift Right A		A INH	47	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ASRB	Arithmetic Shift Right B		B INH	57	—	2	—	—	—	—	Δ	Δ	Δ	Δ
BCC (rel)	Branch if Carry Clear	? C = 0	REL	24	rr	3	—	—	—	—	—	—	—	—
BCLR (opr) (msk)	Clear Bit(s)	M • ($\overline{mm}$) ⇒ M	DIR IND,X IND,Y	15 1D 18 1D	dd mm ff mm ff mm	6 7 8	—	—	—	—	Δ	Δ	0	—
BCS (rel)	Branch if Carry Set	? C = 1	REL	25	rr	3	—	—	—	—	—	—	—	—

Table 3–2. Instruction Set (2 of 7)

Mnemonic	Operation	Description	Addressing Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	X	H	I	N	Z	V	C
BEQ (rel)	Branch if = Zero	? Z = 1	REL	27	rr	3	—	—	—	—	—	—	—	—
BGE (rel)	Branch if ≥ Zero	? N ⊕ V = 0	REL	2C	rr	3	—	—	—	—	—	—	—	—
BGT (rel)	Branch if > Zero	? Z + (N ⊕ V) = 0	REL	2E	rr	3	—	—	—	—	—	—	—	—
BHI (rel)	Branch if Higher	? C + Z = 0	REL	22	rr	3	—	—	—	—	—	—	—	—
BHS (rel)	Branch if Higher or Same	? C = 0	REL	24	rr	3	—	—	—	—	—	—	—	—
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 3 4 4 5	—	—	—	—	Δ	Δ	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 3 4 4 5	—	—	—	—	Δ	Δ	0	—
BLE (rel)	Branch if ≤ Zero	? Z + (N ⊕ V) = 1	REL	2F	rr	3	—	—	—	—	—	—	—	—
BLO (rel)	Branch if Lower	? C = 1	REL	25	rr	3	—	—	—	—	—	—	—	—
BLS (rel)	Branch if Lower or Same	? C + Z = 1	REL	23	rr	3	—	—	—	—	—	—	—	—
BLT (rel)	Branch if < Zero	? N ⊕ V = 1	REL	2D	rr	3	—	—	—	—	—	—	—	—
BMI (rel)	Branch if Minus	? N = 1	REL	2B	rr	3	—	—	—	—	—	—	—	—
BNE (rel)	Branch if not = Zero	? Z = 0	REL	26	rr	3	—	—	—	—	—	—	—	—
BPL (rel)	Branch if Plus	? N = 0	REL	2A	rr	3	—	—	—	—	—	—	—	—
BRA (rel)	Branch Always	? 1 = 1	REL	20	rr	3	—	—	—	—	—	—	—	—
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	6 7 8	—	—	—	—	—	—	—	—
BRN (rel)	Branch Never	? 1 = 0	REL	21	rr	3	—	—	—	—	—	—	—	—
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	6 7 8	—	—	—	—	—	—	—	—
BSET (opr) (msk)	Set Bit(s)	M + mm ⇒ M	DIR IND,X IND,Y	14 1C 18 1C	dd mm ff mm ff mm	6 7 8	—	—	—	—	Δ	Δ	0	—
BSR (rel)	Branch to Subroutine	See Figure 3–2	REL	8D	rr	6	—	—	—	—	—	—	—	—
BVC (rel)	Branch if Overflow Clear	? V = 0	REL	28	rr	3	—	—	—	—	—	—	—	—
BVS (rel)	Branch if Overflow Set	? V = 1	REL	29	rr	3	—	—	—	—	—	—	—	—
CBA	Compare A to B	A – B	INH	11	—	2	—	—	—	—	Δ	Δ	Δ	Δ
CLC	Clear Carry Bit	0 ⇒ C	INH	0C	—	2	—	—	—	—	—	—	—	0
CLI	Clear Interrupt Mask	0 ⇒ I	INH	0E	—	2	—	—	—	0	—	—	—	—
CLR (opr)	Clear Memory Byte	0 ⇒ M	EXT IND,X IND,Y	7F 6F 18 6F	hh ll ff ff	6 6 7	—	—	—	—	0	1	0	0

Table 3–2. Instruction Set (3 of 7)

Mnemonic	Operation	Description	Addressing Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	X	H	I	N	Z	V	C
CLRA	Clear Accumulator A	0 ⇒ A	A INH	4F	—	2	—	—	—	—	0	1	0	0
CLRB	Clear Accumulator B	0 ⇒ B	B INH	5F	—	2	—	—	—	—	0	1	0	0
CLV	Clear Overflow Flag	0 ⇒ V	INH	0A	—	2	—	—	—	—	—	—	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 3 4 4 5	—	—	—	—	Δ	Δ	Δ	Δ
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 3 4 4 5	—	—	—	—	Δ	Δ	Δ	Δ
COM (opr)	Ones Complement Memory Byte	$FF – M ⇒ M	EXT IND,X IND,Y	73 63 18 63	hh ll ff ff	6 6 7	—	—	—	—	Δ	Δ	0	1
COMA	Ones Complement A	$FF – A ⇒ A	A INH	43	—	2	—	—	—	—	Δ	Δ	0	1
COMB	Ones Complement B	$FF – B ⇒ B	B INH	53	—	2	—	—	—	—	Δ	Δ	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	5 6 7 7 7	—	—	—	—	Δ	Δ	Δ	Δ
CPX (opr)	Compare X to Memory 16-Bit	IX – M : M + 1	IMM DIR EXT IND,X IND,Y	8C 9C BC AC CD AC	jj kk dd hh ll ff ff	4 5 6 6 7	—	—	—	—	Δ	Δ	Δ	Δ
CPY (opr)	Compare Y to Memory 16-Bit	IY – M : M + 1	IMM DIR EXT IND,X IND,Y	18 8C 18 9C 18 BC 1A AC 18 AC	jj kk dd hh ll ff ff	5 6 7 7 7	—	—	—	—	Δ	Δ	Δ	Δ
DAA	Decimal Adjust A	Adjust Sum to BCD	INH	19	—	2	—	—	—	—	Δ	Δ	Δ	Δ
DEC (opr)	Decrement Memory Byte	M – 1 ⇒ M	EXT IND,X IND,Y	7A 6A 18 6A	hh ll ff ff	6 6 7	—	—	—	—	Δ	Δ	Δ	—
DECA	Decrement Accumulator A	A – 1 ⇒ A	A INH	4A	—	2	—	—	—	—	Δ	Δ	Δ	—
DECB	Decrement Accumulator B	B – 1 ⇒ B	B INH	5A	—	2	—	—	—	—	Δ	Δ	Δ	—
DES	Decrement Stack Pointer	SP – 1 ⇒ SP	INH	34	—	3	—	—	—	—	—	—	—	—
DEX	Decrement Index Register X	IX – 1 ⇒ IX	INH	09	—	3	—	—	—	—	—	Δ	—	—
DEY	Decrement Index Register Y	IY – 1 ⇒ IY	INH	18 09	—	4	—	—	—	—	—	Δ	—	—
EORA (opr)	Exclusive OR A with Memory	A ⊕ M ⇒ A	A IMM A DIR A EXT A IND,X A IND,Y	88 98 B8 A8 18 A8	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	0	—
EORB (opr)	Exclusive OR B with Memory	B ⊕ M ⇒ B	B IMM B DIR B EXT B IND,X B IND,Y	C8 D8 F8 E8 18 E8	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	0	—

Table 3–2. Instruction Set (4 of 7)

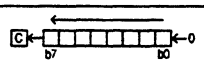
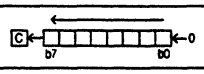
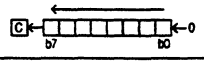
Mnemonic	Operation	Description	Addressing Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	X	H	I	N	Z	V	C
FDIV	Fractional Divide 16 by 16	D / IX ⇒ IX; r ⇒ D	INH	03	—	41	—	—	—	—	—	Δ	Δ	Δ
IDIV	Integer Divide 16 by 16	D / IX ⇒ IX; r ⇒ D	INH	02	—	41	—	—	—	—	—	Δ	0	Δ
INC (opr)	Increment Memory Byte	M + 1 ⇒ M	EXT IND,X IND,Y	7C 6C 18 6C	hh ll ff ff	6 6 7	—	—	—	—	Δ	Δ	Δ	—
INCA	Increment Accumulator A	A + 1 ⇒ A	A INH	4C	—	2	—	—	—	—	Δ	Δ	Δ	—
INCB	Increment Accumulator B	B + 1 ⇒ B	B INH	5C	—	2	—	—	—	—	Δ	Δ	Δ	—
INS	Increment Stack Pointer	SP + 1 ⇒ SP	INH	31	—	3	—	—	—	—	—	—	—	—
INX	Increment Index Register X	IX + 1 ⇒ IX	INH	08	—	3	—	—	—	—	—	Δ	—	—
INY	Increment Index Register Y	IY + 1 ⇒ IY	INH	18 08	—	4	—	—	—	—	—	Δ	—	—
JMP (opr)	Jump	See Figure 3–2	EXT IND,X IND,Y	7E 6E 18 6E	hh ll ff ff	3 3 4	—	—	—	—	—	—	—	—
JSR (opr)	Jump to Subroutine	See Figure 3–2	DIR EXT IND,X IND,Y	9D BD AD 18 AD	dd hh ll ff ff	5 6 6 7	—	—	—	—	—	—	—	—
LDAA (opr)	Load Accumulator A	M ⇒ A	A IMM A DIR A EXT A IND,X A IND,Y	86 96 B6 A6 18 A6	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	0	—
LDAB (opr)	Load Accumulator B	M ⇒ B	B IMM B DIR B EXT B IND,X B IND,Y	C6 D6 F6 E6 18 E6	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	0	—
LDD (opr)	Load Double Accumulator D	M ⇒ A,M + 1 ⇒ B	IMM DIR EXT IND,X IND,Y	CC DC FC EC 18 EC	jj kk dd hh ll ff ff	3 4 5 5 6	—	—	—	—	Δ	Δ	0	—
LDS (opr)	Load Stack Pointer	M : M + 1 ⇒ SP	IMM DIR EXT IND,X IND,Y	8E 9E BE AE 18 AE	jj kk dd hh ll ff ff	3 4 5 5 6	—	—	—	—	Δ	Δ	0	—
LDX (opr)	Load Index Register X	M : M + 1 ⇒ IX	IMM DIR EXT IND,X IND,Y	CE DE FE EE CD EE	jj kk dd hh ll ff ff	3 4 5 5 6	—	—	—	—	Δ	Δ	0	—
LDY (opr)	Load Index Register Y	M : M + 1 ⇒ IY	IMM DIR EXT IND,X IND,Y	18 CE 18 DE 18 FE 1A EE 18 EE	jj kk dd hh ll ff ff	4 5 6 6 6	—	—	—	—	Δ	Δ	0	—
LSL (opr)	Logical Shift Left		EXT IND,X IND,Y	78 68 18 68	hh ll ff ff	6 6 7	—	—	—	—	Δ	Δ	Δ	Δ
LSLA	Logical Shift Left A		A INH	48	—	2	—	—	—	—	Δ	Δ	Δ	Δ
LSLB	Logical Shift Left B		B INH	58	—	2	—	—	—	—	Δ	Δ	Δ	Δ

Table 3–2. Instruction Set (5 of 7)

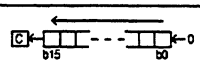
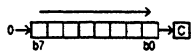
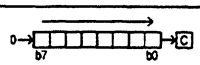
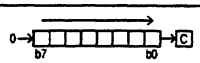
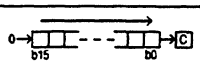
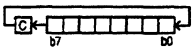
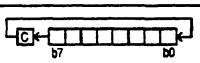
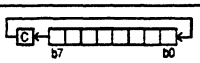
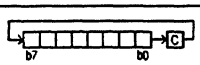
Mnemonic	Operation	Description	Addressing Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	X	H	I	N	Z	V	C
LSLD	Logical Shift Left Double		INH	05	—	3	—	—	—	—	Δ	Δ	Δ	Δ
LSR (opr)	Logical Shift Right		EXT IND,X IND,Y	74 64 18 64	hh ll ff ff	6 6 7	—	—	—	—	0	Δ	Δ	Δ
LSRA	Logical Shift Right A		A INH	44	—	2	—	—	—	—	0	Δ	Δ	Δ
LSRB	Logical Shift Right B		B INH	54	—	2	—	—	—	—	0	Δ	Δ	Δ
LSRD	Logical Shift Right Double		INH	04	—	3	—	—	—	—	0	Δ	Δ	Δ
MUL	Multiply 8 by 8	A * B ⇒ D	INH	3D	—	10	—	—	—	—	—	—	—	Δ
NEG (opr)	Two's Complement Memory Byte	0 – M ⇒ M	EXT IND,X IND,Y	70 60 18 60	hh ll ff ff	6 6 7	—	—	—	—	Δ	Δ	Δ	Δ
NEGA	Two's Complement A	0 – A ⇒ A	A INH	40	—	2	—	—	—	—	Δ	Δ	Δ	Δ
NEGB	Two's Complement B	0 – B ⇒ B	B INH	50	—	2	—	—	—	—	Δ	Δ	Δ	Δ
NOP	No operation	No Operation	INH	01	—	2	—	—	—	—	—	—	—	—
ORAA (opr)	OR Accumulator A (Inclusive)	A + M ⇒ A	A IMM A DIR A EXT A IND,X A IND,Y	8A 9A BA AA 18 AA	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	0	—
ORAB (opr)	OR Accumulator B (Inclusive)	B + M ⇒ B	B IMM B DIR B EXT B IND,X B IND,Y	CA DA FA EA 18 EA	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	0	—
PSHA	Push A onto Stack	A ⇒ Stk,SP = SP – 1	A INH	36	—	3	—	—	—	—	—	—	—	—
PSHB	Push B onto Stack	B ⇒ Stk,SP = SP – 1	B INH	37	—	3	—	—	—	—	—	—	—	—
PSHX	Push X onto Stack (Lo First)	IX ⇒ Stk,SP = SP – 2	INH	3C	—	4	—	—	—	—	—	—	—	—
PSHY	Push Y onto Stack (Lo First)	IY ⇒ Stk,SP = SP – 2	INH	18 3C	—	5	—	—	—	—	—	—	—	—
PULA	Pull A from Stack	SP = SP + 1, A ⇐ Stk	A INH	32	—	4	—	—	—	—	—	—	—	—
PULB	Pull B from Stack	SP = SP + 1, B ⇐ Stk	B INH	33	—	4	—	—	—	—	—	—	—	—
PULX	Pull X From Stack (Hi First)	SP = SP + 2, IX ⇐ Stk	INH	38	—	5	—	—	—	—	—	—	—	—
PULY	Pull Y from Stack (Hi First)	SP = SP + 2, IY ⇐ Stk	INH	18 38	—	6	—	—	—	—	—	—	—	—
ROL (opr)	Rotate Left		EXT IND,X IND,Y	79 69 18 69	hh ll ff ff	6 6 7	—	—	—	—	Δ	Δ	Δ	Δ
ROLA	Rotate Left A		A INH	49	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ROLB	Rotate Left B		B INH	59	—	2	—	—	—	—	Δ	Δ	Δ	Δ
ROR (opr)	Rotate Right		EXT IND,X IND,Y	76 66 18 66	hh ll ff ff	6 6 7	—	—	—	—	Δ	Δ	Δ	Δ

Table 3–2. Instruction Set (6 of 7)

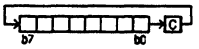
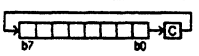
Mnemonic	Operation	Description	Addressing Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	X	H	I	N	Z	V	C
RORA	Rotate Right A		A INH	46	—	2	—	—	—	—	Δ	Δ	Δ	Δ
RORB	Rotate Right B		B INH	56	—	2	—	—	—	—	Δ	Δ	Δ	Δ
RTI	Return from Interrupt	See Figure 3–2	INH	3B	—	12	Δ	↓	Δ	Δ	Δ	Δ	Δ	Δ
RTS	Return from Subroutine	See Figure 3–2	INH	39	—	5	—	—	—	—	—	—	—	—
SBA	Subtract B from A	A – B ⇒ A	INH	10	—	2	—	—	—	—	Δ	Δ	Δ	Δ
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 3 4 4 5	—	—	—	—	Δ	Δ	Δ	Δ
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 3 4 4 5	—	—	—	—	Δ	Δ	Δ	Δ
SEC	Set Carry	1 ⇒ C	INH	0D	—	2	—	—	—	—	—	—	—	1
SEI	Set Interrupt Mask	1 ⇒ I	INH	0F	—	2	—	—	—	1	—	—	—	—
SEV	Set Overflow Flag	1 ⇒ V	INH	0B	—	2	—	—	—	—	—	—	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	3 4 4 5	—	—	—	—	Δ	Δ	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	3 4 4 5	—	—	—	—	Δ	Δ	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	4 5 5 6	—	—	—	—	Δ	Δ	0	—
STOP	Stop Internal Clocks	—	INH	CF	—	2	—	—	—	—	—	—	—	—
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	4 5 5 6	—	—	—	—	Δ	Δ	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	4 5 5 6	—	—	—	—	Δ	Δ	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	5 6 6 6	—	—	—	—	Δ	Δ	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 3 4 4 5	—	—	—	—	Δ	Δ	Δ	Δ
SUBB (opr)	Subtract Memory from B	B – M ⇒ B	A IMM A DIR A EXT A IND,X A IND,Y	C0 D0 F0 E0 18 E0	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	Δ	Δ
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	4 5 6 6 7	—	—	—	—	Δ	Δ	Δ	Δ

Table 3–2. Instruction Set (7 of 7)

Mnemonic	Operation	Description	Addressing Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	X	H	I	N	Z	V	C
SWI	Software Interrupt	See Figure 3–2	INH	3F	—	14	—	—	—	1	—	—	—	—
TAB	Transfer A to B	A ⇒ B	INH	16	—	2	—	—	—	—	Δ	Δ	0	—
TAP	Transfer A to CC Register	A ⇒ CCR	INH	06	—	2	Δ	↓	Δ	Δ	Δ	Δ	Δ	Δ
TBA	Transfer B to A	B ⇒ A	INH	17	—	2	—	—	—	—	Δ	Δ	0	—
TEST	TEST (Only in Test Modes)	Address Bus Counts	INH	00	—	*	—	—	—	—	—	—	—	—
TPA	Transfer CC Register to A	CCR ⇒ A	INH	07	—	2	—	—	—	—	—	—	—	—
TST (opr)	Test for Zero or Minus	M – 0	EXT IND,X IND,Y	7D 6D 18 6D	hh ll ff ff	6 6 7	—	—	—	—	Δ	Δ	0	0
TSTA	Test A for Zero or Minus	A – 0	A INH	4D	—	2	—	—	—	—	Δ	Δ	0	0
TSTB	Test B for Zero or Minus	B – 0	B INH	5D	—	2	—	—	—	—	Δ	Δ	0	0
TSX	Transfer Stack Pointer to X	SP + 1 ⇒ IX	INH	30	—	3	—	—	—	—	—	—	—	—
TSY	Transfer Stack Pointer to Y	SP + 1 ⇒ IY	INH	18 30	—	4	—	—	—	—	—	—	—	—
TXS	Transfer X to Stack Pointer	IX – 1 ⇒ SP	INH	35	—	3	—	—	—	—	—	—	—	—
TYS	Transfer Y to Stack Pointer	IY – 1 ⇒ SP	INH	18 35	—	4	—	—	—	—	—	—	—	—
WAI	Wait for Interrupt	Stack Regs & WAIT	INH	3E	—	**	—	—	—	—	—	—	—	—
XGDX	Exchange D with X	IX ⇒ D, D ⇒ IX	INH	8F	—	3	—	—	—	—	—	—	—	—
XGDY	Exchange D with Y	IY ⇒ D, D ⇒ IY	INH	18 8F	—	4	—	—	—	—	—	—	—	—

Cycle

* 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).

Operands

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

Operators

( ) Contents of register shown inside parentheses
⇐ Is transferred to
⇑ Is pulled from stack
⇓ Is pushed onto stack
• Boolean AND
\+ Arithmetic Addition Symbol except where used as Inclusive-OR symbol in Boolean Formula
⊕ Exclusive-OR
* Multiply
: Concatenation
– Arithmetic subtraction symbol or Negation symbol (Two's Complement)

Condition Codes

— Bit not changed
0 Bit always cleared
1 Bit always set
Δ Bit cleared or set, depending on operation
↓ Bit can be cleared, cannot become set

SECTION 4
OPERATING MODES AND ON-CHIP MEMORY

Section 4 contains information about the modes that define M68HC11 N-series operating conditions, and about the on-chip memory that allows the MCU to be configured for various applications.

4.1 Operating Modes

The values of the mode select inputs MODB and MODA during reset determine the operating mode. Single chip and expanded modes are the normal modes. In single-chip mode only on-board memory is available. Expanded mode, however, allows access to external memory or peripheral devices. Each of these two normal modes is paired with a special mode. Bootstrap mode, a variation of the single-chip mode, executes a bootloader program from an internal bootstrap ROM. Test mode allows privileged access to internal resources.

4.1.1 Single-Chip Operating Mode

In single-chip operating mode, the M68HC11 N-series MCUs have no external address or data bus. Ports B, C, F, and the R/$\overline{W}$ pin are available for general-purpose I/O.

4.1.2 Expanded Operating Mode

In expanded operating mode, the MCU can access a 64 Kbyte physical address space. The address space includes the same on-chip memory addresses used for single-chip mode, in addition to external memory and peripheral devices.

The expansion bus is made up of ports B, C, and F, and the R/$\overline{W}$ signal. In expanded mode, high order address bits are output on the port B pins, low order address bits on the port F pins, and the data bus on port C. The PG7/R/$\overline{W}$ pin controls the direction of data transfer on the port C bus.

4.1.3 Special Test Mode

Special test mode, a variation of the expanded mode, is primarily used during Motorola's internal production testing; however, it is accessible for programming the CONFIG register, programming calibration data into EEPROM, and supporting emulation and debugging during development.

4.1.4 Bootstrap Mode

When the MCU is reset in bootstrap mode, a small on-chip ROM is enabled at address $BE00–$BFFF. The ROM contains a reset vector and a bootloader program. The MCU fetches the reset vector, then executes the bootloader.

For normal use of the bootloader program, send $FF to the SCI receiver at either E clock ÷ 16, or E clock ÷ 104 (1200 baud for E clock equals 2 MHz). Then download up to 768 bytes of program data, which is put into RAM starting at $0080. These characters are echoed through the transmitter. The bootloader program ends the download after a timeout of four character times or 768 bytes. When loading is complete, the program jumps to location $0080 and begins executing the code. Use of an external pull-up resistor is required when using the SCI transmitter pin (TxD) because port D pins are configured for wired-OR operation by the bootloader. In bootstrap mode, the interrupt vectors point to RAM. This allows the use of interrupts through a jump table. Refer to Motorola application note AN1060, *M68HC11 Bootstrap Mode.*

4.2 On-Chip Memory

The M68HC11 N-series MCUs include 768 bytes of on-chip RAM, 24 Kbytes of ROM/EPROM and 640 bytes of EEPROM. The bootloader ROM occupies 512 bytes. The CONFIG register is implemented as a separate EEPROM byte.

4.2.1 Mapping Allocations

Memory locations for on-chip resources are the same for both expanded and single-chip modes. The 128-byte register block originates at $0000 after reset and can be placed at any other 4 Kbyte boundary ($x000) after reset by writing an appropriate value to the INIT register. Refer to Figure 4–1, which illustrates the memory map.

The on-board 768-byte RAM is initially located at $0080 after reset. The RAM is divided into two sections of 128 bytes and 640 bytes. If RAM and registers are both mapped to the same 4 Kbyte boundary, RAM starts at $x080 and 128 bytes are remapped at $x300–$x37F. Otherwise, RAM starts at $x000. Remapping is accomplished by writing appropriate values into the INIT register.

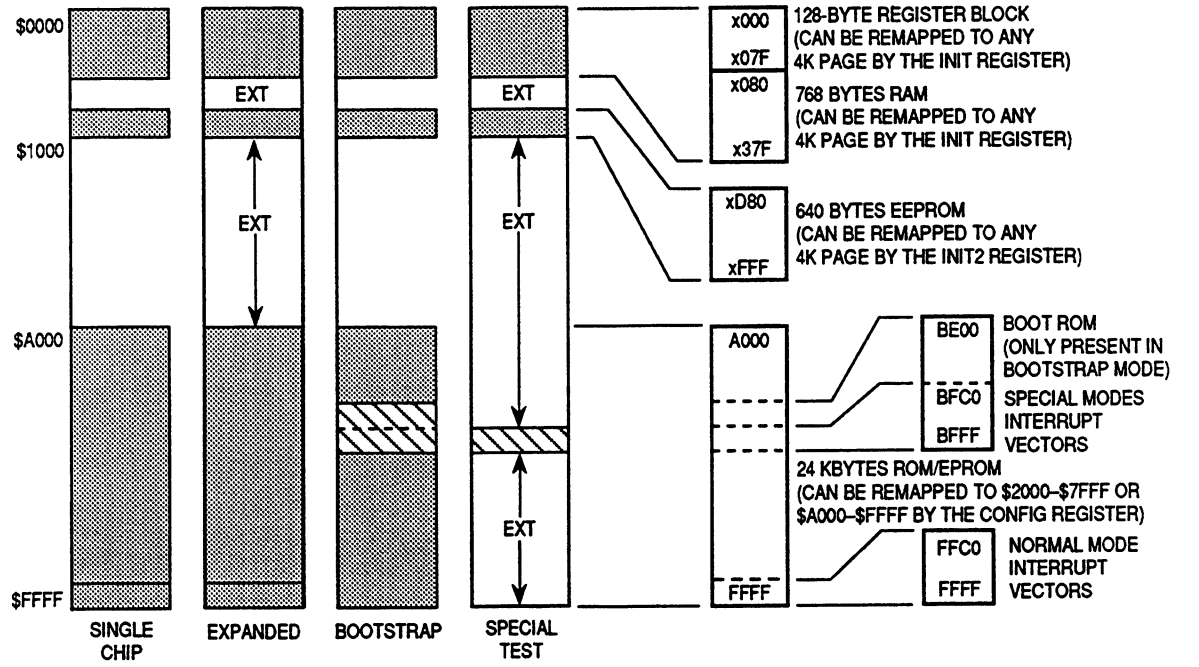
The 640-byte EEPROM is initially located at $0D80 after reset when EEPROM is enabled in the memory map by the CONFIG register. EEPROM can be placed at any other 4 Kbyte boundary ($xD80) by writing to the INIT2 register.

If ROM/EPROM is available, the ROMAD and ROMON bits in the CONFIG register control the position and presence of ROM/EPROM in the memory map. In special test mode, the ROMON bit is cleared so the ROM is removed from the memory map. In single-chip mode, the ROMAD bit is set to one after reset,

which enables the ROM at $A000–$FFFF. In expanded mode, the ROM/EPROM may be enabled from $2000–7FFF (ROMAD = 0) to allow an external memory to contain the interrupt vectors and initialization code.

In bootstrap mode, a bootloader ROM is enabled at locations $BE00–$BFFF. The vectors for bootstrap mode are contained in the bootloader program. The boot ROM fills 512 bytes of the memory map even though not all locations are used.

4.2.2 Memory Map



NOTE: ROM/EPROM CAN BE ENABLED IN SPECIAL TEST MODE BY SETTING ROMON BIT IN THE CONFIG REGISTER AFTER RESET.

N4 MEM MAP

Figure 4–1. MC68HC11N4 Memory Map

4.2.2.1 RAM

The M68HC11 N-series microcontrollers have 768 bytes of fully static RAM that is used for storing instructions, variables, and temporary data during program execution. RAM can be placed at any 4 Kbyte boundary in the 64 Kbyte address space by writing an appropriate value to the INIT register.

RAM is initially located at $0080 in the memory map upon reset. Direct addressing mode can access the first 128 locations of RAM using a one-byte address operand. Direct mode accesses save program memory space and execution time. Registers can be moved to other 4 Kbyte boundaries to allow 256 bytes of RAM to be located in direct addressing space.

The on-chip RAM is a fully static memory. RAM contents can be preserved during periods of processor inactivity by either of two methods, both of which reduce power consumption.

During the software-based STOP mode, MCU clocks are stopped, but the MCU continues to draw power from V_{DD}. Power supply current is directly related to operating frequency in CMOS integrated circuits and there is very little leakage when the clocks are stopped. These two factors reduce power consumption while the MCU is in STOP mode.

To reduce power consumption to a minimum, V_{DD} can be turned off, and the MODB/V_{STBY} pin can be used to supply RAM power from either a battery back-up or a second power supply. Although this method requires external hardware, it is very effective. Refer to **SECTION 2 PIN DESCRIPTIONS** for information about how to connect the standby RAM power supply. Refer to **SECTION 5 RESETS AND INTERRUPTS** for a description of low power operation.

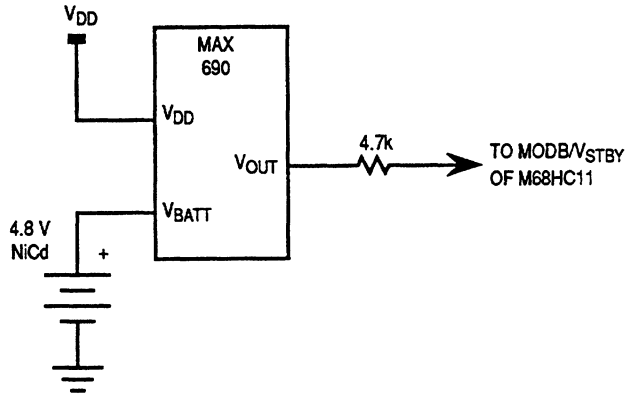


Figure 4–2. RAM Standby MODB/V_{STBY} Connections

4.2.2.2 ROM and EPROM

The M68HC11 N-series MCUs have 24 Kbytes of ROM/EPROM. The ROM/EPROM array is enabled when the ROMON bit in the CONFIG register is set to one (erased). The ROMAD bit in CONFIG places the ROM/EPROM at either $A000–$FFFF out of reset (ROMAD = 1) or at $2000–$7FFF (ROMAD = 0) in expanded mode.

4.2.2.3 Bootloader ROM

The bootloader ROM is enabled at address $BE00–$BFFF during bootstrap mode. The reset vector is fetched from this ROM and the MCU executes the bootloader firmware. In normal modes, the bootloader ROM is disabled.

4.2.3 Registers

In Table 4–1, a summary of registers and control bits, the registers are shown in ascending order within the 128-byte register block. The addresses shown are for default block mapping ($0000–$007F), however, the INIT register can remap the register block to any 4 Kbyte page, $x000–$x07F.

Table 4–1. Register and Control Bit Assignments (1 of 4)

The register block can be remapped to any 4 Kbyte boundary.

	Bit 7	6	5	4	3	2	1	Bit 0	
$0000	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0	PORTA
$0001	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	DDRA
$0002	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	DDRB
$0003	DDF7	DDF6	DDF5	DDF4	DDF3	DDF2	DDF1	DDF0	DDRF
$0004	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0	PORTB
$0005	PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0	PORTF
$0006	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0	PORTC
$0007	DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	DDRC
$0008	0	0	PD5	PD4	PD3	PD2	PD1	PD0	PORTD
$0009	0	0	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	DDRD
$000A	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0	PORTE
$000B	FOC1	FOC2	FOC3	FOC4	FOC5	0	0	0	CFORC
$000C	OC1M7	OC1M6	OC1M5	OC1M4	OC1M3	0	0	0	OC1M
$000D	OC1D7	OC1D6	OC1D5	OC1D4	OC1D3	0	0	0	OC1D
$000E	Bit 15	14	13	12	11	10	9	Bit 8	TCNT (High)
$000F	Bit 7	6	5	4	3	2	1	Bit 0	TCNT (Low)
$0010	Bit 15	14	13	12	11	10	9	Bit 8	TIC1 (High)
$0011	Bit 7	6	5	4	3	2	1	Bit 0	TIC1 (Low)
$0012	Bit 15	14	13	12	11	10	9	Bit 8	TIC2 (High)
$0013	Bit 7	6	5	4	3	2	1	Bit 0	TIC2 (Low)
$0014	Bit 15	14	13	12	11	10	9	Bit 8	TIC3 (High)
$0015	Bit 7	6	5	4	3	2	1	Bit 0	TIC3 (Low)
$0016	Bit 15	14	13	12	11	10	9	Bit 8	TOC1 (High)
$0017	Bit 7	6	5	4	3	2	1	Bit 0	TOC1 (Low)
$0018	Bit 15	14	13	12	11	10	9	Bit 8	TOC2 (High)
$0019	Bit 7	6	5	4	3	2	1	Bit 0	TOC2 (Low)

Table 4–1. Register and Control Bit Assignments (2 of 4)

	Bit 7	6	5	4	3	2	1	Bit 0	
$001A	Bit 15	14	13	12	11	10	9	Bit 8	TOC3 (High)
$001B	Bit 7	6	5	4	3	2	1	Bit 0	TOC3 (Low)
$001C	Bit 15	14	13	12	11	10	9	Bit 8	TOC4 (High)
$001D	Bit 7	6	5	4	3	2	1	Bit 0	TOC4 (Low)
$001E	Bit 15	14	13	12	11	10	9	Bit 8	TI4/O5 (High)
$001F	Bit 7	6	5	4	3	2	1	Bit 0	TI4/O5 (Low)
$0020	OM2	OL2	OM3	OL3	OM4	OL4	OM5	OL5	TCTL1
$0021	EDG4B	EDG4A	EDG1B	EDG1A	EDG2B	EDG2A	EDG3B	EDG3A	TCTL2
$0022	OC1I	OC2I	OC3I	OC4I	I4/O5I	IC1I	IC2I	IC3I	TMSK1
$0023	OC1F	OC2F	OC3F	OC4F	I4/O5F	IC1F	IC2F	IC3F	TFLG1
$0024	TOI	RTII	PAOVI	PAII	0	0	PR1	PR0	TMSK2
$0025	TOF	RTIF	PAOVF	PAIF	0	0	0	0	TFLG2
$0026	0	PAEN	PAMOD	PEDGE	0	I4/O5	RTR1	RTR0	PACTL
$0027	Bit 7	6	5	4	3	2	1	Bit 0	PACNT
$0028	SPIE	SPE	DWOM	MSTR	CPOL	CPHA	SPR1	SPR0	SPCR
$0029	SPIF	WCOL	0	MODF	0	0	0	Bit 0	SPSR
$002A	Bit 7	6	5	4	3	2	1	Bit 0	SPDR
$002B	MBE	0	ELAT	EXCOL	EXROW	0	0	EPGM	EPROG*
$002C	0	0	0	0	HPPUE	GPPUE	FPPUE	BPPUE	PPAR
$002D									Reserved
$002E									Reserved
$002F									Reserved
$0030	CCF	0	SCAN	MULT	CD	CC	CB	CA	ADCTL
$0031	Bit 7	6	5	4	3	2	1	Bit 0	ADR1
$0032	Bit 7	6	5	4	3	2	1	Bit 0	ADR2
$0033	Bit 7	6	5	4	3	2	1	Bit 0	ADR3
$0034	Bit 7	6	5	4	3	2	1	Bit 0	ADR4
$0035	BULKP	0	BPRT4	PTCON	BPRT3	BPRT2	BPRT1	BPRT0	BPROT
$0036									Reserved

* Devices with EPROM only

Table 4–1. Register and Control Bit Assignments (3 of 4)

	Bit 7	6	5	4	3	2	1	Bit 0	
$0037	EE3	EE2	EE1	EE0	0	0	0	0	INIT2
$0038	LIRDV	CWOM	STRCH	IRVNE	LSBF	SPR2	0	0	OPT2
$0039	ADPU	CSEL	IRQE	DLY	CME	FCME	CR1	CR0	OPTION
$003A	Bit 7	6	5	4	3	2	1	Bit 0	COPRST
$003B	ODD	EVEN	0	BYTE	ROW	ERASE	EELAT	EEPGM	PPROG
$003C	RBOOT	SMOD	MDA	PSEL4	PSEL3	PSEL2	PSEL1	PSEL0	HPRIO
$003D	RAM3	RAM2	RAM1	RAM0	REG3	REG2	REG1	REG0	INIT
$003E	TILOP	0	OCCR	CBYP	DISR	FCM	FCOP	0	TEST1
$003F	ROMAD	1	1	PAREN	NOSEC	NOCOP	ROMON	EEON	CONFIG
$0040	Bit 31	30	29	28	27	26	25	Bit 24	CREG (High)
$0041	Bit 23	22	21	20	19	18	17	Bit 16	CREG (Mid-Hi)
$0042	Bit 15	14	13	12	11	10	9	Bit 8	CREG (Mid-Lo)
$0043	Bit 7	6	5	4	3	2	1	Bit 0	CREG (Low)
$0044	SIG	DIV	MAC	DCC	TRG	0	0	0	ALUC
$0045	Bit 15	14	13	12	11	10	9	Bit 8	AREG (High)
$0046	Bit 7	6	5	4	3	2	1	Bit 0	AREG (Low)
$0047	Bit 15	14	13	12	11	10	9	Bit 8	BREG (High)
$0048	Bit 7	6	5	4	3	2	1	Bit 0	BREG (Low)
$0049	NEG	RZF	0	0	0	OVF	DZF	ACF	ALUF
$004A									Reserved
$004B									Reserved
$004C									Reserved
$004D	0	0	0	0	0	0	DAE2	DAE1	DACON
$004E	Bit 7	6	5	4	3	2	1	Bit 0	DA1
$004F	Bit 7	6	5	4	3	2	1	Bit 0	DA2
$0050	0	0	PCKC2	PCKC1	0	0	PPOL6	PPOL5	PWCTL
$0051	0	0	0	0	PW6S2	PW6S1	PW5S2	PW5S1	PWSIZ
$0052	0	0	0	Bit 12	11	10	9	Bit 8	PWTDY5 (Hi)
$0053	Bit 7	6	5	4	3	2	1	Bit 0	PWTDY5 (Lo)
$0054	0	0	0	Bit 12	11	10	9	Bit 8	PWTDY6 (Hi)
$0055	Bit 7	6	5	4	3	2	1	Bit 0	PWTDY6 (Lo)
$0056	0	0	0	0	Bit 11	10	9	Bit 8	PWCNT5 (Hi)
$0057	Bit 7	6	5	4	3	2	1	Bit 0	PWCNT5 (Lo)
$0058									Reserved
to									
$005F									Reserved

Table 4–1. Register and Control Bit Assignments (4 of 4)

	Bit 7	6	5	4	3	2	1	Bit 0	
$0060	CON34	CON12	PCKA2	PCKA1	0	PCKB3	PCKB2	PCKB1	PWCLK
$0061	PCLK4	PCLK3	PCLK2	PCLK1	PPOL4	PPOL3	PPOL2	PPOL1	PWPOL
$0062	Bit 7	6	5	4	3	2	1	Bit 0	PWSCAL
$0063	TPWSL	DISCP	0	0	PWEN4	PWEN3	PWEN2	PWEN1	PWEN
$0064	Bit 7	6	5	4	3	2	1	Bit 0	PWCNT1
$0065	Bit 7	6	5	4	3	2	1	Bit 0	PWCNT2
$0066	Bit 7	6	5	4	3	2	1	Bit 0	PWCNT3
$0067	Bit 7	6	5	4	3	2	1	Bit 0	PWCNT4
$0068	Bit 7	6	5	4	3	2	1	Bit 0	PWPER1
$0069	Bit 7	6	5	4	3	2	1	Bit 0	PWPER2
$006A	Bit 7	6	5	4	3	2	1	Bit 0	PWPER3
$006B	Bit 7	6	5	4	3	2	1	Bit 0	PWPER4
$006C	Bit 7	6	5	4	3	2	1	Bit 0	PWDTY1
$006D	Bit 7	6	5	4	3	2	1	Bit 0	PWDTY2
$006E	Bit 7	6	5	4	3	2	1	Bit 0	PWDTY3
$006F	Bit 7	6	5	4	3	2	1	Bit 0	PWDTY4
$0070	BTST	BSPL	0	SBR12	SBR11	SBR10	SBR9	SBR8	SCBDH
$0071	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0	SCBDL
$0072	LOOPS	WOMS	0	M	WAKE	ILT	PE	PT	SCCR1
$0073	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK	SCCR2
$0074	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF	SCSR1
$0075	0	0	0	0	0	0	0	RAF	SCSR2
$0076	R8	T8	0	0	0	0	0	0	SCDRH
$0077	R7/T7	R6/T6	R5/T5	R4/T4	R3/T3	R2/T2	R1/T1	R0/T0	SCDRL
$0078									Reserved
to									
$007B									Reserved
$007C	PH7	PH6	PH5	PH4	PH3	PH2	PH1	PH0	PORTH
$007D	DDH7	DDH6	DDH5	DDH4	DDH3	DDH2	DDH1	DDH0	DDRH
$007E	PG7	PG6	PG5	PG4	PG3	PG2	PG1	PG0	PORTG
$007F	DDG7	DDG6	0	0	0	0	0	0	DDRG

4.3 System Initialization

Registers and bits that control initialization and the basic operation of the MCU are protected against writes except under special circumstances. The following table lists registers that can be written only once after reset or that must be written within the first 64 cycles after reset.

Table 4–2. Write Access Limited Registers

Register Address	Register Name	Must be Written in First 64 Cycles	Write One Time Only
$x024	Timer Interrupt Mask 2 (TMSK2)	Note 1	—
$x035	Block Protect Register (BPROT)	Note 2	—
$x037	EEPROM Mapping Register (INIT2)	No	Yes
$x038	System Configuration Options 2 (OPT2)	No	Note 4
$x039	System Configuration Options (OPTION)	Note 3	—
$x03D	RAM and I/O Map Register (INIT)	Yes	Yes

Notes:

1. Bits 1 and 0 can be written once only in first 64 cycles. When SMOD = 1, these bits can be written any time. All other bits can be written at any time.
2. Bits can be written to zero once only in first 64 cycles or at any time in special modes. Bits can be set to one at any time.
3. Bits 5, 4, 2, 1, and 0 can be written once only in first 64 cycles. When SMOD = 1, bits 5, 4, 2, 1, and 0 can be written at any time. All other bits can be written at any time.
4. Bit 4 (IRVNE) can be written only one time.

4.3.1 Mode Selection

The four mode variations are selected by the logic states of the MODA and MODB pins during reset. The MODA and MODB logic levels determine the logic state of SMOD and the MDA control bits in the highest priority I-bit interrupt and miscellaneous (HPRIO) register.

After reset is released, the mode select pins no longer influence the MCU operating mode. In single-chip operating mode, the MODA pin is connected to a logic level zero. In expanded mode, MODA is normally connected to V_{DD} through a pull-up resistor of 4.7 kΩ. The MODA pin also functions as the load instruction register $\overline{LIR}$ pin when the MCU is not in reset. The open-drain active low $\overline{LIR}$ output pin drives low during the first E cycle of each instruction. The MODB pin also functions as standby power input (V_{STBY}), which allows RAM contents to be maintained in absence of V_{DD}.

Refer to Table 4–3, which is a summary of mode pin operation, the mode control bits, and the four operating modes.

Table 4–3. Hardware Mode Select Summary

Input Levels at Reset		Mode	Control Bits in HPRIO (Latched at Reset)		
MODB	MODA		RBOOT	SMOD	MDA
1	0	Single Chip	0	0	0
1	1	Expanded	0	0	1
0	0	Bootstrap	1	1	0
0	1	Special Test	0	1	1

A normal mode is selected when MODB is logic one during reset. One of three reset vectors is fetched from address $FFFA–$FFFF, and program execution begins from the address indicated by this vector. If MODB is logic zero during reset, the special mode reset vector is fetched from addresses $BFFA–$BFFF and software has access to special test features. Refer to **SECTION 5 RESETS AND INTERRUPTS**.

4.3.1.1 HPRIO Register

Bits in the HPRIO register select the highest priority interrupt level, select whether bootstrap ROM is present, and reflect the state of MODA and MODB pins at reset.

HPRIO — Highest Priority I-Bit Interrupt and Miscellaneous **$003C**

	Bit 7	6	5	4	3	2	1	Bit 0	
	RBOOT	SMOD	MDA	PSEL4	PSEL3	PSEL2	PSEL1	PSEL0	
RESET:									
	0	0	0	0	0	1	1	0	Single Chip
	0	0	1	0	0	1	1	0	Expanded
	1	1	0	0	0	1	1	0	Bootstrap
	0	1	1	0	0	1	1	0	Special Test

RBOOT, SMOD, and MDA bits depend on power-up initialization mode. Refer to Table 4–3.

RBOOT — Read Bootstrap ROM

Set to one out of reset in bootstrap mode. Valid while in special modes only. Can be read any time. Can only be written in special modes.

0 = Bootloader ROM disabled and not in map

1 = Bootloader ROM enabled and in map at $BE00–$BFFF

SMOD — Special Mode Select

Can be read any time. Can only be written in special modes (SMOD = 1). Can only be written to zero.

0 = Normal mode variation

1 = Special mode variation

MDA — Mode Select A

Can be read any time. Can be written any time in special modes (SMOD = 1). Can be written only once in normal modes (SMOD = 0).

0 = Normal single-chip or bootstrap mode

1 = Normal expanded or special test mode

PSEL[4:0] — Priority Select Bits

Refer to **SECTION 5 RESETS AND INTERRUPTS**.

4.3.2 Initialization

Because bits in the following registers control the basic configuration of the MCU, an accidental change of their values could cause serious system problems. The protection mechanism, overridden in special operating modes, requires a write to the protected bits only within the first 64 bus cycles after any reset, or only once after each reset. Table 4–2 summarizes the write access limited registers.

4.3.2.1 CONFIG Register

CONFIG controls the presence of ROM/EPROM and EEPROM as well as the location of ROM/EPROM in the memory map. CONFIG also enables the COP watchdog system and a register that controls I/O port pull-up resistor assignments. Refer to **4.4.4 CONFIG Register Programming**.

CONFIG — System Configuration Register **$003F**

	Bit 7	6	5	4	3	2	1	Bit 0
	ROMAD	—	—	PAREN	NOSEC	NOCOP	ROMON	EEON
RESET:	—	1	1	—	1	—	—	—

CONFIG is made up of EEPROM cells and static working latches. The operation of the MCU is controlled directly by these latches and not the actual EEPROM byte. When programming the CONFIG register, the EEPROM byte is being accessed. When the CONFIG register is being read, the static latches are being accessed.

These bits can be read at any time. The value read is the one latched into the register from the EEPROM cells during the last reset sequence. A new value programmed into this register is not readable until after a subsequent reset sequence. Unused bits always read as ones.

If SMOD = 1, CONFIG bits can be written at any time. If SMOD = 0, CONFIG bits can only be written using the EEPROM programming sequence, and are neither readable nor active until latched via the next reset.

ROMAD — ROM Mapping Control

In single-chip mode, ROMAD is forced to one out of reset.

0 = ROM addressed from $2000 to $7FFF (expanded mode only)
1 = ROM addressed from $A000 to $FFFF

Bits [6:5] — Not implemented

Always read one

PAREN — Pull-Up Assignment Register Enable

0 = PPAR register disabled; all pull-ups disabled as well
1 = PPAR register enabled; pull-ups can be enabled using PPAR

NOSEC — RAM and EEPROM Security Disable

M68HC11 N-series devices are normally manufactured with NOSEC set to one and the security option is unavailable. On special request, a mask option is selected during fabrication that enables the security mode. On these parts, the secure mode is invoked by programming the NOSEC bit to zero. Refer to **4.4.5 RAM and EEPROM Security**.

0 = Enable Security
1 = Disable Security

NOCOP — COP System Disable

0 = COP system enabled (forces reset on timeout)
1 = COP system disabled

ROMON — ROM Enable

In single-chip mode ROMON is forced to one out of reset. In special test mode ROMON is forced to zero out of reset.

0 = 24 Kbytes of ROM disabled from the memory map
1 = 24 Kbytes of ROM present in the memory map

EEON — EEPROM Enable

0 = 640 bytes of EEPROM is disabled from the memory map
1 = 640 bytes of EEPROM is present in the memory map

4.3.2.2 INIT Register

The internal registers used to control the operation of the MCU can be relocated on 4 Kbyte boundaries within the memory space with the use of INIT. This 8-bit special-purpose register can change the default locations of the RAM and control registers within the MCU memory map. It can be written only once within the first 64 E-clock cycles after a reset. It then becomes a read-only register.

INIT — RAM and I/O Mapping Register **$003D**

	Bit 7	6	5	4	3	2	1	Bit 0
	RAM3	RAM2	RAM1	RAM0	REG3	REG2	REG1	REG0
RESET:	0	0	0	0	0	0	0	0

RAM[3:0] — RAM Map Position

These four bits, which specify the upper hexadecimal digit of the RAM address, control position of RAM in the memory map. RAM can be positioned at the beginning of any 4 Kbyte page in the memory map. Refer to Table 4–4.

REG[3:0] — 128-Byte Register Block Position

These four bits specify the upper hexadecimal digit of the address for the 128-byte block of internal registers. The register block is positioned at the beginning of any 4 Kbyte page in the memory map. Refer to Table 4–4.

Table 4–4. RAM and Register Remapping

RAM[3:0]	Location
0000	$0000–$02FF
0001	$1000–$12FF
0010	$2000–$22FF
0011	$3000–$32FF
0100	$4000–$42FF
0101	$5000–$52FF
0110	$6000–$62FF
0111	$7000–$72FF
1000	$8000–$82FF
1001	$9000–$92FF
1010	$A000–$A2FF
1011	$B000–$B2FF
1100	$C000–$C2FF
1101	$D000–$D2FF
1110	$E000–$E2FF
1111	$F000–$F2FF

REG[3:0]	Location
0000	$0000–$007F
0001	$1000–$107F
0010	$2000–$207F
0011	$3000–$307F
0100	$4000–$407F
0101	$5000–$507F
0110	$6000–$607F
0111	$7000–$707F
1000	$8000–$807F
1001	$9000–$907F
1010	$A000–$A07F
1011	$B000–$B07F
1100	$C000–$C07F
1101	$D000–$D07F
1110	$E000–$E07F
1111	$F000–$F07F

When the memory map has the 128-byte register block mapped at the same location as RAM, the registers have priority and the RAM is relocated to the memory space immediately following the register block. This mapping feature keeps all the RAM available for use. Refer to Figure 4–3 which illustrates the overlap.

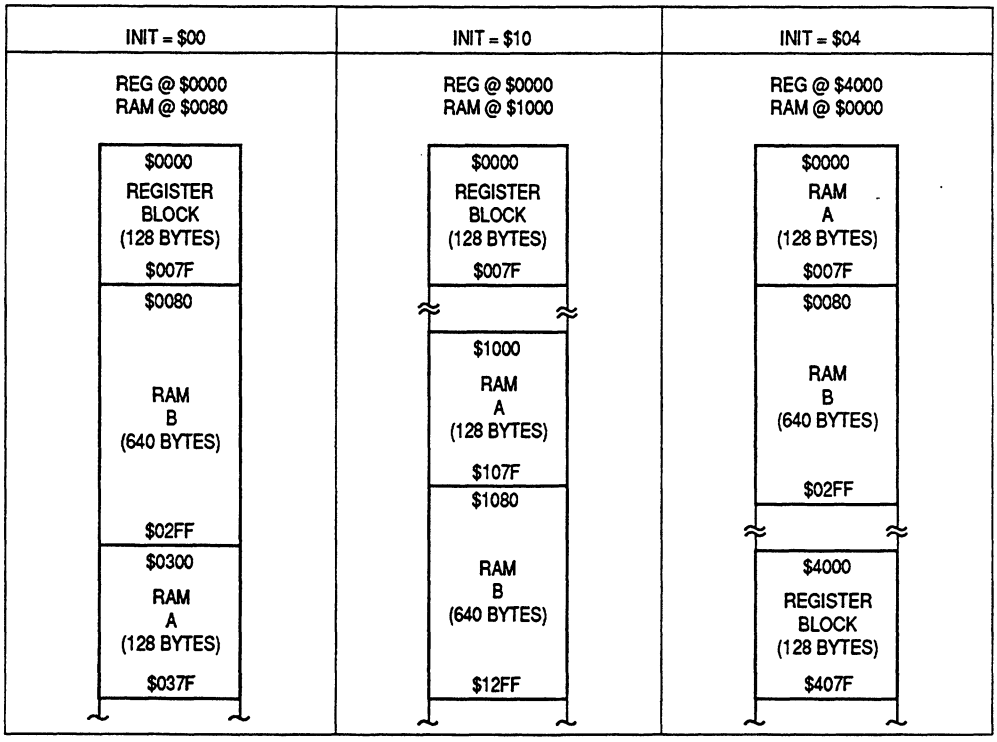


Figure 4–3. RAM and Register Overlap

4.3.2.3 INIT2 Register

This register determines the location of EEPROM in the memory map. INIT2 may be read at any time but may be written only once after reset in normal modes.

INIT2 — EEPROM Mapping **$0037**

	Bit 7	6	5	4	3	2	1	Bit 0
	EE3	EE2	EE1	EE0	—	—	—	—
RESET:	0	0	0	0	0	0	0	0

EE[3:0] — EEPROM Map Position
EEPROM is located at $xD80–$xFFF, where x is the hexadecimal digit represented by EE[3:0]. Refer to Table 4–5.

Bits [3:0] — Not implemented
Always read zero

Table 4–5. EEPROM Map

EE[3:0]	Location
0000	$0D80–$0FFF
0001	$1D80–$1FFF
0010	$2D80–$2FFF
0011	$3D80–$3FFF
0100	$4D80–$4FFF
0101	$5D80–$5FFF
0110	$6D80–$6FFF
0111	$7D80–$7FFF
1000	$8D80–$8FFF
1001	$9D80–$9FFF
1010	$AD80–$AFFF
1011	$BD80–$BFFF
1100	$CD80–$CFFF
1101	$DD80–$DFFF
1110	$ED80–$EFFF
1111	$FD80–$FFFF

4.3.2.4 OPTION Register

The 8-bit special-purpose OPTION register sets internal system configuration options during initialization. The time protected control bits, IRQE, DLY, and CR[1:0] can be written to only once after a reset and then they become read-only bits. This minimizes the possibility of any accidental changes to the system configuration.

OPTION — System Configuration Options **$0039**

	Bit 7	6	5	4	3	2	1	Bit 0
	ADPU	CSEL	IRQE*	DLY*	CME	FCME*	CR1*	CR0*
RESET:	0	0	0	1	0	0	0	0

*Can be written only once in first 64 cycles out of reset in normal modes or at any time in special modes.

ADPU — A/D Power-Up
- 0 = A/D system disabled
- 1 = A/D system power enabled

CSEL — Clock Select

Selects alternate clock source for on-chip EEPROM and A/D charge pumps. On-chip resistance-capacitance (RC) clock should be used when E clock falls below 1 MHz.
- 0 = A/D and EEPROM use system E clock
- 1 = A/D and EEPROM use internal RC clock

IRQE — Configure $\overline{\text{IRQ}}$ for Falling Edge-Sensitive Operation
- 0 = Low level-sensitive operation.
- 1 = Falling edge-sensitive only operation.

DLY — Enable Oscillator Startup Delay
- 0 = The oscillator startup delay coming out of STOP is bypassed and the MCU resumes processing within about four bus cycles.
- 1 = A delay of approximately 4000 E-clock cycles is imposed as the MCU is started up from the STOP power-saving mode.

CME — Clock Monitor Enable

In order to use both STOP and clock monitor, the CME bit should be written to one before executing STOP, then written to one after recovering from STOP. Refer to **SECTION 5 RESETS AND INTERRUPTS**.
- 0 = Clock monitor disabled
- 1 = Clock monitor enabled

FCME — Force Clock Monitor Enable

When FCME equals one, slow or stopped clocks will cause a clock failure reset. To use STOP mode, FCME should always equal zero. Refer to **SECTION 5 RESETS AND INTERRUPTS**.
- 0 = Clock monitor follows state of CME bit
- 1 = Clock monitor enabled and cannot be disabled until next reset

CR[1:0] — COP Timer Rate Select Bits

These control bits determine a scaling factor for the watchdog timer. Refer to **SECTION 5 RESETS AND INTERRUPTS**.

OPT2 — System Configuration Options 2 **$0038**

	Bit 7	6	5	4	3	2	1	Bit 0
	LIRDV	CWOM	STRCH	IRVNE*	LSBF	SPR2	—	—
RESET:	0	0	0	—	0	0	0	0

*Can be written only once in any mode.

LIRDV — LIR Driven

In single-chip and bootstrap modes, this bit has no meaning or effect. The LIR pin is normally configured for wired-OR operation (only pulls low). In order to detect consecutive instructions in a high-speed application, this signal can be made to drive high for a short time to prevent false triggering.

0 = LIR not driven high out of reset

1 = LIR driven high for one quarter cycle to reduce transition time

CWOM — Port C Wired-OR Mode

Refer to **SECTION 6 PARALLEL INPUT/OUTPUT**.

0 = Port C operates normally.

1 = Port C outputs are open-drain.

STRCH — Clock Stretch for External Accesses

0 = Normal operation, no effect on E clock.

1 = External accesses to locations $0000–$7FFF are extended by one E clock cycle. E clock is stretched externally but there is no effect on the internal clocks. In single-chip and bootstrap modes, this bit has no effect.

IRVNE — Internal Read Visibility/Not E

IRVNE can be written once in any mode. In expanded modes, IRVNE determines whether IRV is on or off. In special test mode, IRVNE is reset to one. In all other modes, IRVNE is reset to zero.

0 = No internal read visibility on external bus

1 = Data from internal reads is driven out of the external data bus.

In single-chip modes this bit determines whether the E clock drives out from the chip.

0 = E is driven out from the chip.

1 = E pin is driven low. Refer to the following table.

Mode	IRVNE Out of Reset	E Clock Out of Reset	IRV Out of Reset	IRVNE Affects Only	IRVNE Can Be Written
Single Chip	0	On	Off	E	Once
Expanded	0	On	Off	IRV	Once
Boot	0	On	Off	E	Once
Special Test	1	On	On	IRV	Once

NOTE

Use caution when changing modes since the IRVNE bit takes on the function of the new mode as soon as the write is complete. Only one write is allowed so make sure IRVNE is written to the state desired for the mode change.

LSBF — LSB First Enable
Refer to **SECTION 8 SERIAL PERIPHERAL INTERFACE.**

SPR2 — SPI Clock Rate Select
Refer to **SECTION 8 SERIAL PERIPHERAL INTERFACE.**

Bits [1:0] — Not implemented
Always read zero.

4.3.2.5 Block Protect Register (BPROT)

BPROT prevents accidental writes to EEPROM and the CONFIG register. The bits in this register can be written to zero during the first 64 E-clock cycles after reset in the normal modes. Once the bits are cleared to zero, the EEPROM array and the CONFIG register can be programmed or erased. Setting the bits in the BPROT register to logic one protects the EEPROM and CONFIG register until the next reset. Refer to Table 4–6.

BPROT — Block Protect **$0035**

	Bit 7	6	5	4	3	2	1	Bit 0
	BULKP	—	BPRT4	PTCON	BPRT3	BPRT2	BPRT1	BPRT0
RESET:	1	1	1	1	1	1	1	1

BULKP — Bulk Erase of EEPROM Protect
0 = EEPROM can be bulk erased normally.
1 = EEPROM cannot be bulk or row erased.

Bit 6 — Not implemented
Always reads zero.

PTCON — Protect for CONFIG
0 = CONFIG register can be programmed or erased normally
1 = CONFIG register cannot be programmed or erased

BPRT[4:0] — Block Protect Bits for EEPROM
0 = Protection disabled for associated block
1 = Protection enabled for associated block

Table 4–6. EEPROM Block Protect

Bit Name	Block Protected	Block Size
BPRT0	$xD80–$xD9F	32 Bytes
BPRT1	$xDA0–$xDDF	64 Bytes
BPRT2	$xDE0–$xE5F	128 Bytes
BPRT3	$xE60–$xF7F	288 Bytes
BPRT4	$xF80–$xFFF	128 Bytes

4.3.2.6 Timer Interrupt Mask Register 2 (TMSK2)

Bits in TMSK2 correspond bit for bit with flag bits in TFLG2. Ones in TMSK2 enable the corresponding interrupt sources.

TMSK2 — Timer Interrupt Mask Register 2 **$0024**

	Bit 7	6	5	4	3	2	1	Bit 0
	TOI	RTII	PAOVI	PAII	—	—	PR1*	PR0*
RESET:	0	0	0	0	0	0	0	0

*Can be written only once during the first 64 cycles out of reset, any time in special modes.

TOI — Timer Overflow Interrupt Enable
0 = TOF interrupts disabled
1 = Interrupt requested when TOF is set to one

RTII — Real-Time Interrupt Enable
0 = RTIF interrupts disabled
1 = Interrupt requested when RTIF set to one

PAOVI — Pulse Accumulator Overflow Interrupt Enable
Refer to **SECTION 9 TIMING SYSTEM.**

PAII — Pulse Accumulator Interrupt Enable
Refer to **SECTION 9 TIMING SYSTEM.**

Bits [3:2] — Not implemented
Always read zero

PR[1:0] — Timer Prescaler Select
These two bits select the prescale rate for the main 16-bit free-running timer system. These bits can be written only once during the first 64 E-Clock cycles after reset in normal modes or any time in special modes. Refer to the following table.

PR[1:0]	Prescale Factor
00	1
01	4
10	8
11	16

4.4 EPROM, EEPROM, and CONFIG Register

4.4.1 EPROM Programming

Using the on-chip EPROM programming feature requires an external 12.25-volt power supply (V_{PPE}). Normal programming is accomplished using the EPROG register.

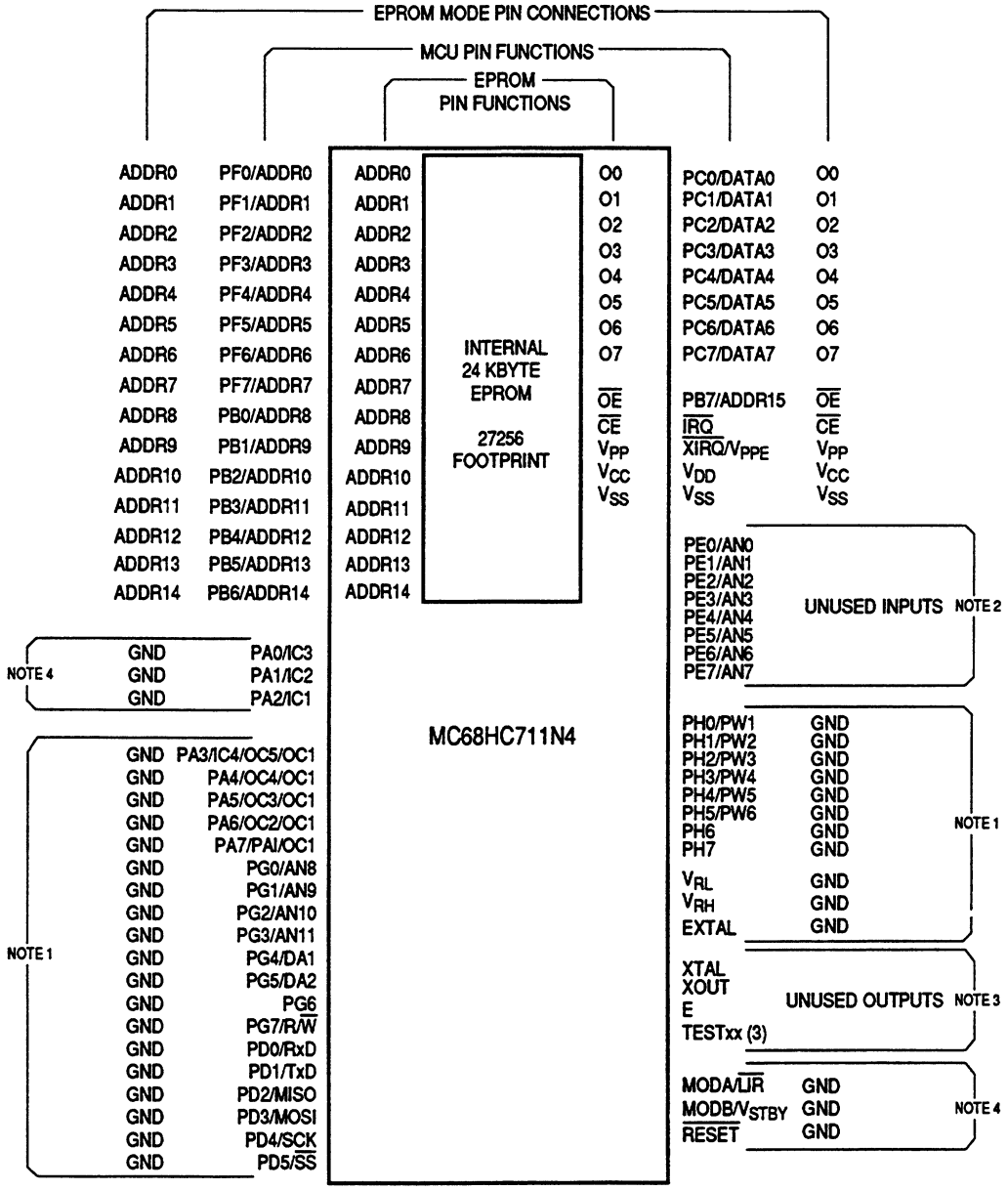
There are three methods of programming and verifying EPROM. In the first method, the PROM emulation (PROG) mode, program the EPROM as an industry-standard EPROM by adapting the MCU footprint to that of the 27256-type EPROM, as shown in Figure 4–4, and use an appropriate EPROM programmer.

If the MCU is operating with programming voltage present on the $\overline{XIRQ}/V_{PPE}$ pin, the $\overline{IRQ}/\overline{CE}$ pin must be pulled to a high level before the address and data are changed to program the next location.

In the second method, use the following procedure to program the EPROM through the MCU with the ROMON bit set in the CONFIG register. On entry, A = data to be programmed and X = EPROM address.

```
EPROG       LDAB        #$20
            STAB        $002B       Set ELAT bit (PGM=0) to enable EPROM latches.
            STAA        $0, X       Store data to EPROM address
            LDAB        #$21
            STAB        $002B       Set EPGM bit with ELAT=1 to enable EPROM
                                    programming voltage
            JSR         DLYEP       Delay 2-4 ms
            CLR         $002B       Turn off programming voltage and set to READ mode
```

In the third method, the EPROM is programmed by software while in the special test or bootstrap modes. User-developed software can be uploaded through the SCI, or a ROM-resident EPROM programming utility can be used. To use the resident utility, bootload a three-byte program consisting of a single jump instruction to $BF00. $BF00 is the starting address of a resident EPROM programming utility. The utility program sets the X and Y index registers to default values, then receives programming data from an external host and puts it in EPROM. The value in IX determines programming delay time. The value in IY is a pointer to the first address in EPROM to be programmed (default = $D000). When the utility program is ready to receive programming data, it sends the host the $FF character. Then it waits. When the host sees the $FF character, the EPROM programming data is sent, starting with location $D000. After the last byte to be programmed is sent and the corresponding verification data is returned, the programming operation is terminated by resetting the MCU.



NOTES:
1. UNUSED INPUTS — GROUNDING IS RECOMMENDED.
2. UNUSED INPUTS — THESE PINS MAY BE LEFT UNTERMINATED.
3. UNUSED OUTPUTS — THESE PINS SHOULD BE LEFT UNCONNECTED.
4. GROUNDING THESE SIX PINS CONFIGURES THE MC68HC711N4 FOR EPROM EMULATION MODE.

Figure 4–4. MC68HC711N4 EPROM PROG Mode Connections

4.4.1.1 EPROM Programming Control Register (EPROG)

The EPROG register is used when programming the EPROM using the on-chip method instead of the standard EPROM emulation method. On parts without EPROM, this register has no meaning or effect.

EPROG — EPROM Programming Control Register **$002B**

	Bit 7	6	5	4	3	2	1	Bit 0
	MBE	—	ELAT	EXCOL	EXROW	—	—	EPGM
RESET:	0	0	0	0	0	0	0	0

MBE — Multiple-Byte Program Enable

When MBE equals one, program two bytes with same data. Address bit 5 is ignored so that the bytes with ADDR5 = 0 and ADDR5 = 1 both get programmed. Always reads zero in normal modes. Can only be written in special modes.

0 = Normal programming
1 = Multiple-byte programming enabled

Bit 6 — Not implemented

Always reads zero.

ELAT — EPROM Latch Control

When ELAT equals one, writes to EPROM cause address and data to be latched. When programming EPROM, be sure that EELAT bit in PPROG register is cleared. EELAT negates the function of ELAT. Can be read or written any time.

0 = EPROM address and data bus configured for normal reads
1 = EPROM address and data bus configured for programming

EXCOL — Select Extra Columns

When EXCOL equals one, extra columns can be accessed at bit 7 and bit 0. Addresses use bits [11:5]. Bits [4:1] are ignored. Always reads zero in normal modes. Can only be written in special modes.

0 = User array selected
1 = Extra columns selected and user array is disabled

EXROW — Select Extra Rows

When EXROW equals one, two extra rows are available. Addresses use bits [5:0]. Bits [11:6] are ignored. Always reads zero in normal modes. Can only be written in special modes.

0 = User array selected
1 = Extra rows selected and user array is disabled

Bits [2:1] — Not implemented

Always read zero.

EPGM — EPROM Programming Enable

Can be read any time. EPGM can only be written to one if ELAT equals one.

0 = Programming voltage (V_{PPE}) is off
1 = Programming voltage (V_{PPE}) is on

4.4.2 EEPROM

The 640-byte on-board EEPROM is initially located from $0D80 to $0FFF after reset in all modes. It can be mapped to any other 4 Kbyte boundary by writing to the INIT2 register. The EEPROM is enabled by the EEON bit in the CONFIG register. Programming and erasing is controlled by the PPROG register.

Unlike information stored in ROM, data in the 640 bytes of EEPROM can be erased and reprogrammed under software control. Because programming and erasing operations use an on-chip charge pump driven by V_{DD}, a separate external power supply is not required.

An internal charge pump supplies the programming voltage. Use of the block protect register (BPROT) prevents inadvertent writes to (or erases of) blocks of EEPROM, and enables or disables a low voltage inhibit circuit for further protection. The CSEL bit in the OPTION register selects an on-chip oscillator clock for programming and erasing while operating at frequencies below 1 MHz. Refer to **SECTION 5 RESETS AND INTERRUPTS**.

In special modes there is an extra row of 16 bytes of EEPROM (located at $0D60), which is used for factory testing. Endurance and data retention specifications do not apply to this row.

4.4.2.1 EEPROM Programming

Writes to EEPROM addresses are inhibited while EEPGM is one. A write to a different EEPROM location is prevented while a programming or erase operation is in progress.

When the EELAT bit in the PPROG register is cleared, the EEPROM can be read as if it were a ROM. The block protect register has no effect during reads.

During EEPROM programming, the ROW and BYTE bits of PPROG are not used. If the frequency of the E clock is 1 MHz or less, set the CSEL bit in the OPTION register. Recall that zeros must be erased by a separate erase operation before programming. The following example of how to program an EEPROM byte assumes that the appropriate bits in BPROT have been cleared.

```
PROG    LDAB    #$02        EELAT=1
        STAB    $003B       Set EELAT bit
        STAA    $0D80       Store data to EEPROM address
        LDAB    #$03        EELAT=EEPGM=1
        STAB    $003B       Turn on programming voltage
        JSR     DLY10       Delay 10 ms
        CLR     $003B       Turn off high voltage and set to READ mode
```

4.4.2.2 EEPROM Bulk Erase

To erase the EEPROM, ensure that the proper bits of the BPROT register are cleared, then complete the following steps using the PPROG register:

1. Write to PPROG with the ERASE, EELAT, and appropriate BYTE and ROW bits set.
2. Write to the appropriate EEPROM address with any data. Row erase only requires a write to any location in the row. Bulk erase is accomplished by writing to any location in the array.
3. Write to PPROG with ERASE, EELAT, EEPGM, and the appropriate BYTE and ROW bits set.
4. Delay for 10 ms or more, as appropriate.
5. Clear the EEPGM bit in PPROG to turn off the high voltage.
6. Clear the PPROG register to reconfigure the EEPROM address and data buses for normal operation.

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

```
BULKE   LDAB    #$06        ERASE=EELAT=1
        STAB    $003B       Set EELAT bit
        STAA    $0D80       Store data to any EEPROM address
        LDAB    #$03        EELAT=EEPGM=1
        STAB    $003B       Turn on programming voltage
        JSR     DLY10       Delay 10 ms
        CLR     $003B       Turn off high voltage and set to READ mode
```

4.4.2.3 EEPROM Row Erase

The following example shows how to perform a fast erase of large sections of EEPROM.

```
ROWE    LDAB    #$0E        ROW=ERASE=EELAT=1
        STAB    $003B       Set to ROW erase mode
        STAB    0,X         Write any data to any address in ROW
        LDAB    #$0F        ROW=ERASE=EELAT=EEPGM=1
        STAB    $003B       Turn on high voltage
        JSR     DLY10       Delay 10 ms
        CLR     $003B       Turn off high voltage and set to READ mode
```

4.4.2.4 EEPROM Byte Erase

The following is an example of how to erase a single byte of EEPROM.

```
BYTEE   LDAB    #$16        BYTE=ERASE=EELAT=1
        STAB    $003B       Set to BYTE erase mode
        STAB    0,X         Write any data to address to be erased
        LDAB    #$17        BYTE=ERASE=EELAT=EEPGM=1
        STAB    $003B       Turn on high voltage
        JSR     DLY10       Delay 10 ms
        CLR     $003B       Turn off high voltage and set to READ mode
```

4.4.3 EEPROM Programming Control Register (PPROG)

Bits in PPROG register control parameters associated with EEPROM programming.

PPROG — EEPROM Programming Control **$003B**

	Bit 7	6	5	4	3	2	1	Bit 0
	ODD	EVEN	—	BYTE	ROW	ERASE	EELAT	EEPGM
RESET:	0	0	0	0	0	0	0	0

ODD — Program Odd Rows in Half of EEPROM (TEST)

EVEN — Program Even Rows in Half of EEPROM (TEST)

Bit 5 — Not implemented
Always reads zero.

BYTE — Byte/Other EEPROM Erase Mode (only valid when ERASE = 0)
0 = Row or bulk erase mode used
1 = Erase only one byte of EEPROM

ROW — Row/All EEPROM Erase Mode (only valid when ERASE = 0)
0 = All 640 bytes of EEPROM erased
1 = Erase only one 16-byte row of EEPROM

BYTE	ROW	Action
0	0	Bulk Erase (All 640 Bytes)
0	1	Row Erase (16 Bytes)
1	0	Byte Erase
1	1	Byte Erase

ERASE — Erase/Normal Control for EEPROM

Can be read or written any time.

0 = Normal read or program mode

1 = Erase mode

EELAT — EEPROM Latch Control

Can be read or written any time. When EELAT equals one, writes to EEPROM cause address and data to be latched. EELAT negates the function of ELAT (EPROM programming latch control).

0 = EEPROM address and data bus configured for normal reads

1 = EEPROM address and data bus configured for programming or erasing

EEPGM — EEPROM Program Command

Can be read any time. Can only be written while EELAT = 1.

0 = Program or erase voltage switched off to EEPROM array

1 = Program or erase voltage switched on to EEPROM array

4.4.4 CONFIG Register Programming

Because the CONFIG register is implemented with EEPROM cells, use EEPROM procedures to erase and program this register. The procedure for programming is the same as for programming a byte in the EEPROM array, except that the CONFIG register address is used. CONFIG can be programmed or erased (including byte erase) while the MCU is operating in any mode, provided that PTCON in BPROT is clear. To change the value in the CONFIG register, complete the following procedure. Do not initiate a reset until the procedure is complete.

1. Erase the CONFIG register.
2. Program the new value to the CONFIG address.
3. Initiate reset.

CONFIG — System Configuration Register **$003F**

	Bit 7	6	5	4	3	2	1	Bit 0
	ROMAD	—	—	PAREN	NOSEC	NOCOP	ROMON	EEON
RESET:	—	1	1	—	1	—	—	—

For a description of the bits contained in the CONFIG register refer to **4.3.2.1 CONFIG Register**.

4.4.5 RAM and EEPROM Security

The optional security feature protects the contents of EEPROM and RAM from unauthorized access. A program, or a key portion of a program, can be protected against duplication. To accomplish this, the protection mechanism restricts operation of protected devices to single-chip modes, and thus prevents the memory locations from being monitored externally because single-chip modes do not allow visibility of the internal address and data buses. Resident programs, however, have unlimited access to the internal EEPROM and RAM and can read, write, or transfer the contents of these memories. The NOSEC bit in the CONFIG register disables this feature on devices that incorporate it. Contact a Motorola representative for information on the availability of this feature.

If the security feature is present and enabled and bootstrap mode is selected, the following sequence is performed by the bootstrap program:

1. Output $FF (all ones) on the SCI.
2. Turn block protect off. Clear BPROT register.
3. If EEPROM is enabled, erase EEPROM.
4. Verify that the EEPROM is erased. If EEPROM is not erased, begin sequence again.
5. Write $FF (all ones) to the entire block of RAM.
6. Erase CONFIG register.

If all of the operations above are successful, the bootloading process continues as if the device were never secured.

SECTION 5
RESETS AND INTERRUPTS

Resets and interrupt operations load the program counter with a vector that points to a new location from which instructions are to be fetched. A reset immediately stops execution of the current instruction and forces the program counter to a known starting address. Internal registers and control bits are initialized so the MCU can resume executing instructions. An interrupt temporarily suspends normal program execution while an interrupt service routine is being executed; once the interrupt has been serviced, the main program resumes as if there had been no interruption.

5.1 Resets

There are four possible sources of reset. Power-on reset (POR) and external reset share the normal reset vector. The computer operating properly (COP) reset and the clock monitor reset each has its own vector.

5.1.1 Power-On Reset

A positive transition on V_{DD} generates a power-on reset (POR), which is used only for power-up conditions. POR cannot be used to detect drops in power supply voltages. A 4064 t_{cyc} (internal clock cycle) delay after the oscillator becomes active allows the clock generator to stabilize. If $\overline{\text{RESET}}$ is at logic zero at the end of 4064 t_{cyc}, the CPU remains in the reset condition until $\overline{\text{RESET}}$ goes to logic one.

It is important to protect the MCU during power transitions. Most M68HC11 systems need an external circuit that holds the $\overline{\text{RESET}}$ pin low whenever V_{DD} is below the minimum operating level. This external voltage level detector, or other external reset circuits, are the usual source of reset in a system. The POR circuit only initializes internal circuitry during cold starts. Refer to Figure 2–2.

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

The CPU distinguishes between internal and external reset conditions by sensing whether the reset pin rises to a logic one in less than two E-clock cycles after an internal device releases reset. When a reset condition is sensed, the $\overline{\text{RESET}}$ pin is driven low by an internal device for four E-clock cycles, then released. Two E-clock cycles later it is sampled. If the pin is still held low, the

CPU assumes that an external reset has occurred. If the pin is high, it indicates that the reset was initiated internally by either the COP system or the clock monitor. It is not advisable to connect an external RC power-up delay circuit to the reset pin of M68HC11 devices because the circuit charge time constant can cause the device to misinterpret the type of reset that occurred.

5.1.3 COP Reset

The MCU includes a COP system to help protect against software failures. When the COP is enabled, the software is responsible for keeping a free-running watchdog timer from timing out. When the software is no longer being executed in the intended sequence, a system reset is initiated.

The state of the NOCOP bit in the CONFIG register determines whether the COP system is enabled or disabled. To change the enable status of the COP system, change the contents of the CONFIG register and then perform a system reset. In the special test and bootstrap operating modes, the COP system is initially inhibited by the disable resets (DISR) control bit in the TEST1 register. The DISR bit can subsequently be written to zero to enable COP resets.

The COP timer rate control bits CR[1:0] in the OPTION register determine the COP timeout period. The system E clock is divided by 2^{15} and then further scaled by a factor shown in Table 5–1. After reset, these bits are zero, which selects the fastest timeout period. In normal operating modes, these bits can only be written once within 64 bus cycles after reset.

Table 5–1. COP Timer Rate Select

CR[1:0]	Divide $E/2^{15}$ By	XTAL = 8.0 MHz Timeout –0/+16.4 ms	XTAL = 12.0 MHz Timeout –0/+10.9 ms	XTAL = 16.0 MHz Timeout –0/+8.2 ms
0 0	1	16.384 ms	10.923 ms	8.192 ms
0 1	4	65.536 ms	43.691 ms	32.768 ms
1 0	16	262.14 ms	174.76 ms	131.07 ms
1 1	64	1.049 sec	699.05 ms	524.29 ms
	E =	2.0 MHz	3.0 MHz	4.0 MHz

COPRST — Arm/Reset COP Timer Circuitry **$103A**

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

Complete the following reset sequence to service the COP timer. Write $55 to COPRST to arm the COP timer clearing mechanism. Then write $AA to COPRST to clear the COP timer. Performing instructions between these two steps is possible as long as both steps are completed in the correct sequence before the timer times out.

5.1.4 Clock Monitor Reset

The clock monitor circuit is based on an internal RC time delay. If no MCU clock edges are detected within this RC time delay, the clock monitor can optionally generate a system reset. The clock monitor function is enabled or disabled by the CME control bit in the OPTION register. The presence of a timeout is determined by the RC delay, which allows the clock monitor to operate without any MCU clocks.

Clock monitor is used as a backup for the COP system. Because the COP needs a clock to function, it is disabled when the clocks stop. Therefore, the clock monitor system can detect clock failures not detected by the COP system.

Semiconductor wafer processing causes variations of the RC timeout values between individual devices. An E-clock frequency below 10 kHz is detected as a clock monitor error. An E-clock frequency of 200 kHz or more prevents clock monitor errors. Using the clock monitor function when the E-clock is below 200 kHz is not recommended.

Special considerations are needed when a STOP instruction is executed and the clock monitor is enabled. Because the STOP function causes the clocks to be halted, the clock monitor function generates a reset sequence if it is enabled at the time the STOP mode was initiated. Before executing a STOP instruction, clear the CME bit in the OPTION register to zero to disable the clock monitor. After recovery from STOP, set the CME bit to logic one to enable the clock monitor.

5.1.5 OPTION Register

OPTION — System Configuration Options **$0039**

	Bit 7	6	5	4	3	2	1	Bit 0
	ADPU	CSEL	IRQE*	DLY*	CME	FCME*	CR1*	CR0*
RESET:	0	0	0	1	0	0	0	0

*Can be written only once in first 64 cycles out of reset in normal modes, or at any time in special modes.

ADPU — Analog-to-Digital Converter Power-Up
Refer to **SECTION 10 ANALOG-TO-DIGITAL CONVERTER.**

CSEL — Clock Select
Refer to **SECTION 10 ANALOG-TO-DIGITAL CONVERTER.**

IRQE — Configure $\overline{\text{IRQ}}$ for Edge-Sensitive Only Operation
Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY.**

DLY — Enable Oscillator Startup Delay
Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY.**

CME — Clock Monitor Enable
This control bit can be read or written at any time and controls whether or not the internal clock monitor circuit triggers a reset sequence when the system clock is slow or absent. When it is clear, the clock monitor circuit is disabled, and when it is set, the clock monitor circuit is enabled. Reset clears the CME bit.

FCME — Force Monitor Clock Enable
0 = Clock monitor follows the state of the CME bit
1 = Clock monitor circuit is enabled until next reset

CR[1:0] — COP Timer Rate Select
The internal E clock is first divided by 2^{15} before it enters the COP watchdog system. These control bits determine a scaling factor for the watchdog timer.

5.1.6 CONFIG Register

CONFIG — System Configuration Register **$003F**

	Bit 7	6	5	4	3	2	1	Bit 0
	ROMAD	—	—	PAREN	NOSEC	NOCOP	ROMON	EEON
RESET:	—	1	1	—	1	—	—	—

ROMAD — ROM (EPROM) Mapping Control
Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY.**

Bits [6:5] — Not implemented
Always read one

PAREN — Pull-Up Assignment Register Enable
Refer to **SECTION 6 PARALLEL INPUT/OUTPUT.**

NOSEC — RAM and EEPROM Security Disable
Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY.**

NOCOP — COP System Disable
0 = COP enabled (forces reset on timeout)
1 = COP disabled (does not force reset on timeout)

ROMON — ROM (EPROM) Enable
Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY.**

EEON — EEPROM Enable
Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY.**

5.2 Effects of Reset

When a reset condition is recognized, the internal registers and control bits are forced to an initial state. Depending on the cause of the reset and the operating mode, the reset vector can be fetched from any of six possible locations. Refer to Table 5–2.

Table 5–2. Reset Cause, Reset Vector, and Operating Mode

Cause of Reset	Normal Mode Vector	Special Test or Bootstrap
POR or RESET Pin	$FFFE, FFFF	$BFFE, $BFFF
Clock Monitor Failure	$FFFC, FFFD	$BFFC, $BFFD
COP Watchdog Timeout	$FFFA, FFFB	$BFFA, $BFFB

These initial states then control on-chip peripheral systems to force them to known startup states, as follows:

5.2.1 Central Processing Unit

After reset, the CPU fetches the restart vector from the appropriate address 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 (CCR) are set to mask any interrupt requests. Also, the S bit in the CCR is set to inhibit the STOP mode.

5.2.2 Memory Map

After reset, the INIT register is initialized to $00, putting the 768 bytes of RAM at locations $0080 through $037F, and the control registers at locations $0000 through $007F. The INIT2 register puts EEPROM at locations $0D80–$0FFF.

5.2.3 Parallel I/O

When a reset occurs in expanded operating modes, port B, C, and F pins used for parallel I/O are dedicated to the expansion bus. If a reset occurs during single-chip operating mode, all ports are configured as general-purpose high-impedance inputs.

NOTE

Do not confuse pin function with the electrical state of the pin at reset. All general-purpose I/O pins configured as inputs at reset are in a high-impedance state. Port data registers reflect the port's functional state at reset. The pin function is mode dependent.

5.2.4 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 so that they do not affect any I/O pins on successful compares. All 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 because their mask bits have been cleared.

The I4/O5 bit in the PACTL register is cleared to configure the I4/O5 function as OC5; however, the OM5:OL5 control bits in the TCTL1 register are clear so OC5 does not control the PA3 pin.

5.2.5 Real-Time Interrupt (RTI)

The real-time interrupt flag (RTIF) is cleared and automatic hardware interrupts are masked. The rate control bits are cleared after reset and can be initialized by software before the real-time interrupt (RTI) system is used.

5.2.6 Pulse Accumulator

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

5.2.7 Computer Operating Properly (COP)

The COP watchdog system is enabled if the NOCOP control bit in the CONFIG register is cleared, and disabled if NOCOP is set. The COP rate is set for the shortest duration timeout.

5.2.8 Serial Communications Interface (SCI)

The reset condition of the SCI system is independent of the operating mode. At reset, the SCI baud rate control register is initialized to $0004. 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 wakeup 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, FE, PF, and RAF receive-related status bits are cleared.

5.2.9 Serial Peripheral Interface (SPI)

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

5.2.10 Analog-to-Digital Converter

The A/D converter configuration is indeterminate after reset. The ADPU bit is cleared by reset, which disables the A/D system. The conversion complete flag is cleared by reset.

5.2.11 System

The EEPROM programming controls are disabled, so the memory system is configured for normal read operation. PSEL[4:0] are initialized with the binary value %00110, causing the external $\overline{IRQ}$ pin to have the highest I-bit interrupt priority. The $\overline{IRQ}$ pin is configured for level-sensitive operation (for wired-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 because CME and FCME are cleared.

5.3 Reset and Interrupt Priority

Resets and interrupts have a hardware priority that determines which reset or interrupt is serviced first when simultaneous requests occur. Any maskable interrupt can be given priority over other maskable interrupts.

The first six interrupt sources are not maskable. The priority arrangement for these sources is as follows:

1. POR or $\overline{\text{RESET}}$ pin
2. Clock monitor reset
3. COP watchdog reset
4. $\overline{\text{XIRQ}}$ interrupt
5. Illegal opcode interrupt
6. Software interrupt (SWI)

The maskable interrupt sources have the following priority arrangement:

1. $\overline{\text{IRQ}}$
2. Real-time interrupt
3. Timer input capture 1
4. Timer input capture 2
5. Timer input capture 3
6. Timer output compare 1
7. Timer output compare 2
8. Timer output compare 3
9. Timer output compare 4
10. Timer input capture 4/output compare 5
11. Timer overflow
12. Pulse accumulator overflow
13. Pulse accumulator input edge
14. SPI transfer complete
15. SCI system (refer to Figure 5–3)

Any one of these interrupts can be assigned the highest maskable interrupt priority by writing the appropriate value to the PSEL bits in the HPRIO register. Otherwise, the priority arrangement remains the same. An interrupt that is assigned highest priority is still subject to global masking by the I bit in the CCR, or by any associated local bits. Interrupt vectors are not affected by priority assignment. To avoid race conditions, HPRIO can only be written while I-bit interrupts are inhibited.

5.3.1 Highest Priority Interrupt and Miscellaneous Register

HPRIO — Highest Priority I-Bit Interrupt and Miscellaneous **$003C**

	Bit 7	6	5	4	3	2	1	Bit 0	
	RBOOT*	SMOD*	MDA*	PSEL4	PSEL3	PSEL2	PSEL1	PSEL0	
RESET:									
	0	0	0	0	0	1	1	0	Single Chip
	0	0	1	0	0	1	1	0	Expanded
	1	1	0	0	0	1	1	0	Bootstrap
	0	1	1	0	0	1	1	0	Special Test

*The values of the RBOOT, SMOD, and MDA reset bits depend on the mode during power-up. Refer to Table 4–3.

RBOOT — Read Bootstrap ROM

Set to one out of reset in bootstrap mode. Valid while in special modes only. Can be read any time. Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY** for more information.

SMOD — Special Mode Select

Can be read any time. Can only be written in special modes (SMOD = 1). Can only be written to zero. Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY** for more information.

MDA — Mode Select A

Can be read any time. Can be written any time in special modes (SMOD = 1). Can be written only once in normal modes (SMOD = 0). Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY** for more information.

PSEL[4:0] — Priority Select Bits

These bits select one interrupt source to be elevated above all other I-bit-related sources and can be written to only while the I bit in the CCR is set (interrupts disabled).

Table 5–3. Highest Priority Interrupt Selection

PSELx					
4	3	2	1	0	Interrupt Source Promoted
0	0	0	X	X	Reserved (Default to $\overline{\text{IRQ}}$)
0	0	1	0	0	Reserved (Default to $\overline{\text{IRQ}}$)
0	0	1	0	1	Reserved (Default to $\overline{\text{IRQ}}$)
0	0	1	1	0	$\overline{\text{IRQ}}$ (External Pin)
0	0	1	1	1	Real-Time Interrupt
0	1	0	0	0	Timer Input Capture 1
0	1	0	0	1	Timer Input Capture 2
0	1	0	1	0	Timer Input Capture 3
0	1	0	1	1	Timer Output Compare 1
0	1	1	0	0	Timer Output Compare 2
0	1	1	0	1	Timer Output Compare 3
0	1	1	1	0	Timer Output Compare 4
0	1	1	1	1	Timer Input Capture 4/Output Compare 5
1	0	0	0	0	Timer Overflow
1	0	0	0	1	Pulse Accumulator Overflow
1	0	0	1	0	Pulse Accumulator Input Edge
1	0	0	1	1	SPI Serial Transfer Complete
1	0	1	0	0	SCI Serial System
1	0	1	0	1	Reserved (Default to $\overline{\text{IRQ}}$)
1	0	1	1	0	Reserved (Default to $\overline{\text{IRQ}}$)
1	0	1	1	1	Reserved (Default to $\overline{\text{IRQ}}$)
1	1	X	X	X	Reserved (Default to $\overline{\text{IRQ}}$)

5.4 Interrupts

The MCU has 18 interrupt vectors that support 22 interrupt sources. The 15 maskable interrupts are generated by on-chip peripheral systems. These interrupts are recognized when the global interrupt mask bit (I) in the condition code register (CCR) is clear. The three nonmaskable interrupt sources are illegal opcode trap, software interrupt, and $\overline{\text{XIRQ}}$ pin. Refer to Table 5–4, which shows the interrupt sources and vector assignments for each source.

Table 5–4. Interrupt and Reset Vector Assignments

Vector Address	Interrupt Source	CCR Mask Bit	Local Mask
FFC0, C1 — FFD4, D5	Reserved	—	—
FFD6, D7	SCI Serial System	I	
	• SCI Receive Data Register Full		RIE
	• SCI Receiver Overrun		RIE
	• SCI Transmit Data Register Empty		TIE
	• SCI Transmit Complete		TCIE
	• SCI Idle Line Detect		ILIE
FFD8, D9	SPI Serial Transfer Complete	I	SPIE
FFDA, DB	Pulse Accumulator Input Edge	I	PAII
FFDC, DD	Pulse Accumulator Overflow	I	PAOVI
FFDE, DF	Timer Overflow	I	TOI
FFE0, E1	Timer Input Capture 4/Output Compare 5	I	I4/O5I
FFE2, E3	Timer Output Compare 4	I	OC4I
FFE4, E5	Timer Output Compare 3	I	OC3I
FFE6, E7	Timer Output Compare 2	I	OC2I
FFE8, E9	Timer Output Compare 1	I	OC1I
FFEA, EB	Timer Input Capture 3	I	IC3I
FFEC, ED	Timer Input Capture 2	I	IC2I
FFEE, EF	Timer Input Capture 1	I	IC1I
FFF0, F1	Real-Time Interrupt	I	RTII
FFF2, F3	$\overline{\text{IRQ}}$ (External Pin)	I	None
FFF4, F5	$\overline{\text{XIRQ}}$ Pin	X	None
FFF6, F7	Software Interrupt	None	None
FFF8, F9	Illegal Opcode Trap	None	None
FFFA, FB	COP Failure	None	NOCOP
FFFC, FD	Clock Monitor Fail	None	CME
FFFE, FF	$\overline{\text{RESET}}$	None	None

For some interrupt sources, such as the SCI interrupts, the flags are automatically cleared during the normal course of responding to the interrupt requests. For example, the RDRF flag in the SCI system is cleared by the automatic clearing mechanism consisting of a read of the SCI status register while RDRF is set, followed by a read of the SCI data register. The normal response to an RDRF interrupt request would be to read the SCI status register to check for receive errors, then to read the received data from the SCI data register. These two steps satisfy the automatic clearing mechanism without requiring any special instructions.

5.4.1 Interrupt Recognition and Register Stacking

An interrupt can be recognized at any time after it is enabled by its local mask, if any, and by the global mask bit in the CCR. Once an interrupt source is recognized, the CPU responds at the completion of the instruction being executed. Interrupt latency varies according to the number of cycles required to complete the current instruction. When the CPU begins to service an interrupt, the contents of the CPU registers are pushed onto the stack in the order shown in Table 5–5. After the CCR value is stacked, the I bit (and the X bit, if $\overline{\text{XIRQ}}$ is pending) is set to inhibit further interrupts. The interrupt vector for the highest priority pending source is fetched, and execution continues at the address specified by the vector. At the end of the interrupt service routine, the return from interrupt instruction is executed and the saved registers are pulled from the stack in reverse order so that normal program execution can resume. Refer to **SECTION 3 CENTRAL PROCESSING UNIT** for further information.

Table 5–5. Stacking Order on Entry to Interrupts

Memory Location	CPU Registers
SP	PCL
SP – 1	PCH
SP – 2	IYL
SP – 3	IYH
SP – 4	IXL
SP – 5	IXH
SP – 6	ACCA
SP – 7	ACCB
SP – 8	CCR

5.4.2 Nonmaskable Interrupt Request ($\overline{\text{XIRQ}}$)

Nonmaskable interrupts are useful because they can always interrupt CPU operations. The most common use for such an interrupt is for serious system problems, such as program runaway or power failure. The $\overline{\text{XIRQ}}$ input is an updated version of the $\overline{\text{NMI}}$ (nonmaskable interrupt) input of earlier MCUs.

Upon reset, both the X bit and I bit of the CCR are set to inhibit all maskable interrupts and $\overline{\text{XIRQ}}$. After minimum system initialization, software can clear the X bit by a TAP instruction, enabling $\overline{\text{XIRQ}}$ interrupts. Thereafter, software cannot set the X bit. Thus, an $\overline{\text{XIRQ}}$ interrupt is a nonmaskable interrupt. Because the operation of the I-bit-related interrupt structure has no effect on the X bit, the internal $\overline{\text{XIRQ}}$ pin remains nonmasked. In the interrupt priority logic, the $\overline{\text{XIRQ}}$ interrupt has 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 CCR byte. The X bit is not affected. When an X-bit-related interrupt occurs, both the X and I bits are automatically set by hardware after stacking the CCR. A return from interrupt instruction restores the X and I bits to their pre-interrupt request state.

5.4.3 Illegal Opcode Trap

Because not all possible opcodes or opcode sequences are defined, the MCU includes an illegal opcode detection circuit, which generates an interrupt request. When an illegal opcode is detected and the interrupt is recognized, the current value of the program counter is stacked. After interrupt service is complete, reinitialize the stack pointer so repeated execution of illegal opcodes does not cause stack underflow. Left uninitialized, the illegal opcode vector can point to a memory location that contains an illegal opcode. This condition causes an infinite loop that causes stack underflow. The stack grows until the system crashes.

The illegal opcode trap mechanism works for all unimplemented opcodes on all four opcode map pages. The address stacked as the return address for the illegal opcode interrupt is the address of the first byte of the illegal opcode. Otherwise, it would be almost impossible to determine whether the illegal opcode had been one or two bytes. The stacked return address can be used as a pointer to the illegal opcode so the illegal opcode service routine can evaluate the offending opcode.

5.4.4 Software Interrupt

SWI is an instruction, and thus cannot be interrupted until complete. SWI is not inhibited by the global mask bits in the CCR. Because execution of SWI sets the I mask bit, once an SWI interrupt begins, other interrupts are inhibited until SWI is complete, or until user software clears the I bit in the CCR.

5.4.5 Maskable Interrupts

The maskable interrupt structure of the MCU can be extended to include additional external interrupt sources through the $\overline{\text{IRQ}}$ pin. The default configuration of this pin is a low-level sensitive wired-OR network. When an event triggers an interrupt, a software accessible interrupt flag is set. When enabled, this flag causes a constant request for interrupt service. After the flag is cleared, the service request is released.

5.4.6 Reset and Interrupt Processing

Figures 5–1 and 5–2 illustrate the reset and interrupt process. Figure 5–1 shows how the CPU begins from a reset and how interrupt detection relates to normal opcode fetches. Figure 5–2 is an expansion of a block in Figure 5–1 and illustrates interrupt priorities. Figure 5–3 shows the resolution of interrupt sources within the SCI subsystem.

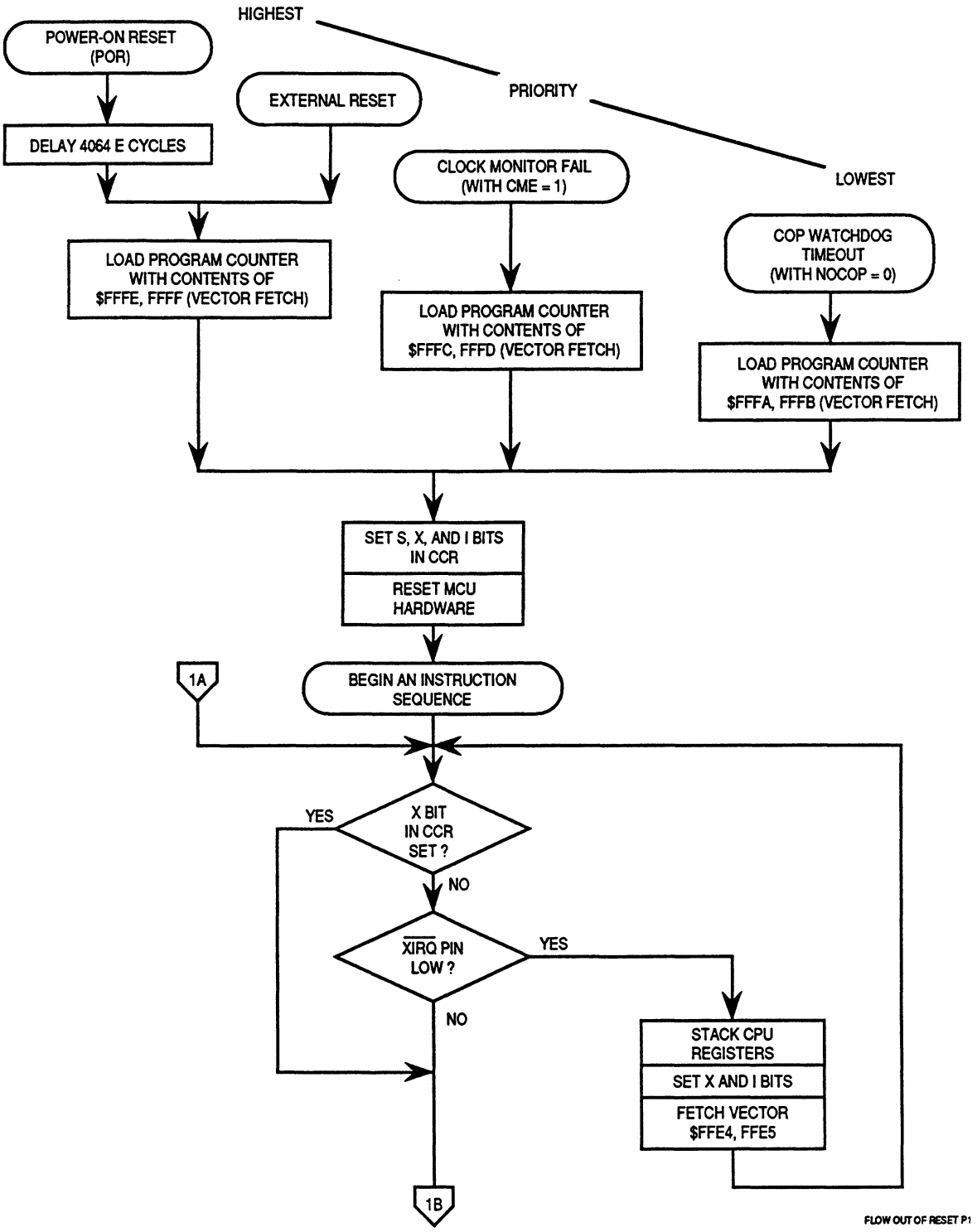


Figure 5–1. Processing Flow out of Reset (1 of 2)

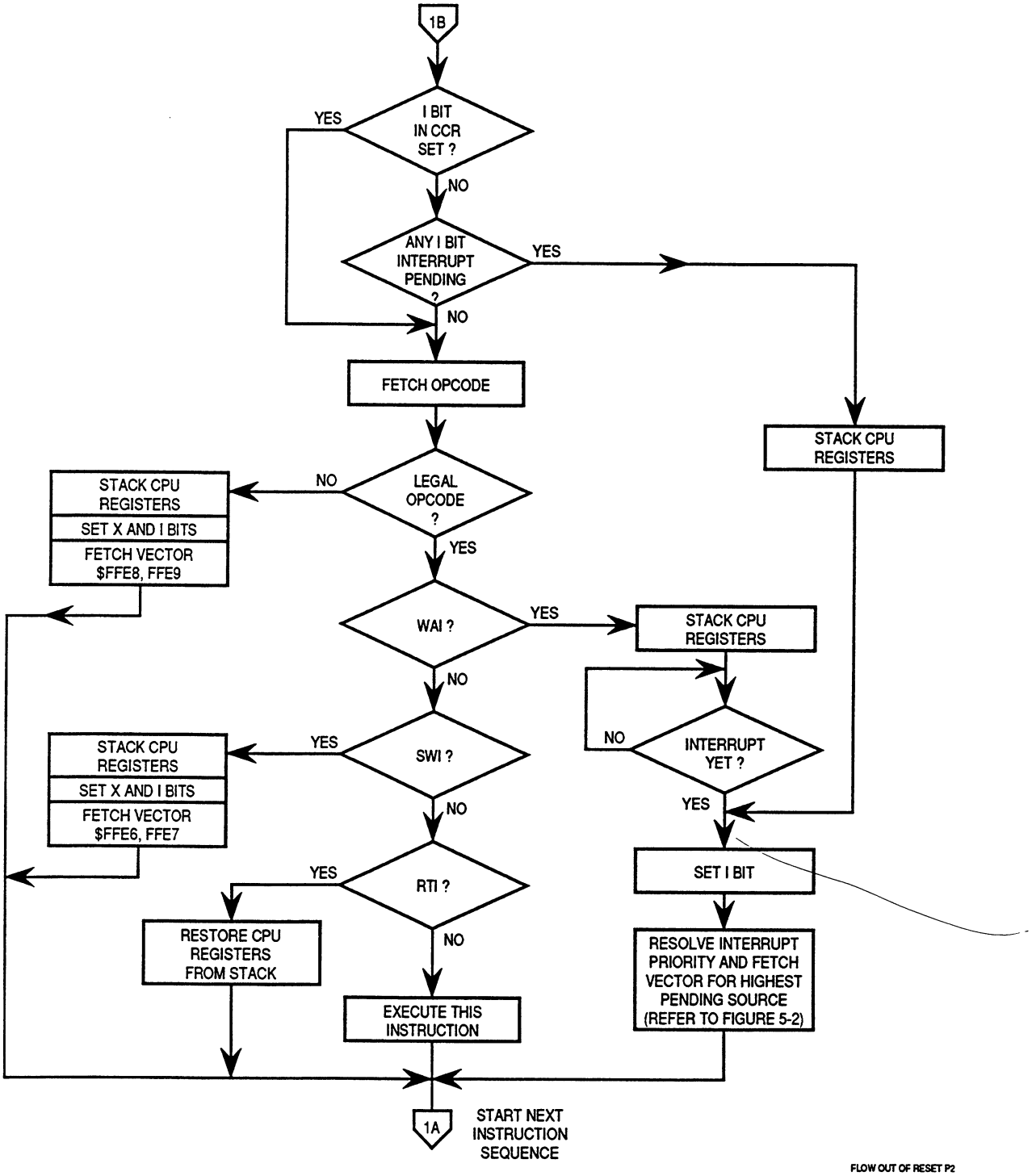


Figure 5–1. Processing Flow out of Reset (2 of 2)

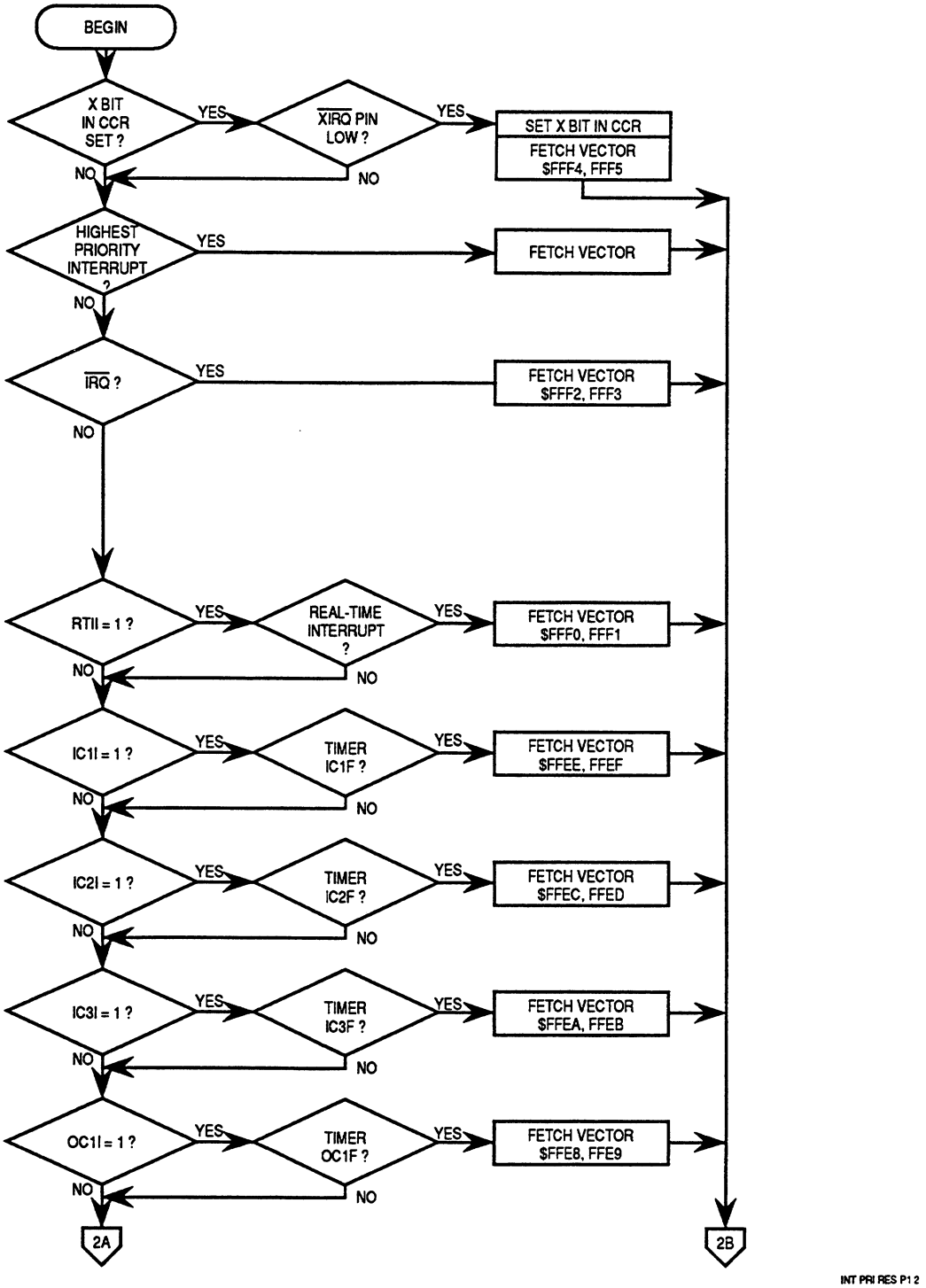


Figure 5–2. Interrupt Priority Resolution (1 of 2)

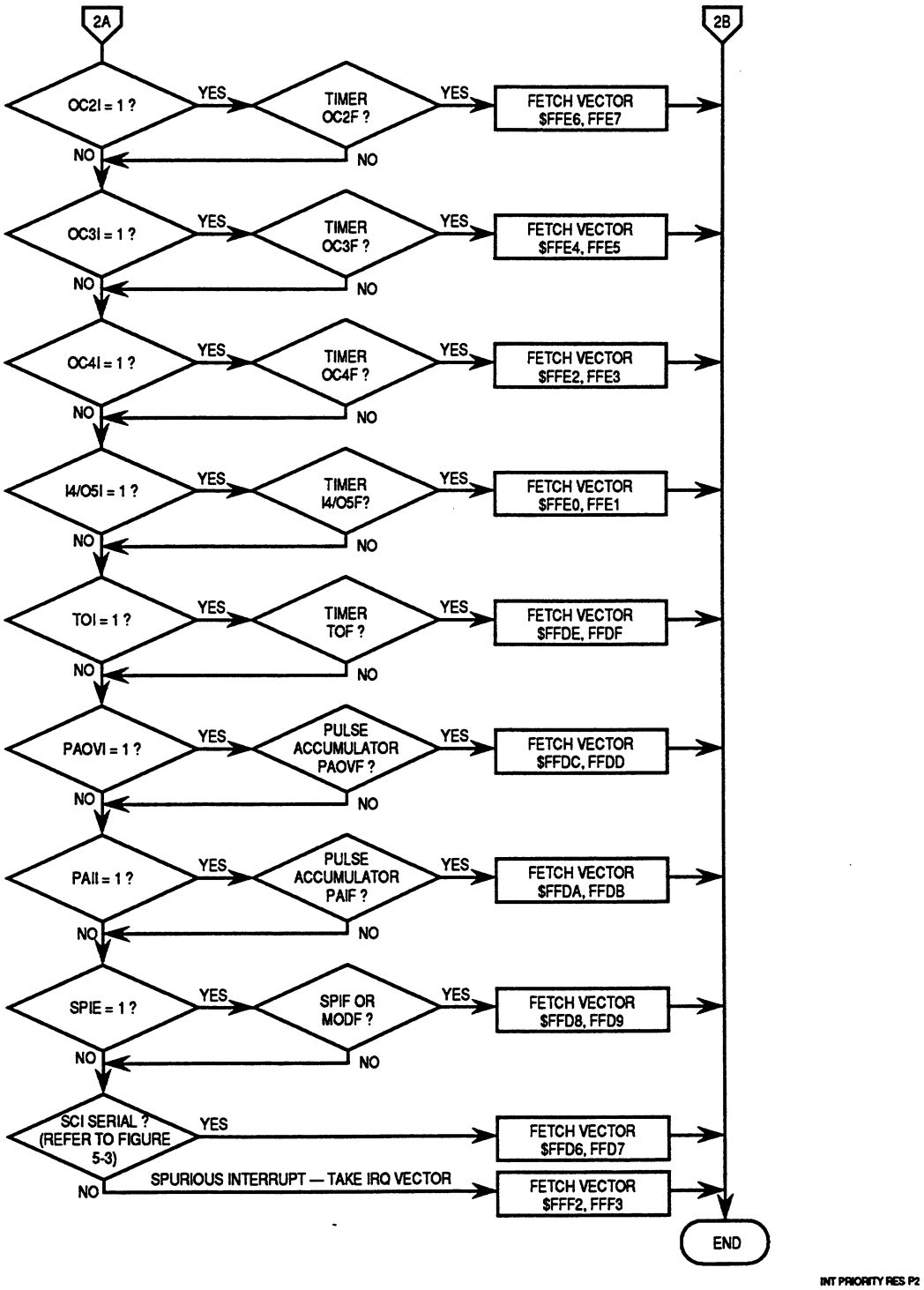


Figure 5–2. Interrupt Priority Resolution (2 of 2)

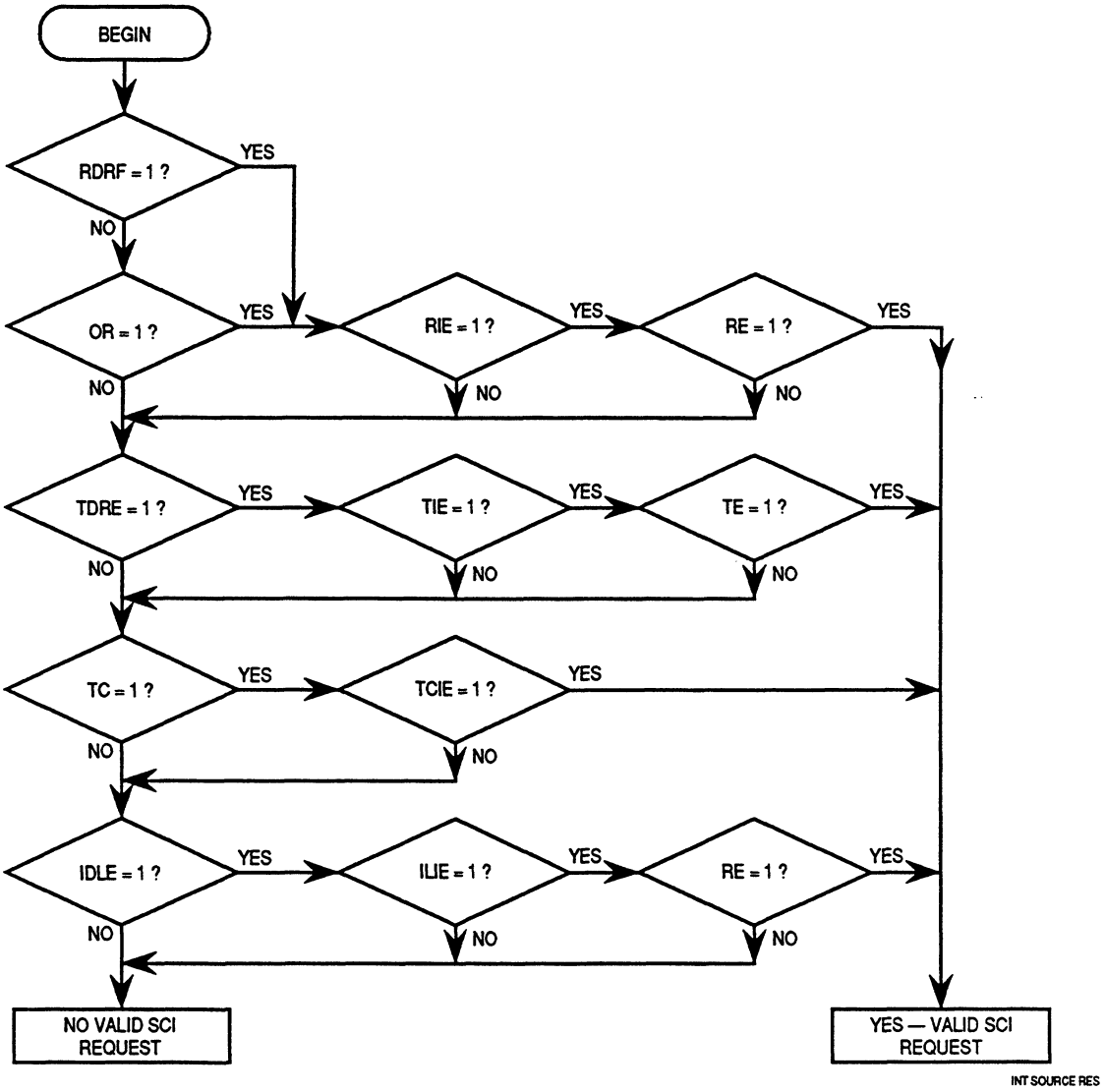


Figure 5–3. Interrupt Source Resolution Within SCI

5.5 Low Power Operation

Both STOP and WAIT suspend CPU operation until a reset or interrupt occurs. The WAIT condition suspends processing and reduces power consumption to an intermediate level. The STOP condition turns off all on-chip clocks and reduces power consumption to an absolute minimum while retaining the contents of all 768 bytes of RAM.

5.5.1 WAIT

The WAI opcode places the MCU in the WAIT condition, during which the CPU registers are stacked and CPU processing is suspended until a qualified interrupt is detected. The interrupt can be an external $\overline{IRQ}$, an $\overline{XIRQ}$, or any of the internally generated interrupts, such as the timer or serial interrupts. The on-chip crystal oscillator remains active throughout the WAIT standby period.

The reduction of power in the WAIT condition depends on how many internal clock signals driving on-chip peripheral functions can be shut down. The CPU is always shut down during WAIT. While in the wait state, the address/data bus repeatedly runs read cycles to the address where the CCR contents were stacked. The MCU leaves the wait state when it senses any interrupt that has not been masked.

The free-running timer system is shut down only if the I bit is set to one and the COP system is disabled by NOCOP being set to one. Several other systems can also be in a reduced power consumption state depending on the state of software-controlled configuration control bits. Power consumption by the analog-to-digital (A/D) converter is not affected significantly by the WAIT condition. However, the A/D converter current can be eliminated by writing the ADPU bit to zero. The SPI system is enabled or disabled by the SPE control bit. The SCI transmitter is enabled or disabled by the TE bit, and the SCI receiver is enabled or disabled by the RE bit. Therefore the power consumption in WAIT is dependent on the particular application.

5.5.2 STOP

Executing the STOP instruction while the S bit in the CCR is equal to zero places the MCU in the STOP condition. If the S bit is not zero, the STOP opcode is treated as a no-op (NOP). The STOP condition offers minimum power consumption because all clocks, including the crystal oscillator, are stopped while in this mode. To exit STOP and resume normal processing, a logic low level must be applied to one of the external interrupts ($\overline{IRQ}$ or $\overline{XIRQ}$) or to the $\overline{RESET}$ pin. A pending edge-triggered $\overline{IRQ}$ can also bring the CPU out of STOP.

Because all clocks are stopped in this mode, all internal peripheral functions also stop. The data in the internal RAM is retained as long as V_{DD} power is maintained. The CPU state and I/O pin levels are static and are unchanged by STOP. Therefore, when an interrupt comes to restart the system, the MCU resumes processing as if there were no interruption. If reset is used to restart the system a normal reset sequence results where all I/O pins and functions are also restored to their initial states.

To use the $\overline{IRQ}$ pin as a means of recovering from STOP, the I bit in the CCR must be clear ($\overline{IRQ}$ not masked). The $\overline{XIRQ}$ pin can be used to wake up the MCU from STOP regardless of the state of the X bit in the CCR, although the recovery sequence depends on the state of the X bit. If X is set to zero ($\overline{XIRQ}$ not masked), the MCU starts up, beginning with the stacking sequence leading to normal service of the $\overline{XIRQ}$ request. If X is set to one ($\overline{XIRQ}$ masked or inhibited), then processing continues with the instruction that immediately follows the STOP instruction, and no $\overline{XIRQ}$ interrupt service is requested or pending.

Because the oscillator is stopped in STOP mode, a restart delay may be imposed to allow oscillator stabilization upon leaving STOP. If the internal oscillator is being used, this delay is required; however, if a stable external oscillator is being used, the DLY control bit can be used to bypass this startup delay. The DLY control bit is set by reset and can be optionally cleared during initialization. If the DLY equal to zero option is used to avoid startup delay on recovery from STOP, then reset should not be used as the means of recovering from STOP, as this causes DLY to be set again by reset, imposing the restart delay. This same delay also applies to power-on-reset, regardless of the state of the DLY control bit, but does not apply to a reset while the clocks are running.

SECTION 6
PARALLEL INPUT/OUTPUT

The M68HC11 N-series MCUs have up to 62 input/output lines, depending on the operating mode. The data bus of this microcontroller is nonmultiplexed. I/O lines are organized into eight parallel ports. Ports with bidirectional pins have an associated data direction control register. This register (DDRx) contains a data direction control bit for each port line that can be bidirectional. Ports B, F, G, and H have an internal pull-up device for each port pin. These internal pull-up resistors are enabled using bits contained in the port pull-up assignment register (PPAR). The following table is a summary of the configuration and features of each port.

Table 6–1. Port Configuration

Port	Input Pins	Output Pins	Bidirectional Pins	Shared Functions
Port A	—	—	8	Timer
Port B	—	—	8	High-Order Address
Port C	—	—	8	Data Bus
Port D	—	—	6	SCI and SPI
Port E	8	—	—	A/D Converter
Port F	—	—	8	Low-Order Address
Port G	6	—	2	A/D and D/A Converters
Port H	—	—	8	PWM

NOTE

Do not confuse pin function with the electrical state of the pin at reset. All general-purpose I/O pins configured as inputs at reset are in a high-impedance state and the contents of port data registers is undefined. In port descriptions, a "U" indicates this condition. The pin function is mode dependent.

6.1 Port A

Port A has eight bidirectional I/O pins and shares functions with the timer system.

PORTA — Port A Data **$0000**

	Bit 7	6	5	4	3	2	1	Bit 0
	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
RESET:	U	U	U	U	U	U	U	U
Alt. Pin Func.:	PAI	OC2	OC3	OC4	IC4/OC5	IC1	IC2	IC3
And/or:	OC1	OC1	OC1	OC1	OC1	—	—	—

DDRA — Data Direction Register for Port A **$0001**

	Bit 7	6	5	4	3	2	1	Bit 0
	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0
RESET:	0	0	0	0	0	0	0	0

DDA[7:0] — Data Direction for Port A
0 = Corresponding I/O pin configured for input only
1 = Corresponding I/O pin configured for output

NOTE

To enable PA3 as fourth input capture, set the I4/O5 bit in the PACTL register. Otherwise, PA3 is configured as a fifth output compare out of reset, with bit I4/O5 being cleared. If the DDA3 bit is set (configuring PA3 as an output), and IC4 is enabled, writes to PA3 cause edges on the pin to result in input captures. Writing to TI4/O5 has no effect when the TI4/O5 register is acting as IC4. PA7 drives the pulse accumulator input but also can be configured for general-purpose I/O, or output compare. Note that even when PA7 is configured as an output, the pin still drives the pulse accumulator input.

6.2 Port B

Reset state is mode dependent. In single-chip or bootstrap modes, port B pins are high-impedance inputs with selectable internal pull-up resistors. BPPUE bit in PPAR register enables or disables port B pull-up devices. In expanded and test modes, port B pins are high-order address outputs, PORTB is not in the memory map, and pull-up devices have no effect on port B pins.

PORTB — Port B Data **$0004**

	Bit 7	6	5	4	3	2	1	Bit 0
	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
S. Chip or Boot:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
RESET:	U	U	U	U	U	U	U	U
Expan. or Test:	ADDR15	ADDR14	ADDR13	ADDR12	ADDR11	ADDR10	ADDR9	ADDR8

DDRB — Data Direction Register for Port B **$0002**

	Bit 7	6	5	4	3	2	1	Bit 0
	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0
RESET:	0	0	0	0	0	0	0	0

DDB[7:0] — Data Direction for Port B
0 = Corresponding I/O pin configured for input only
1 = Corresponding I/O pin configured for output

6.3 Port C

Reset state is mode dependent. In single-chip and bootstrap modes, port C pins are high-impedance inputs. It is customary to have an external pull-up resistor on lines that are driven by open-drain devices. In expanded or test modes, port C pins are data bus inputs and outputs, PORTC is not in the memory map and the R/$\overline{W}$ signal (PG7/R/$\overline{W}$) controls the direction of data.

PORTC — Port C Data **$0006**

	Bit 7	6	5	4	3	2	1	Bit 0
	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
S. Chip or Boot:	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
RESET:	U	U	U	U	U	U	U	U
Expan. or Test:	DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0

DDRC — Data Direction Register for Port C **$0007**

	Bit 7	6	5	4	3	2	1	Bit 0
	DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0
RESET:	0	0	0	0	0	0	0	0

DDC[7:0] — Data Direction for Port C
0 = Corresponding I/O pin configured for input only
1 = Corresponding I/O pin configured for output

6.4 Port D

In all modes, port D bits [5:0] can be used either for general-purpose I/O, or with the SCI and SPI subsystems. During reset, port D pins are configured as high impedance inputs (DDRD bits cleared).

PORTD — Port D Data **$0008**

	Bit 7	6	5	4	3	2	1	Bit 0
	—	—	PD5	PD4	PD3	PD2	PD1	PD0
RESET:	0	0	U	U	U	U	U	U
Alt. Pin Func.:	—	—	$\overline{SS}$	SCK	MOSI	MISO	TxD	RxD

DDRD — Data Direction Register for Port D **$0009**

	Bit 7	6	5	4	3	2	1	Bit 0
	—	—	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0
RESET:	0	0	0	0	0	0	0	0

Bits [7:6] — Not implemented
Always read zero

DDD[5:0] — Data Direction for Port D
0 = Corresponding I/O pin configured for input only
1 = Corresponding I/O pin configured for output

NOTE

When the SPI system is in slave mode, DDD5 has no meaning nor effect. When the SPI system is in master mode, DDD5 determines whether bit 5 of PORTD is an error detect input (DDD5 = 0) or a general-purpose output (DDD5 = 1). If the SPI system is enabled and expects any of bits [4:2] to be an input, that bit will be an input regardless of the state of the associated DDR bit. If any of bits [4:2] are expected to be outputs that bit will be an output **only** if the associated DDR bit is set.

6.5 Port E

Port E has eight general-purpose input pins and shares functions with the A/D converter system. When some port E pins are being used for general-purpose input and others are being used as A/D inputs, PORTE should not be read during the sample portion of an A/D conversion.

PORTE — Port E Data **$000A**

	Bit 7	6	5	4	3	2	1	Bit 0
	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
RESET:	U	U	U	U	U	U	U	U
Alt. Pin Func.:	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0

6.6 Port F

Reset state is mode dependent. In single-chip or bootstrap modes, port F pins are high-impedance inputs with selectable internal pull-up resistors. FPPUE bit in PPAR register enables or disables port F pull-up devices. In expanded and test modes, port F pins are low order address outputs, PORTF is not in the memory map, and pull-up devices have no effect on port F pins.

PORTF — Port F Data **$0005**

	Bit 7	6	5	4	3	2	1	Bit 0
	PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0
S. Chip or Boot:	PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0
RESET:	U	U	U	U	U	U	U	U
Expan. or Test:	ADDR7	ADDR6	ADDR5	ADDR4	ADDR3	ADDR2	ADDR1	ADDR0

DDRF — Data Direction Register for Port F **$0003**

	Bit 7	6	5	4	3	2	1	Bit 0
	DDF7	DDF6	DDF5	DDF4	DDF3	DDF2	DDF1	DDF0
RESET:	0	0	0	0	0	0	0	0

DDF[7:0] — Data Direction for Port F

0 = Corresponding I/O pin configured for input only

1 = Corresponding I/O pin configured for output

6.7 Port G

Port G pins reset to high-impedance inputs. In expanded and special test modes PG7 becomes R/$\overline{W}$. Alternate functions for port G bits [5:4] are digital-to-analog converter outputs. Port G bits [5:4] are input only unless an associated D/A channel is enabled, in which case bits [5:4] are outputs from the D/A converter. Alternate functions for port G bits [3:0] are analog-to-digital converter inputs (AN[11:8]). Port G bits [7:6] are bidirectional and have corresponding DDR bits. Port G pins 6 and 7 include on-chip pull-up devices. GPPUE bit in PPAR register enables or disables port G pull-up devices. Bit 6 is always general-purpose input/output.

PORTG — Port G Data **$007E**

	Bit 7	6	5	4	3	2	1	Bit 0
	PG7	PG6	PG5	PG4	PG3	PG2	PG1	PG0
RESET:	0	0	0	0	0	0	0	0
Alt. Pin Func.:	R/$\overline{W}$	—	DA2	DA1	AN11	AN10	AN9	AN8

DDRG — Data Direction Register for Port G **$007F**

	Bit 7	6	5	4	3	2	1	Bit 0
	DDG7	DDG6	—	—	—	—	—	—
RESET:	0	0	0	0	0	0	0	0

DDG[7:6] — Data direction for port G
0 = Corresponding I/O pin configured for input only
1 = Corresponding I/O pin configured for output

Bits [5:0] — Not implemented
Always read zero

6.8 Port H

Port H pins reset to high-impedance inputs with selectable internal pull-up resistors. Alternate functions for port H bits [5:0] are PWM timer outputs.

PORTH — Port H Data **$007C**

	Bit 7	6	5	4	3	2	1	Bit 0
	PH7	PH6	PH5	PH4	PH3	PH2	PH1	PH0
RESET:	0	0	0	0	0	0	0	0
Alt. Pin Func.:	—	—	PW6	PW5	PW4	PW3	PW2	PW1

DDRH — Data Direction Register for Port H **$007D**

	Bit 7	6	5	4	3	2	1	Bit 0
	DDH7	DDH6	DDH5	DDH4	DDH3	DDH2	DDH1	DDH0
RESET:	0	0	0	0	0	0	0	0

DDH[7:0] — Data direction for port H
0 = Corresponding I/O pin configured for input only
1 = Corresponding I/O pin configured for output

NOTE

In any mode, PWM circuitry forces the I/O state to be an output for each port H line associated with an enabled pulse-width modulator channel. In these cases, data direction bits are not changed and have no effect on these lines. DDRH reverts to controlling the I/O state of a pin when the associated PWM channel is disabled. Refer to **SECTION 9 TIMING SYSTEM** for further information.

6.9 Port Pull-Up Assignment Register

The port pull-up assignment register controls whether or not on-chip pull-up devices are to be used for the associated I/O port. I/O ports affected by this register are ports B, F, G, and H.

PPAR — Port Pull-Up Assignment **$002C**

	Bit 7	6	5	4	3	2	1	Bit 0
	—	—	—	—	HPPUE	GPPUE	FPPUE	BPPUE
RESET:	0	0	0	0	1	1	1	1

Bits [7:4] — Not implemented
Always read zero

xPPUE — Port x Pin Pull-Up Enable
Refer to PAREN bit in CONFIG register
0 = Port x pin on-chip pull-up devices disabled
1 = Port x pin on-chip pull-up devices enabled

NOTE

FPPUE and BPPUE do not apply in expanded mode because port F and port B are address outputs.

6.10 System Configuration Options 2

The system configuration options 2 register controls several configuration parameters. Bit 6, CWOM, is the only bit in this register that directly affects parallel I/O.

OPT2 — System Configuration Options 2 **$0038**

	Bit 7	6	5	4	3	2	1	Bit 0
	LIRDV	CWOM	STRCH	IRVNE	LSBF	SPR2	—	—
RESET:	0	0	0	—	0	0	0	0

LIRDV — LIR Driven
Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY.**

CWOM — Port C Wired-OR Mode
0 = Port C operates normally
1 = Port C outputs are open drain

STRCH — Clock Stretch for External Accesses
Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY.**

IRVNE — Internal Read Visibility/Not E
Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY.**

LSBF — SPI LSB First Enable
Refer to **SECTION 8 SERIAL PERIPHERAL INTERFACE.**

SPR2 — SPI Clock Rate Select
Refer to **SECTION 8 SERIAL PERIPHERAL INTERFACE.**

Bits [1:0] — Not implemented
Always read zero

6.11 System Configuration Register

The system configuration register controls several configuration parameters. Bit 4, PAREN, is the only bit in this register that directly affects parallel I/O.

CONFIG — System Configuration Register **$003F**

	Bit 7	6	5	4	3	2	1	Bit 0
	ROMAD	—	—	PAREN	NOSEC	NOCOP	ROMON	EEON
RESET:	—	1	1	—	1	—	—	—

ROMAD — ROM Mapping Control
Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY.**

Bits [6:5] — Not implemented
Always read one

PAREN — Pull-Up Assignment Register Enable
0 = PPAR register disabled
1 = PPAR register enabled; pull-ups can be enabled through PPAR

NOSEC — RAM and EEPROM Security Disabled
Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY.**

NOCOP — COP System Disable
Refer to **SECTION 5 RESETS AND INTERRUPTS.**

ROMON — ROM Enable
Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY.**

EEON — EEPROM Enable
Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY.**

SECTION 7
SERIAL COMMUNICATIONS INTERFACE

The serial communications interface (SCI) is a universal asynchronous receiver transmitter (UART), one of two independent serial I/O subsystems in the M68HC11 N-series MCUs. It has a standard nonreturn to zero (NRZ) format (one start bit, eight or nine data bits, and one stop bit). The SCI features enabled on this MCU include the following:

- 13-bit modulus prescaler
- Idle line detect
- Receiver-active flag
- Transmitter and receiver hardware parity

The enhanced baud rate generator is shown in the following diagram. Refer to the table of SCI baud rate control values for standard values and methods of calculation.

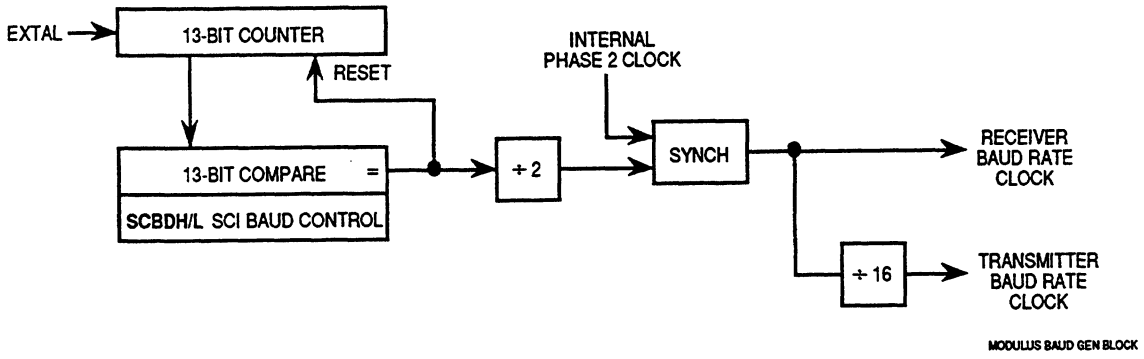


Figure 7–1. SCI Baud Generator Circuit Diagram

7.1 Data Format

The serial data format requires the following conditions:

1. An idle-line condition before transmission or reception of a message.
2. A start bit, logic zero, transmitted or received, that indicates the start of each character.
3. Data that is transmitted and received least significant bit (LSB) first.
4. A stop bit, logic one, 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 (defined as the transmission or reception of a logic zero for some multiple number of frames).

Selection of the word length is controlled by the M bit of SCCR1.

7.2 Transmit Operation

The SCI transmitter includes a parallel data register (SCDRH/SCDRL) and a serial shift register. The contents of the shift register can only be written through the serial data registers. This double buffered operation allows a character to be shifted out serially while another character is waiting in the parallel data registers to be transferred into the shift register. The output of the shift register is applied to TxD as long as transmission is in progress or the transmit enable (TE) bit of serial communication control register 2 (SCCR2) is set. The block diagram, Figure 7–2, shows the transmit serial shift register and the SCI data register at the top of the figure.

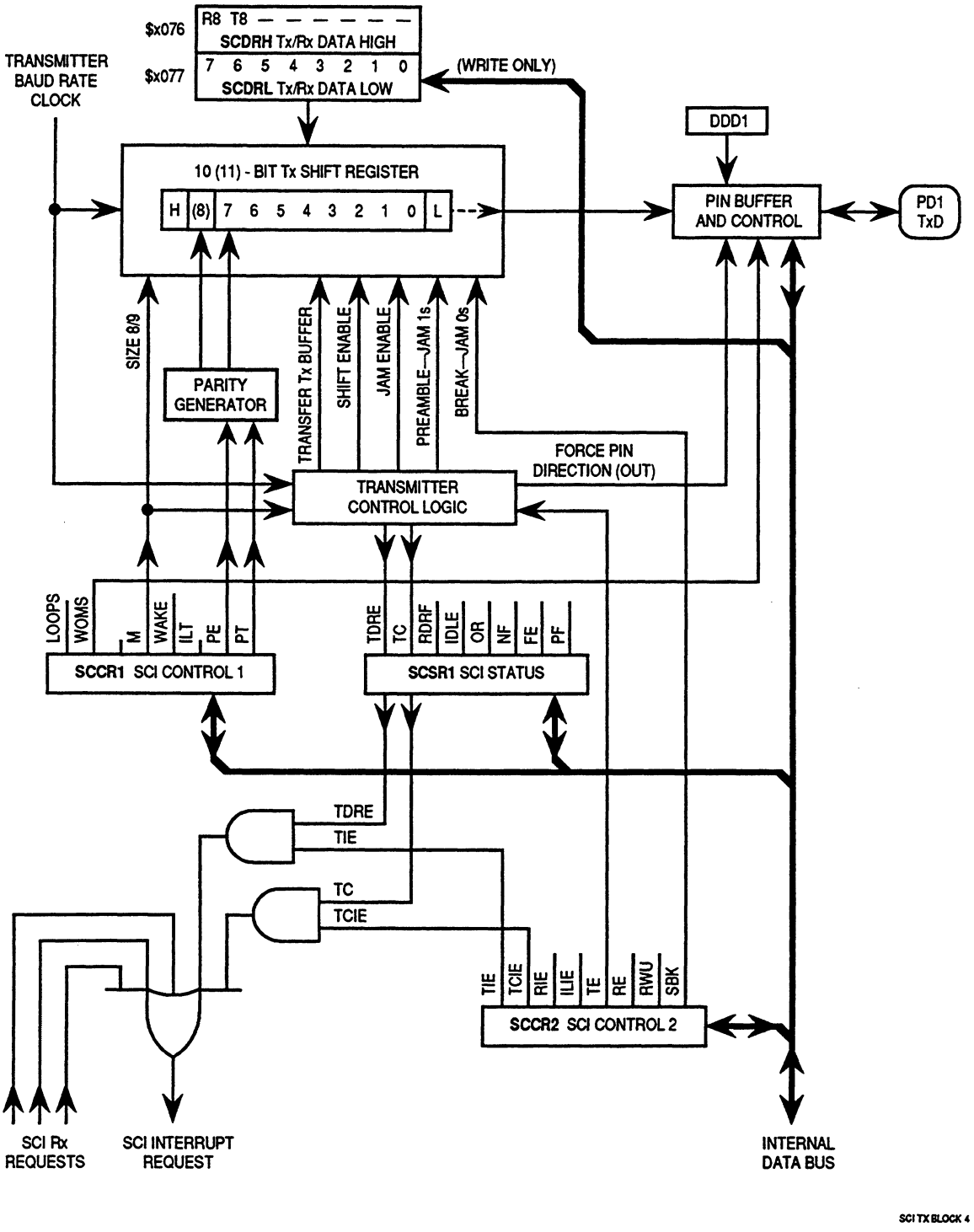


Figure 7–2. SCI Transmitter Block Diagram

7.3 Receive Operation

During receive operations, the transmit sequence is reversed. The serial shift register receives data and transfers it to the parallel receive data registers (SCDRH/SCDRL) as a complete word. This double buffered operation allows a character to be shifted in serially while another character is still in the parallel data registers. An advanced data recovery scheme distinguishes valid data from noise in the serial data stream. The data input is selectively sampled to detect receive data, and majority sampling logic determines the value and integrity of each bit.

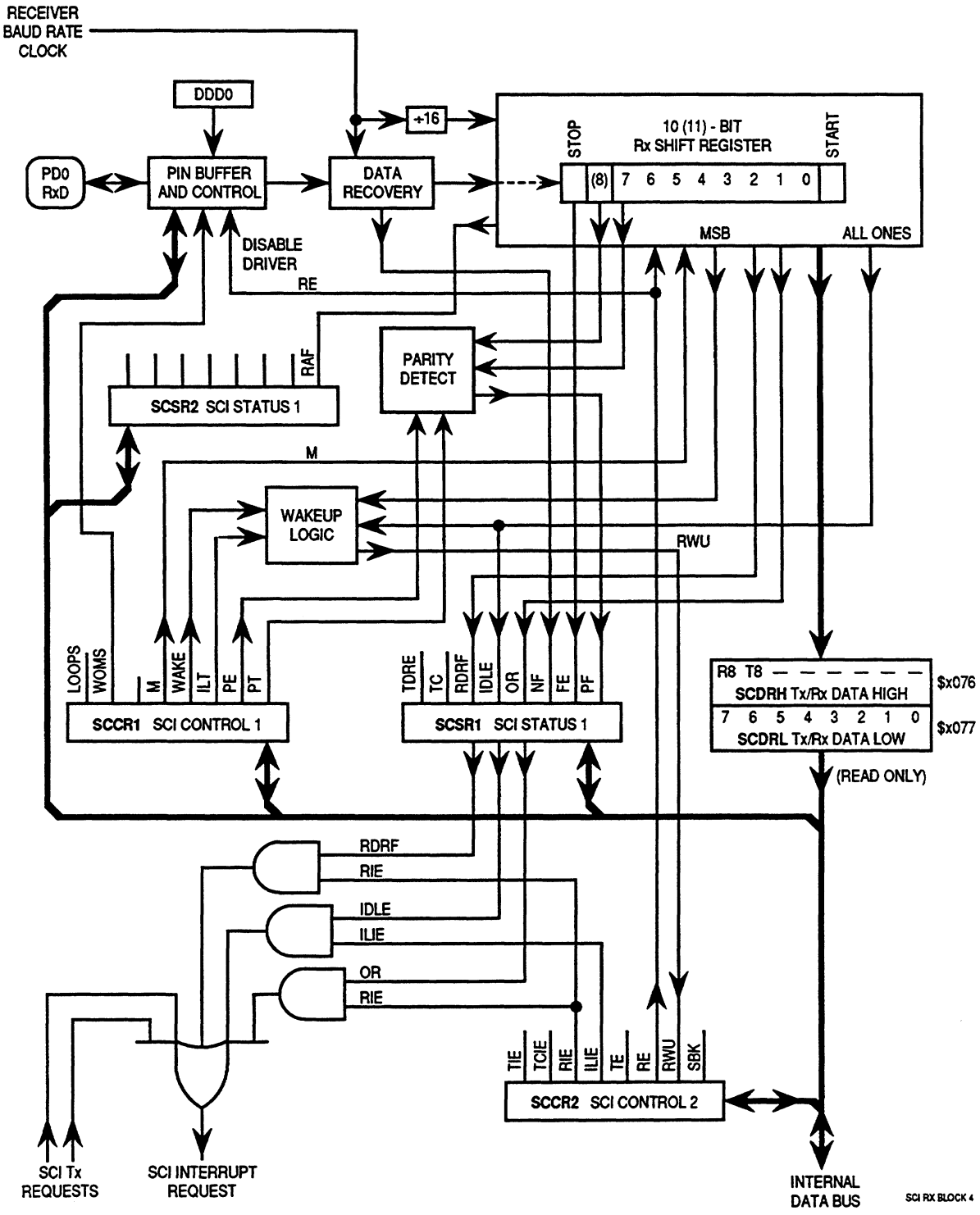


Figure 7–3. SCI Receiver Block Diagram

7.4 Wakeup Feature

The wakeup feature reduces SCI service overhead in multiple receiver systems. Software for each receiver evaluates each message and each receiver responds only to messages that are addressed to it. Whenever a new message begins, logic alerts dormant receivers to wake up and evaluate the initial character of the new message. All receivers are placed in wakeup mode by writing a one to the RWU bit in the SCCR2 register. When RWU is set, the receiver-related status flags (RDRF, IDLE, OR, NF, FE, and PF) are inhibited (cannot be set). Although RWU can be cleared by a software write to SCCR2, to do so would be unusual. Normally RWU is set by software and is cleared automatically by hardware.

Two methods of wakeup are available: idle-line wakeup and address-mark wakeup. When idle-line wakeup is used, a dormant receiver activates as soon as the RxD line becomes idle. When address-mark wakeup is used, logic one in the MSB of a character activates all sleeping receivers.

To use either receiver wakeup method, establish a software addressing scheme to allow transmitting devices to direct messages to individual receivers or to groups of receivers. This addressing scheme can take any form as long as all transmitting and receiving devices are programmed to understand the same scheme.

7.4.1 Idle-Line Wakeup

Clearing the WAKE bit in SCCR1 register enables idle-line wakeup mode. In idle-line wakeup mode, all receivers are active (RWU bit in SCCR2 = 0) when each message begins. The first frame(s) of each message are addressing frames. Each receiver in the system must evaluate the addressing frame(s) of a message to determine whether a message is intended for it. When a receiver finds that the message is not intended for it, it sets its RWU bit and stops receiving the message. In this way, only addressed receivers process the remainder of the message, thus reducing software overhead. As soon as an idle line is detected by receiver logic, hardware automatically clears the RWU bit so that the first frame(s) of the next message can be evaluated by all receivers in the system. This type of receiver wakeup requires a minimum of one idle frame time between messages, and no idle time between frames within a message.

7.4.2 Address-Mark Wakeup

Setting the WAKE bit in SCCR1 register enables address-mark wakeup mode. The address-mark wakeup method uses the MSB of each frame to differentiate between address information (MSB = 1) and actual message data (MSB = 0). All frames consist of seven information bits (eight bits if M bit in SCCR1 = 1) and

an MSB which, when set to one, indicates an address frame. Each receiver in the system must evaluate marked frames to determine whether a message is intended for it. When a receiver finds that a message is not intended for it, it sets its RWU bit and stops receiving the message. In this way, only the addressed receivers process the remainder of the message, thus reducing software overhead. The first frame of the next message will have the MSB set, which will automatically clear the RWU bit and indicate addressing frame. The RWU bit is cleared before the stop bit for the marked frame is received. This method of wakeup allows messages to include idle times, however, there is a loss in efficiency due to the extra bit time required for the address bit in each frame.

7.5 SCI Error Detection

Four error conditions can occur during SCI configuration. These error conditions are: serial data register overrun, received bit noise, framing, and parity error. Four bits (OR, NF, FE, and PF) in serial communications status register 1 (SCSR1) indicate if one of these error conditions exists.

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

The noise flag (NF) bit is set if there is noise on any of the received bits, including the start and stop bits. The NF bit is not set until the RDRF flag is set. The NF bit is cleared when the SCSR is read (with FE equal to one) followed by a read of the SCI data registers.

When no stop bit is detected in the received data character, the framing error (FE) bit is set. FE is set at the same time as the RDRF. If the byte received causes both framing and overrun errors, the processor only recognizes the overrun error. The framing error flag inhibits further transfer of data into the SCI data registers 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 SCI data registers.

The parity error flag (PF) is set if received data has incorrect parity. The flag is cleared by a read of SCSR1 with PE set, followed by a read of SCDR.

7.6 SCI Registers

There are eight addressable registers in the SCI. SCBDH, SCBDL, SCCR1, and SCCR2 are control registers. The contents of these registers control functions within the SCI. The status registers SCSR1 and SCSR2 contain bits that indicate certain conditions within the SCI. SCDRH and SCDRL are SCI data registers. These double buffered registers are used for the transmission and reception of data, and are used to form the 9-bit data word for the SCI. If the SCI is being used with 7- or 8-bit data, only SCDRL needs to be accessed. Note that if 9-bit data format is used, the upper register should be written first to ensure that it is transferred to the transmitter shift register with the lower register.

7.6.1 Serial Communications Baud Rate High/Low (SCBDH/L)

The contents of this register determines the baud rate of the SCI.

SCBDH/L — SCI Baud Rate Control High/Low **$0070, $0071**

	Bit 7	6	5	4	3	2	1	Bit 0	
$0070	BTST	BSPL	—	SBR12	SBR11	SBR10	SBR9	SBR8	High
RESET:	0	0	0	0	0	0	0	0	
$0071	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0	Low
RESET:	0	0	0	0	0	1	0	0	

BTST — Baud Register Test (TEST)

BSPL — Baud Rate Counter Split (TEST)

Bit 5 — Not implemented
Always reads zero

SBR[12:0] — SCI Baud Rate Selects

Use the following formula to calculate SCI baud rate. Refer to the table of baud rate control values for example rates:

SCI baud rate = EXTAL ÷ [16 • (2 • BR)]

Where BR is the contents of SCBDH, L (BR = 1, 2, 3, ... 8191).

BR = 0 disables the baud rate generator.

Table 7–1. SCI Baud Rate Control Values

Target Baud Rate	Crystal Frequency (EXTAL)					
	8 MHz		12 MHz		16 MHz	
	Dec Value	Hex Value	Dec Value	Hex Value	Dec Value	Hex Value
110	2272	$08E0	3409	$0D51	4545	$11C1
150	1666	$0682	2500	$09C4	3333	$0D05
300	833	$0341	1250	$04E2	1666	$0682
600	416	$01A0	625	$0271	833	$0341
1200	208	$00D0	312	$0138	416	$01A0
2400	104	$0068	156	$009C	208	$00D0
4800	52	$0034	78	$004E	104	$0068
9600	26	$001A	39	$0027	52	$0034
19.2 K	13	$000D	20	$0014	26	$001A
38.4 K	—	—	—	—	13	$000D

7.6.2 Serial Communications Control Register 1 (SCCR1)

The SCCR1 register provides the control bits that determine word length and select the method used for the wakeup feature.

SCCR1 — SCI Control Register 1 **$0072**

	Bit 7	6	5	4	3	2	1	Bit 0
	LOOPS	WOMS	—	M	WAKE	ILT	PE	PT
RESET:	0	0	0	0	0	0	0	0

LOOPS — SCI LOOP Mode Enable

0 = SCI transmit and receive operate normally.

1 = SCI transmit and receive are disconnected from TxD and RxD pins, and transmitter output is fed back into the receiver input.

Both the transmitter and receiver must be enabled to use the LOOP mode. When the LOOP mode is enabled, the TxD pin is driven high (idle line state) if the transmitter is enabled.

WOMS — Wired-OR Mode for SCI Pins (PD1, PD0; See also DWOM bit in SPCR.)

0 = TxD and RxD operate normally

1 = TxD and RxD are open drains if operating as outputs

Bit 5 — Not implemented

Always reads zero

M — Mode (Select Character Format)

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

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

WAKE — Wakeup by Address Mark/Idle
0 = Wakeup by IDLE line recognition
1 = Wakeup by address mark (most significant data bit set)

ILT — Idle Line Type
0 = Short (SCI counts consecutive ones after start bit.)
1 = Long (SCI counts ones only after stop bit.)
This bit determines which of two types of idle line detection method is used by the SCI receiver. In short mode the stop bit and any bits that were ones before the stop bit will be considered as part of that string of ones, possibly resulting in erroneous or premature detection of an idle line condition. In long mode the SCI system does not begin counting ones until a stop bit is received.

PE — Parity Enable
0 = Parity disabled
1 = Parity enabled

PT — Parity Type
0 = Parity even (even number of ones causes parity bit to be zero, odd number of ones causes parity bit to be one.)
1 = Parity odd (odd number of ones causes parity bit to be zero, even number of ones causes parity bit to be one.)

7.6.3 Serial Communications Control Register 2 (SCCR2)

The SCCR2 register provides the control bits that enable or disable individual SCI functions.

SCCR2 — SCI Control Register 2 **$0073**

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

TIE — Transmit Interrupt Enable
0 = TDRE interrupts disabled
1 = SCI interrupt requested when TDRE status flag is set

TCIE — Transmit Complete Interrupt Enable
0 = TC interrupts disabled
1 = SCI interrupt requested when TC status flag is set

RIE — Receiver Interrupt Enable
0 = RDRF and OR interrupts disabled
1 = SCI interrupt requested when RDRF flag or the OR status flag is set

ILIE — Idle Line Interrupt Enable
0 = IDLE interrupts disabled
1 = SCI interrupt requested when IDLE status flag is set

TE — Transmitter Enable
0 = Transmitter disabled
1 = Transmitter enabled

RE — Receiver Enable
0 = Receiver disabled
1 = Receiver enabled

RWU — Receiver Wakeup Control
0 = Normal SCI receiver
1 = Wakeup enabled and receiver interrupts inhibited

SBK — Send Break
0 = Break generator off
1 = Break codes generated as long as SBK = 1

7.6.4 Serial Communication Status Register 1 (SCSR1)

The bits in SCSR1 indicate certain conditions in the SCI hardware and are automatically cleared by special acknowledge sequences.

SCSR1 — SCI Status Register 1 **$0074**

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

TDRE — Transmit Data Register Empty Flag
This flag is set when SCDR is empty. Clear the TDRE flag by reading SCSR1 with TDRE set and then writing to SCDR.
0 = SCDR busy
1 = SCDR empty

TC — Transmit Complete Flag
This flag is set when the transmitter is idle (no data, preamble, or break transmission in progress). Clear the TC flag by reading SCSR1 with TC set and then writing to SCDR.
0 = Transmitter busy
1 = Transmitter idle

RDRF — Receive Data Register Full Flag

Once cleared, IDLE is not set again until the RxD line has been active and becomes idle again. RDRF is set if a received character is ready to be read from SCDR. Clear the RDRF flag by reading SCSR1 with RDRF set and then reading SCDR.

0 = SCDR empty
1 = SCDR full

IDLE — Idle Line Detected Flag

This flag is set if the RxD line is idle. Once cleared, IDLE is not set again until the RxD line has been active and becomes idle again. The IDLE flag is inhibited when RWU = 1. Clear IDLE by reading SCSR1 with IDLE set and then reading SCDR.

0 = RxD line is active
1 = RxD line is idle

OR — Overrun Error Flag

OR is set if a new character is received before a previously received character is read from SCDR. Clear the OR flag by reading SCSR1 with OR set and then reading SCDR.

0 = No overrun
1 = Overrun detected

NF — Noise Error Flag

NF is set if majority sample logic detects anything other than a unanimous decision. Clear NF by reading SCSR1 with NF set and then reading SCDR.

0 = Unanimous decision
1 = Noise detected

FE — Framing Error

FE is set when a zero is detected where a stop bit was expected. Clear the FE flag by reading SCSR1 with FE set and then reading SCDR.

0 = Stop bit detected
1 = Zero detected

PF — Parity Error Flag

PF is set if received data has incorrect parity. Clear PF by reading SCSR1 with PE set and then reading SCDR.

0 = Parity correct
1 = Incorrect parity detected

7.6.5 Serial Communication Status Register 2 (SCSR2)

In the SCSR2 only bit 0 is used to indicate receiver active. The other seven bits always read zero.

SCSR2 — SCI Status Register 2 **$0075**

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

Bits [7:1] — Not implemented
Always read zero

RAF — Receiver Active Flag (Read Only)
0 = A character is not being received
1 = A character is being received

7.6.6 Serial Communications Data Register (SCDRH/L)

SCDRH/SCDRL is a parallel register that performs two functions. It is the receive data register when it is read, and the transmit data register when it is written. Reads access the receive data buffer and writes access the transmit data buffer. Data received or transmitted is double buffered.

SCDRH/L — SCI Data High/Low **$0076, $0077**

	Bit 7	6	5	4	3	2	1	Bit 0	
$0076	R8	T8	—	—	—	—	—	—	SCDRH (High)
$0077	R7/T7	R6/T6	R5/T5	R4/T4	R3/T3	R2/T2	R1/T1	R0/T0	SCDRL (Low)

R8 — Receiver Bit 8
Ninth serial data bit received when SCI is configured for a nine data bit operation

T8 — Transmitter Bit 8
Ninth serial data bit transmitted when SCI is configured for a nine data bit operation

Bits [5:0] — Not implemented
Always read zero

R/T[7:0] — Receiver/Transmitter Data Bits [7:0]
SCI data is double buffered in both directions.

7.7 Status Flags and Interrupts

SCI status flags can be read by software (polled) to tell when certain conditions exist. Alternatively, a local interrupt enable bit can be set to enable each of these status conditions to generate interrupt requests. Status flags are automatically set by hardware logic conditions, but must be cleared by software. This provides an interlock mechanism that enables logic to know when software has noticed the status indication. The software clearing sequence for these flags is automatic — functions that are normally performed in response to the status flags also satisfy the conditions of the clearing sequence.

7.7.1 Transmitter Flags

The SCI transmitter has two status flags. TDRE and TC flags are normally set when the transmitter is first enabled (TE set to one). The TDRE flag indicates there is room in the transmit queue to store another data character in the transmit data register. The TIE bit is the local interrupt mask for TDRE. When TIE is zero, TDRE must be polled. When TIE and TDRE are one, an interrupt is requested.

The TC flag indicates the transmitter has completed the queue. The TCIE bit is the local interrupt mask for TC. When TCIE is zero, TC must be polled; when TCIE is one and TC is one, an interrupt is requested.

Writing a zero to TE requests that the transmitter stop when it can. The transmitter completes any transmission in progress before actually shutting down. Only an MCU reset can cause the transmitter to stop and shut down immediately. If TE is cleared when the transmitter is already idle, the pin reverts to its general-purpose I/O function (synchronized to the bit-rate clock). If anything is being transmitted when TE is cleared, that character is completed before the pin reverts to general-purpose I/O, but any other characters waiting in the transmit queue are lost. The TC and TDRE flags are set at the completion of this last character, even though TE has been disabled.

7.7.2 Receiver Flags

The SCI receiver has seven status flags, three of which can generate interrupt requests. The status flags are set by the SCI logic in response to specific conditions in the receiver. These flags can be read (polled) at any time by software. Refer to Figure 7–4, which shows SCI interrupt arbitration.

When an overrun takes place, the new character is lost, and the character that was in its way in the parallel receive data register (RDR) is undisturbed. RDRF is set when a character has been received and transferred into the parallel RDR. The OR flag is set instead of RDRF if overrun occurs. A new character is ready to be transferred into the RDR before a previous character is read from the RDR.

The NF, FE, and PF flags provide additional information about the character in the RDR, but do not generate interrupt requests.

The receiver active flag (RAF) indicates that the receiver is busy.

The last receiver status flag and interrupt source come from the IDLE flag. The RxD line is idle if it has constantly been at logic one for a full character time. The IDLE flag is set only after the RxD line has been busy and becomes idle. This prevents repeated interrupts for the time RxD remains idle.

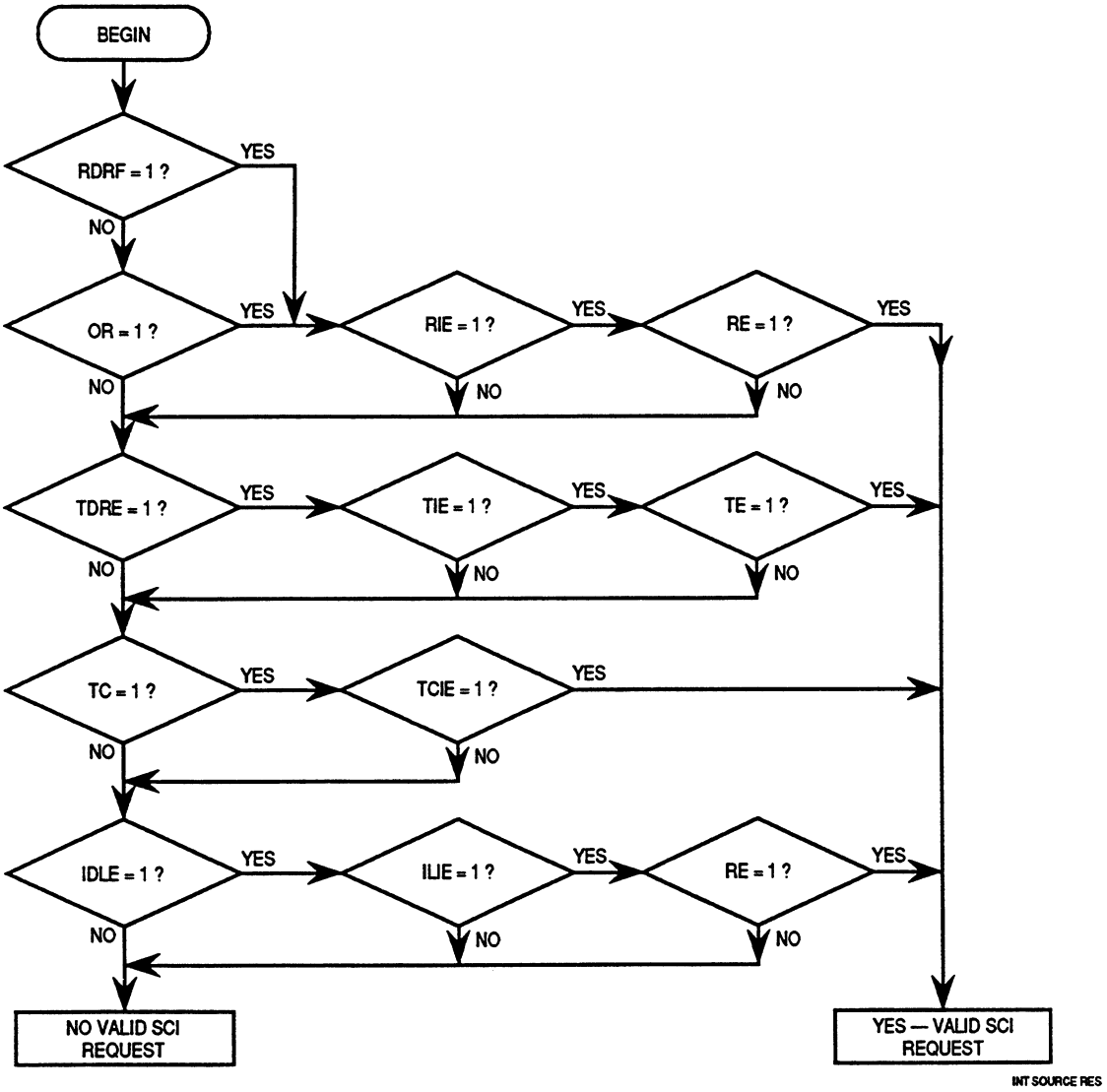


Figure 7–4. Interrupt Source Resolution Within SCI

SECTION 8
SERIAL PERIPHERAL INTERFACE

The serial peripheral interface (SPI), an independent serial communications subsystem, allows the MCU to communicate synchronously with peripheral devices, such as transistor-transistor logic (TTL) shift registers, liquid crystal display (LCD) drivers, analog-to-digital converter subsystems, and other microprocessors. The SPI is also capable of inter-processor communication in a multiple master system. The SPI system can be configured as either a master or a slave device with data rates as high as one half of the E-clock rate when configured as master and as fast as the E-clock rate when configured as slave.

8.1 Functional Description

The central element in the SPI system is the block containing the shift register and the read data buffer. The system is single buffered in the transmit direction and double buffered in the receive direction. This means that new data for transmission cannot be written to the shifter until the previous transfer is complete; however, received data is transferred into a parallel read data buffer so the shifter is free to accept a second serial character. As long as the first character is read out of the read data buffer before the next serial character is ready to be transferred, no overrun condition occurs. A single MCU register address is used for reading data from the read data buffer and for writing data to the shifter.

The SPI status block represents the SPI status functions (transfer complete, write collision, and mode fault) performed by the serial peripheral status register (SPSR). The SPI control block represents those functions that control the SPI system through the serial peripheral control register (SPCR).

Refer to Figure 8–1, which shows the SPI block diagram.

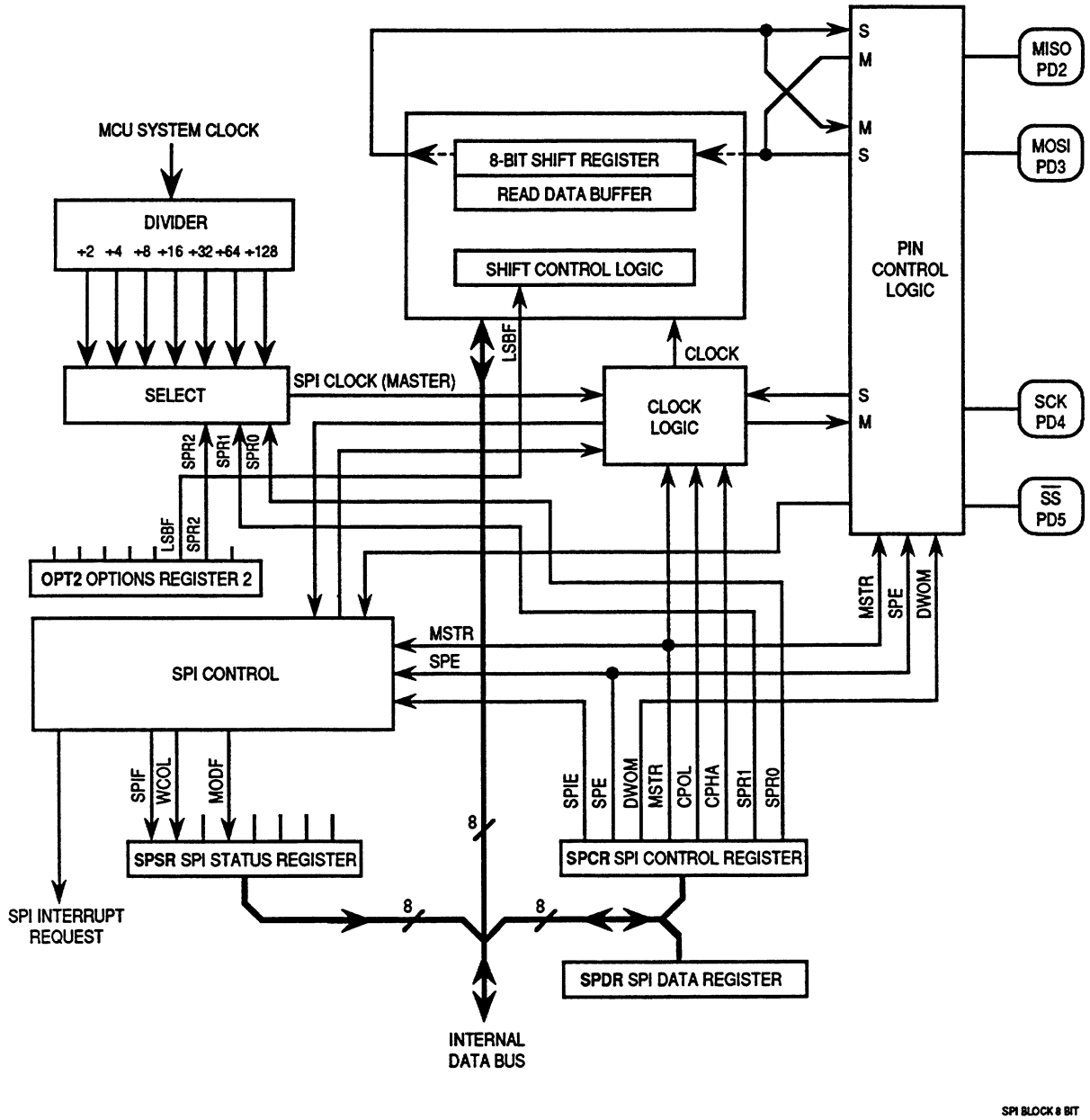


Figure 8–1. SPI Block Diagram

8.2 SPI Transfer Formats

During an SPI transfer, data is simultaneously transmitted and received. A serial clock line synchronizes shifting and sampling of the information on the two serial data lines. A slave select line allows individual selection of a slave SPI device; slave devices that are not selected do not interfere with SPI bus activities. On a master SPI device, the select line can optionally be used to indicate a multiple master bus contention. Refer to Figure 8–2.

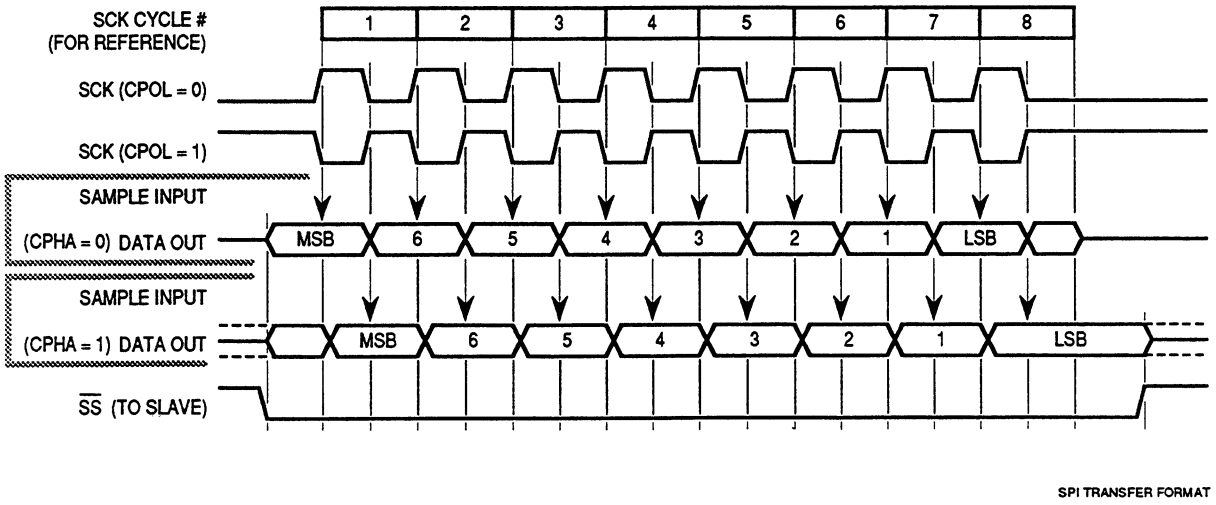


Figure 8–2. SPI Transfer Format

8.2.1 Clock Phase and Polarity Controls

Software can select one of four combinations of serial clock phase and polarity using two bits in the SPI control register (SPCR). The clock polarity is specified by the CPOL control bit, which selects an active high or active low clock, and has no significant effect on the transfer format. The clock phase (CPHA) control bit selects one of two different transfer formats. The clock phase and polarity should be identical for the master SPI device and the communicating slave device. In some cases, the phase and polarity are changed between transfers to allow a master device to communicate with peripheral slaves having different requirements.

When CPHA equals zero, the $\overline{SS}$ line must be deasserted and reasserted between each successive serial byte. Also, if the slave writes data to the SPI data register (SPDR) while $\overline{SS}$ is low, a write collision error results.

When CPHA equals one, the $\overline{SS}$ line can remain low between successive transfers.

8.3 SPI Signals

The following paragraphs contain descriptions of the four SPI signals: master in slave out (MISO), master out slave in (MOSI), serial clock (SCK), and slave select ($\overline{SS}$).

Any SPI output line must have its corresponding data direction bit in DDRD register set. If the DDR bit is clear, that line is disconnected from the SPI logic and becomes a general-purpose input. All SPI input lines are forced to act as inputs regardless of the state of the corresponding DDR bits in DDRD register.

8.3.1 Master In Slave Out

MISO is one of two unidirectional serial data signals. It is an input to a master device and an output from a slave device. The MISO line of a slave device is placed in the high-impedance state if the slave device is not selected.

8.3.2 Master Out Slave In

The MOSI line is the second of the two unidirectional serial data signals. It is an output from a master device and an input to a slave device. The master device places data on the MOSI line a half-cycle before the clock edge that the slave device uses to latch the data.

8.3.3 Serial Clock

SCK, an input to a slave device, is generated by the master device and synchronizes data movement in and out of the device through the MOSI and MISO lines. Master and slave devices are capable of exchanging a byte of information during a sequence of eight clock cycles.

There are four possible timing relationships that can 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 SPI clock rate select bits, SPR[1:0], in the SPCR of the master device, select the clock rate. In a slave device, SPR[1:0] have no effect on the operation of the SPI.

8.3.4 Slave Select

The slave select ($\overline{SS}$) input of a slave device must be externally asserted before a master device can exchange data with the slave device. $\overline{SS}$ must be low before data transactions and must stay low for the duration of the transaction.

The $\overline{SS}$ line of the master must be held high. If it goes low, a mode fault error flag (MODF) is set in the serial peripheral status register (SPSR). To disable the mode fault circuit, write a one in bit 5 of the port D data direction register. This sets the $\overline{SS}$ pin to act as a general-purpose output rather than the dedicated

input to the slave select circuit, thus inhibiting the mode fault flag. The other three lines are dedicated to the SPI whenever the serial peripheral interface is on.

The state of the master and slave CPHA bits affects the operation of $\overline{SS}$. CPHA settings should be identical for master and slave. 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, $\overline{SS}$ can be left low between successive SPI characters. In cases where there is only one SPI slave MCU, its $\overline{SS}$ line can be tied to V_{SS} as long as only CPHA = 1 clock mode is used.

8.4 SPI System Errors

Two system errors can be detected by the SPI system. The first type of error arises in a multiple-master system when more than one SPI device tries to be a master. This error is called a mode fault. The second type of error, write collision, indicates that an attempt was made to write data to the SPDR while a transfer was in progress.

When the SPI system is configured as a master and the $\overline{SS}$ input line goes to active low, a mode fault error has occurred — usually because two devices have attempted to act as master at the same time. In cases where more than one device is concurrently configured as a master, there is a chance of contention between two pin drivers. For push-pull CMOS drivers, this contention can cause permanent damage. The mode fault mechanism attempts to protect the device by disabling the drivers. The MSTR control bit in the SPCR and all four DDRD control bits associated with the SPI are cleared and an interrupt is generated subject to masking by the SPIE control bit and the I bit in the CCR.

Other precautions may need to be taken to prevent driver damage. If two devices are made masters at the same time, mode fault does not help protect either one unless one of them selects the other as slave. The amount of damage possible depends on the length of time both devices attempt to act as master.

A write collision error occurs if the SPDR is written while a transfer is in progress. Because the SPDR is not double buffered in the transmit direction, writes to SPDR cause data to be written directly into the SPI shift register. Because this write corrupts any transfer in progress, a write collision error is generated. The transfer continues undisturbed, and the write data that caused the error is not written to the shifter.

A write collision is normally a slave error because a slave has no control over when a master initiates a transfer. A master knows when a transfer is in progress, so there is no reason for a master to generate a write-collision error, although the SPI logic can detect write collisions in both master and slave devices.

The SPI configuration determines the characteristics of a transfer in progress. For a master, a transfer begins when data is written to SPDR and ends when SPIF is set. For a slave with CPHA equal to zero, a transfer starts when $\overline{SS}$ goes low and ends when $\overline{SS}$ returns high. In this case, SPIF is set at the middle of the eighth SCK cycle when data is transferred from the shifter to the parallel data register, but the transfer is still in progress until $\overline{SS}$ goes high. For a slave with CPHA equal to one, transfer begins when the SCK line goes to its active level, which is the edge at the beginning of the first SCK cycle. The transfer ends in a slave in which CPHA equals one when SPIF is set.

8.5 SPI Registers

The three SPI registers, SPCR, SPSR, and SPDR, provide control, status, and data storage functions. Refer to the following information for a description of how these registers are organized.

8.5.1 Serial Peripheral Control

SPCR — Serial Peripheral Control Register **$0028**

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

SPIE — Serial Peripheral Interrupt Enable

Set the SPE bit to one to request a hardware interrupt sequence each time the SPIF or MODF status flag is set. SPI interrupts are inhibited if this bit is clear or if the I bit in the condition code register is one.

SPE — Serial Peripheral System Enable

When the SPE bit is set, the port D bit 2, 3, 4, and 5 pins are dedicated to the SPI function. If the SPI is in the master mode and DDRD bit 5 is set, then the port D bit 5 pin becomes a general-purpose output instead of the $\overline{SS}$ input.

DWOM — Port D Wired-OR Mode

DWOM affects port D bit 5, 4, 3, 2 pins.

0 = Normal CMOS outputs

1 = 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, the SCK pin of the master device has a steady state low value. When CPOL is set, SCK idles high. Refer to Figure 8–2 and **8.2.1 Clock Phase and Polarity Controls**.

CPHA — Clock Phase

The clock phase bit, in conjunction with the CPOL bit, controls the clock-data relationship between master and slave. The CPHA bit selects one of two different clocking protocols. Refer to Figure 8–2 and **8.2.1 Clock Phase and Polarity Controls**.

SPR1 and SPR0 — SPI Clock Rate Selects

These two serial peripheral rate bits select the SPI clock rate. Refer to Table 8–1. Note that SPR2 is located in the OPT2 register and its reset state is zero.

Table 8–1. SPI Clock Rates

SPR[2:0]	E Clock Divide By	Frequency at E = 2 MHz (Baud)	Frequency at E = 3 MHz (Baud)	Frequency at E = 4 MHz (Baud)
0 0 0	2	1.0 MHz	1.5 MHz	2.0 MHz
0 0 1	4	500 kHz	750 kHz	1.0 MHz
0 1 0	16	125 kHz	187.5 kHz	250 kHz
0 1 1	32	62.5 kHz	93.7 kHz	125 kHz
1 0 0	8	250 kHz	375 kHz	500 kHz
1 0 1	16	125 kHz	187.5 kHz	250 kHz
1 1 0	64	31.3 kHz	46.9 kHz	62.5 kHz
1 1 1	128	15.6 kHz	23.4 kHz	31.3 kHz

8.5.2 Serial Peripheral Status

SPSR — Serial Peripheral Status Register **$0029**

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

SPIF — SPI Interrupt Complete Flag

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

WCOL — Write Collision

Clearing the WCOL bit is accomplished by reading the SPSR (with WCOL set) followed by an access of SPDR. Refer to **8.3.4 Slave Select** and **8.4 SPI System Errors**.

0 = No write collision
1 = Write collision

Bit 5 — Not implemented

Always reads zero

MODF — Mode Fault

To clear the MODF bit, read the SPSR (with MODF set), then write to the SPCR. Refer to **8.3.4 Slave Select** and **8.4 SPI System Errors**.

0 = No mode fault
1 = Mode fault

Bits [3:0] — Not implemented

Always read zero

8.5.3 Serial Peripheral Data I/O Register

The SPDR is used when transmitting or receiving data on the serial bus. Only a write to this register initiates transmission or reception of a byte, and this only occurs in the master device. At the completion of transferring a byte of data, the SPIF status bit is set in both the master and slave devices.

A read of the SPDR is actually a read of a buffer. To prevent an overrun and the loss of the byte that caused the overrun, the first SPIF must be cleared by the time a second transfer of data from the shift register to the read buffer is initiated.

SPDR — SPI Data Register **$002A**

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

SPI is double buffered in and single buffered out.

8.5.4 OPT2 Register

OPT2 — System Configuration Options 2 **$0038**

	Bit 7	6	5	4	3	2	1	Bit 0
	LIRDV	CWOM	STRCH	IRVNE	LSBF	SPR2	—	—
RESET:	0	0	0	—	0	0	0	0

LIRDV — LIR Driven
Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY.**

CWOM — Port C Wired-OR Mode
Refer to **SECTION 6 PARALLEL INPUT/OUTPUT.**

STRCH — Clock Stretch for External Accesses
Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY.**

IRVNE — Internal Read Visibility/Not E
Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY.**

LSBF — LSB First Enable
If this bit is set, data, which is usually transferred MSB first, is transferred LSB first. LSBF does not affect the position of the MSB and LSB in the data register. Reads and writes of the data register always have MSB in bit 7.

SPR2 — SPI Clock Rate Select Bit
SPR2 adds a divide-by-4 prescaler to the SPI clock chain. With the two bits in the SPCR, this bit specifies the SPI clock rate. Refer to Table 8–1.

Bits [1:0] — Not implemented
Always read zero

SECTION 9
TIMING SYSTEM

The M68HC11 timing system is composed of five clock divider chains. The main clock divider chain includes a 16-bit free-running counter, which is driven by a programmable prescaler. The main timer's programmable prescaler provides one of the four clocking rates to drive the 16-bit counter. Two prescaler control bits select the prescale rate.

The prescaler output divides the system clock by 1, 4, 8, or 16. Taps off of this main clocking chain drive circuitry that generates the slower clocks used by the pulse accumulator, the real-time interrupt (RTI), and the computer operating properly (COP) watchdog subsystems, also described in this section. Refer to Figure 9–1.

All main timer system activities are referenced to this free-running counter. The counter begins incrementing from $0000 as the MCU comes out of reset, and continues to the maximum count, $FFFF. At the maximum count, the counter rolls over to $0000, sets an overflow flag, and continues to increment. As long as the MCU is running in a normal operating mode, there is no way to reset, change, or interrupt the counting. The capture/compare subsystem features three input capture channels, four output compare channels, and one channel that can be selected to perform either input capture or output compare. Each of the three input capture functions has its own 16-bit input capture register (time capture latch) and each of the output compare functions has its own 16-bit compare register. All timer functions, including the timer overflow and RTI have their own interrupt controls and separate interrupt vectors.

The pulse accumulator contains an 8-bit counter and edge select logic. The pulse accumulator can operate in either event counting mode or gated time accumulation mode. During event counting mode, the pulse accumulator's 8-bit counter increments when a specified edge is detected on an input signal. During gated time accumulation mode, an internal clock source increments the 8-bit counter while an input signal has a predetermined logic level.

The real-time interrupt (RTI) is a programmable periodic interrupt circuit that permits pacing the execution of software routines by selecting one of four interrupt rates.

The COP watchdog clock input (E ÷ 2^{15}) is tapped off of the free-running counter chain. The COP automatically times out unless it is serviced within a specific time by a program reset sequence. If the COP is allowed to time out, a reset is generated, which drives the $\overline{\text{RESET}}$ pin low to reset the MCU and the external system. Refer to Table 9–1 for crystal related frequencies and periods.

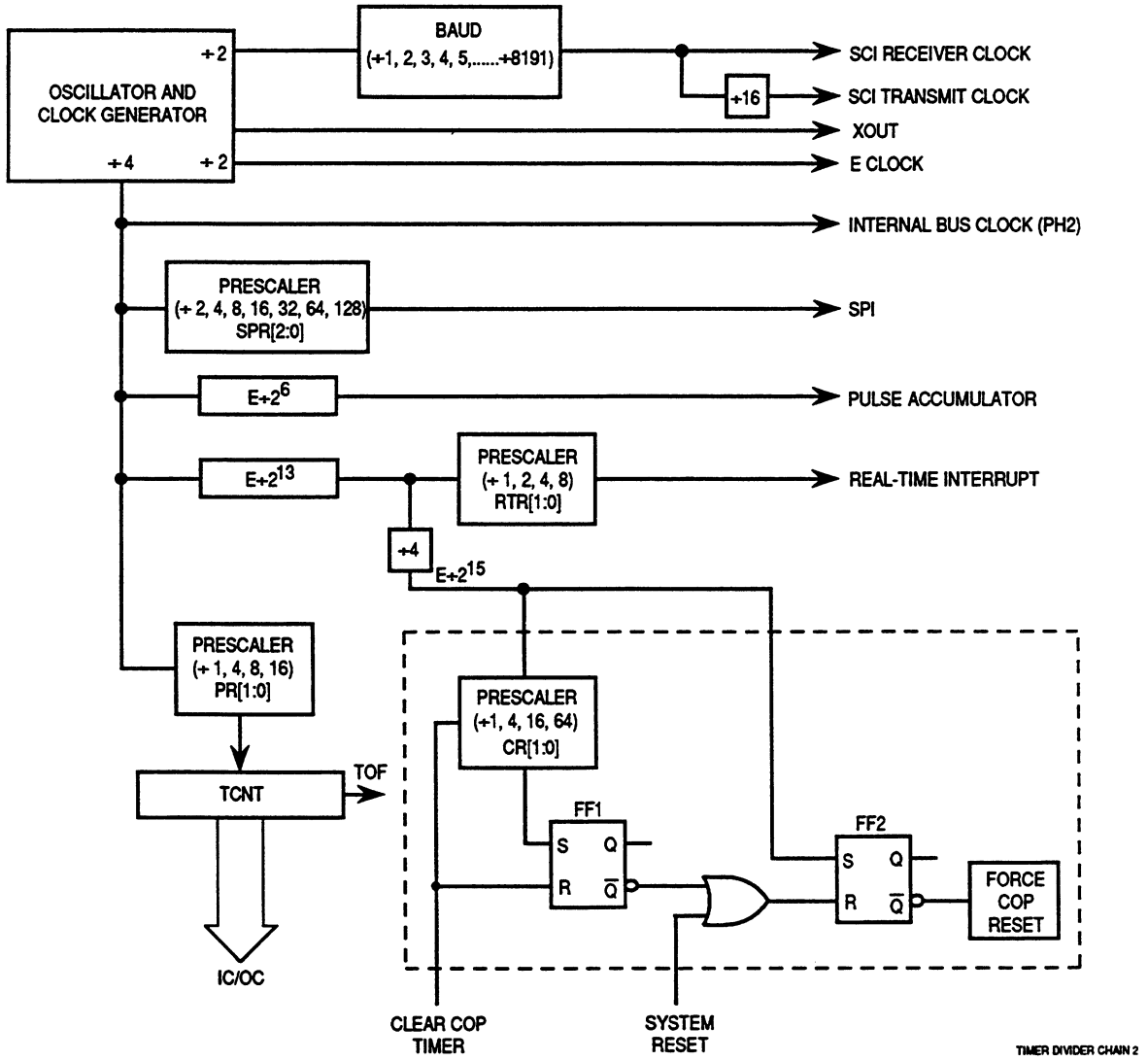


Figure 9–1. Timer Clock Divider Chains

Table 9–1. Timer Summary

Control Bits PR1, PR0	XTAL Frequencies				
	4.0 MHz	8.0 MHz	12.0 MHz	16.0 MHz	Other Rates
	1.0 MHz	2.0 MHz	3.0 MHz	4.0 MHz	(E)
	1000 ns	500 ns	333 ns	250 ns	(1/E)
	Main Timer Count Rates				
0 0 1 count — overflow —	1.0 µs 65.536 ms	500 ns 32.768 ms	333 ns 21.845 ms	250 ns 16.384 ms	(E/1) ($E/2^{16}$)
0 1 1 count — overflow —	4.0 µs 262.14 ms	2.0 µs 131.07 ms	1.333 µs 87.381 ms	1.0 µs 65.536 ms	(E/4) ($E/2^{18}$)
1 0 1 count — overflow —	8.0 µs 524.29 ms	4.0 µs 262.14 ms	2.667 µs 174.76 ms	2.0 µs 131.07 ms	(E/8) ($E/2^{19}$)
1 1 1 count — overflow —	16.0 µs 1.049 s	8.0 µs 524.29 ms	5.333 µs 349.52 ms	4.0 µs 262.14 ms	(E/16) ($E/2^{20}$)

9.1 Timer Structure

Figure 9–2 shows the capture/compare system block diagram. The port A pin control block includes logic for timer functions and for general-purpose I/O. For pins PA3, PA2, PA1, and PA0, this block contains both the edge-detection logic and the control logic that enables the selection of which edge triggers an input capture. The digital level on PA[3:0] can be read at any time (read PORTA register), even if the pin is being used for the input capture function. Pins PA[6:3] are used for either general-purpose I/O, or as output compare pins. When one of these pins is being used for an output compare function, it cannot be written directly as if it were a general-purpose output. Each of the output compare functions (OC[5:2]) is related to one of the port A output pins. Output compare one (OC1) has extra control logic, allowing it optional control of any combination of the PA[7:3] pins. The PA7 pin can be used as a general-purpose I/O pin, as an input to the pulse accumulator, or as an OC1 output pin.

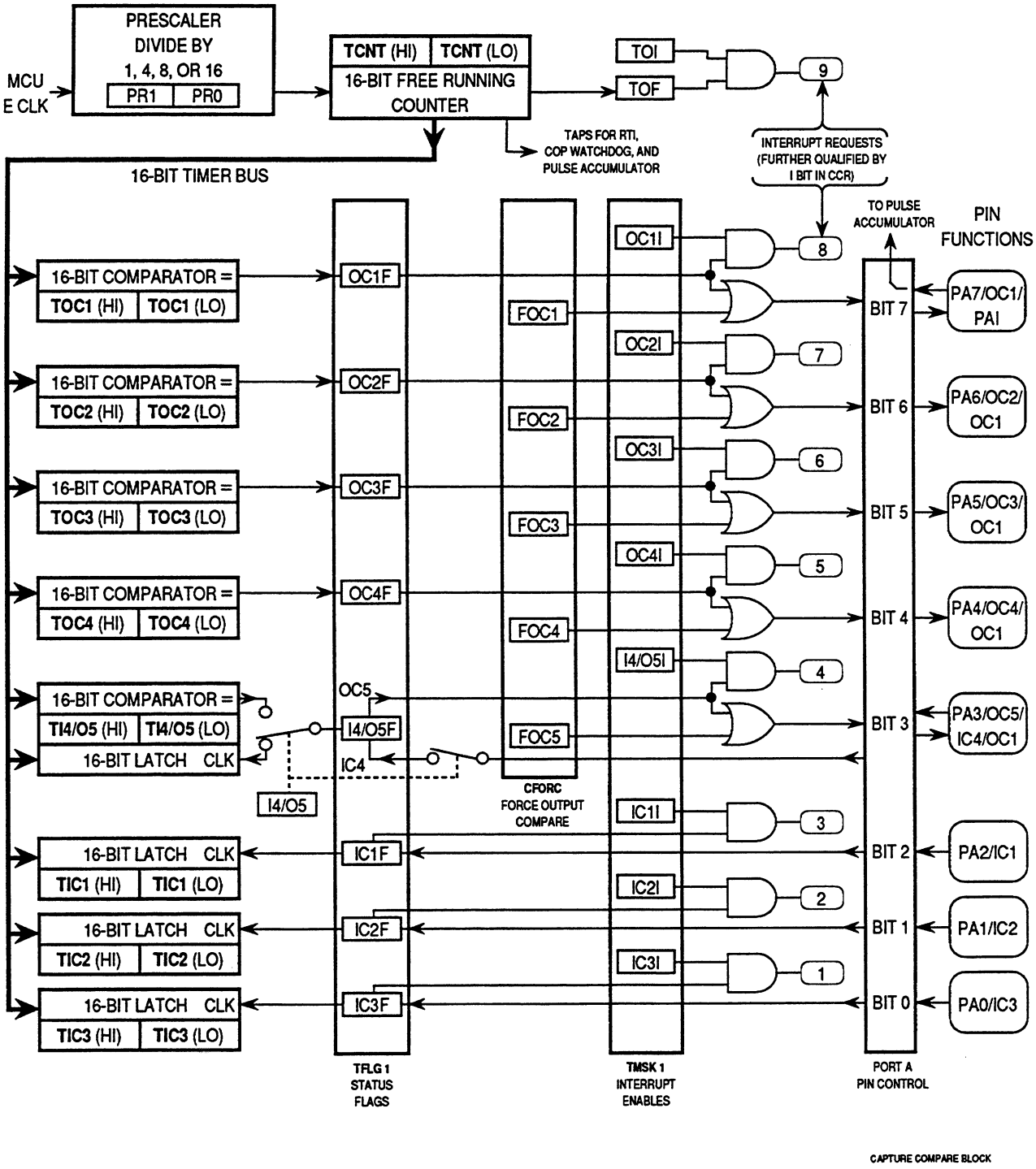


Figure 9–2. Capture/Compare Block Diagram

9.2 Input Capture

The input capture function records the time an external event occurs by latching the value of the free-running counter when a selected edge is detected at the associated timer input pin. Software can store latched values and use them to compute the periodicity and duration of events. For example, by storing the times of successive edges of an incoming signal, software can determine the period and pulse width of a signal. To measure period, two successive edges of the same polarity are captured. To measure pulse width, two alternate polarity edges are captured.

In most cases, input capture edges are asynchronous to the internal timer counter, which is clocked relative to an internal clock (PH2). These asynchronous capture requests are synchronized to PH2 so that the latching occurs on the opposite half cycle of PH2 from when the timer counter is being incremented. This synchronization process introduces a delay from when the edge occurs to when the counter value is detected. Because these delays offset each other when the time between two edges is being measured, the delay can be ignored. When an input capture is being used with an output compare, there is a similar delay between the actual compare point and when the output pin changes state.

The control and status bits that implement the input capture functions are contained in the PACTL, TCTL2, TMSK1, and TFLG1 registers.

To configure port A bit 3 as an input capture, clear the DDA3 bit of the DDRA register. Note that this bit is cleared out of reset. To enable PA3 as the fourth input capture, set the I4/O5 bit in the PACTL register. Otherwise, PA3 is configured as a fifth output compare out of reset, with bit I4/O5 being cleared. If the DDA3 bit is set (configuring PA3 as an output), and IC4 is enabled, then writes to PA3 cause edges on the pin to result in input captures. Writing to TI4/O5 has no effect when the TI4/O5 register is acting as IC4.

9.2.1 Timer Control Register 2

Use the control bits of this register to program input capture functions to detect a particular edge polarity on the corresponding timer input pin. Each of the input capture functions can be independently configured to detect rising edges only, falling edges only, any edge (rising or falling), or to disable the input capture function. The input capture functions operate independently of each other and can capture the same TCNT value if the input edges are detected within the same timer count cycle.

TCTL2 — Timer Control 2 **$0021**

	Bit 7	6	5	4	3	2	1	Bit 0
	EDG4B	EDG4A	EDG1B	EDG1A	EDG2B	EDG2A	EDG3B	EDG3A
RESET:	0	0	0	0	0	0	0	0

EDGxB and EDGxA — Input Capture Edge Control

There are four pairs of these bits. Each pair is cleared to zero by reset and must be encoded to configure the corresponding input capture edge detector circuit. IC4 functions only if the I4/O5 bit in the PACTL register is set. Refer to Table 9–2 for timer control configuration.

Table 9–2. Timer Control Configuration

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

9.2.2 Timer Input Capture Registers

When an edge has been detected and synchronized, the 16-bit free-running counter value is transferred into the input capture register pair as a single 16-bit parallel transfer. Timer counter value captures and timer counter incrementing occur on opposite half-cycles of the phase 2 clock so that the count value is stable whenever a capture occurs. The TICx registers are not affected by reset. Input capture values can be read from a pair of 8-bit read-only registers. A read of the high-order byte of an input capture register pair inhibits a new capture transfer for one bus cycle. If a double-byte read instruction, such as LDD, is used to read the captured value, coherency is assured. When a new input capture occurs immediately after a high-order byte read, transfer is delayed for an additional cycle but the value is not lost.

TIC1–TIC3 — Timer Input Capture **$0010–$0015**

$0010	Bit 15	14	13	12	11	10	9	Bit 8	TIC1 (High)
$0011	Bit 7	6	5	4	3	2	1	Bit 0	TIC1 (Low)
$0012	Bit 15	14	13	12	11	10	9	Bit 8	TIC2 (High)
$0013	Bit 7	6	5	4	3	2	1	Bit 0	TIC2 (Low)
$0014	Bit 15	14	13	12	11	10	9	Bit 8	TIC3 (High)
$0015	Bit 7	6	5	4	3	2	1	Bit 0	TIC3 (Low)

TICx not affected by reset.

9.2.3 Timer Input Capture 4/Output Compare 5 Register

Use TI4/O5 as either an input capture register or an output compare register, depending on the function chosen for the PA3 pin. To enable it as an input capture register, set the I4/O5 bit in the pulse accumulator control register (PACTL) to logic level one. To use it as an output compare register, set the I4/O5 bit to a logic level zero. Refer to **9.6 Pulse Accumulator**.

TI4/O5 — Timer Input Capture 4/Output Compare 5 **$001E, $001F**

$001E	Bit 15	14	13	12	11	10	9	Bit 8	TI4/O5 (High)
$001F	Bit 7	6	5	4	3	2	1	Bit 0	TI4/O5 (Low)

The TI4/O5 register pair resets to ones ($FFFF).

9.3 Output Compare

Use the output compare (OC) function to program an action to occur at a specific time — when the 16-bit counter reaches a specified value. For each of the five output compare functions, there is a separate 16-bit compare register and a dedicated 16-bit comparator. The value in the compare register is compared to the value of the free-running counter on every bus cycle. When the compare register matches the counter value, an output compare status flag is set. The flag can be used to initiate the automatic actions for that output compare function.

To produce a pulse of a specific duration, write a value to the output compare register that represents the time the leading edge of the pulse is to occur. The output compare circuit is configured to set the appropriate output either high or low, depending on the polarity of the pulse being produced. After a match occurs, the output compare register is reprogrammed to change the output pin back to its inactive level at the next match. A value representing the width of the pulse is added to the original value, and then written to the output compare register. Because the pin state changes occur at specific values of the free-running counter, the pulse width can be controlled accurately at the resolution of the free-running counter, independent of software latencies. To generate an output signal of a specific frequency and duty cycle, repeat this pulse-generating procedure.

There are four 16-bit read/write output compare registers: TOC1, TOC2, TOC3, and TOC4, and the TI4/O5 register, which functions under software control as either IC4 or OC5. Each of the OC registers is set to $FFFF on reset. A value written to an OC register is compared to the free-running counter value during each E-clock cycle. If a match is found, the particular output compare flag is set in timer interrupt flag register 1 (TFLG1). If that particular interrupt is enabled in the timer interrupt mask register 1 (TMSK1), an interrupt is generated. In

addition to an interrupt, a specified action can be initiated at one or more timer output pins. For OC[5:2], the pin action is controlled by pairs of bits (OMx and OLx) in the TCTL1 register. The output action is taken on each successful compare, regardless of whether or not the OCxF flag in the TFLG1 register was previously cleared.

OC1 is different from the other output compares in that a successful OC1 compare can affect any or all five of the OC pins. The OC1 output action taken when a match is found is controlled by two 8-bit registers with three bits unimplemented: the output compare 1 mask register, OC1M, and the output compare 1 data register, OC1D. OC1M specifies which port A outputs are to be used, and OC1D specifies what data is placed on these port pins.

9.3.1 Timer Output Compare Registers

All output compare registers are 16-bit read-write. Each is initialized to $FFFF at reset. If an output compare register is not used for an output compare function, it can be used as a storage location. A write to the high-order byte of an output compare register pair inhibits the output compare function for one bus cycle. This inhibition prevents inappropriate subsequent comparisons. Coherency requires a complete 16-bit read or write. However, if coherency is not needed, byte accesses can be used.

For output compare functions, write a comparison value to output compare registers TOC1–TOC4 and TI4/O5. When TCNT value matches the comparison value, specified pin actions occur.

TOC1–TOC4 — Timer Output Compare **$0016–$001D**

$0016	Bit 15	14	13	12	11	10	9	Bit 8	TOC1 (High)
$0017	Bit 7	6	5	4	3	2	1	Bit 0	TOC1 (Low)
$0018	Bit 15	14	13	12	11	10	9	Bit 8	TOC2 (High)
$0019	Bit 7	6	5	4	3	2	1	Bit 0	TOC2 (Low)
$001A	Bit 15	14	13	12	11	10	9	Bit 8	TOC3 (High)
$001B	Bit 7	6	5	4	3	2	1	Bit 0	TOC3 (Low)
$001C	Bit 15	14	13	12	11	10	9	Bit 8	TOC4 (High)
$001D	Bit 7	6	5	4	3	2	1	Bit 0	TOC4 (Low)

All TOCx register pairs reset to ones ($FFFF).

9.3.2 Timer Compare Force Register

The CFORC register allows forced early compares. FOC[1:5] correspond to the five output compares. These bits are 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 were a match between the OCx register and the free-running counter, except that the corresponding interrupt status flag bits are not set. The forced channels trigger their programmed pin actions to occur at the next timer count transition after the write to CFORC.

The CFORC bits should not be used on an output compare function that is programmed to toggle its output on a successful compare because a normal compare that occurs immediately before or after the force can result in an undesirable operation.

CFORC — Timer Compare Force **$000B**

	Bit 7	6	5	4	3	2	1	Bit 0
	FOC1	FOC2	FOC3	FOC4	FOC5	—	—	—
RESET:	0	0	0	0	0	0	0	0

FOC[1:5] — Force Output Comparison

When the FOC bit associated with an output compare circuit is set, the output compare circuit immediately performs the action it is programmed to do when an output match occurs.

0 = Not affected
1 = Output x action occurs

Bits [2:0] — Not implemented

Always read zero

9.3.3 Output Compare Mask Registers

Use OC1M with OC1 to specify the bits of port A that are affected by a successful OC1 compare. The bits of the OC1M register correspond to PA[7:3].

OC1M — Output Compare 1 Mask **$000C**

	Bit 7	6	5	4	3	2	1	Bit 0
	OC1M7	OC1M6	OC1M5	OC1M4	OC1M3	—	—	—
RESET:	0	0	0	0	0	0	0	0

OC1M[7:3] — Output Compare Masks

0 = OC1 is disabled.
1 = OC1 is enabled to control the corresponding pin of port A

Bits [2:0] — Not implemented

Always read zero

9.3.4 Output Compare Data Register

Use this register with OC1 to specify the data that is to be stored on the affected pin of port A after a successful OC1 compare. When a successful OC1 compare occurs, a data bit in OC1D is stored in the corresponding bit of port A for each bit that is set in OC1M.

OC1D — Output Compare 1 Data **$000D**

	Bit 7	6	5	4	3	2	1	Bit 0
	OC1D7	OC1D6	OC1D5	OC1D4	OC1D3	—	—	—
RESET:	0	0	0	0	0	0	0	0

If OC1Mx is set, data in OC1Dx is output to port A bit x on successful OC1 compares.

Bits [2:0] — Not implemented
Always read zero

9.3.5 Timer Counter Register

The 16-bit read-only TCNT register contains the prescaled value of the 16-bit timer. A full counter read addresses the most significant byte (MSB) first. A read of this address causes the least significant byte (LSB) to be latched into a buffer for the next CPU cycle so that a double-byte read returns the full 16-bit state of the counter at the time of the MSB read cycle.

TCNT — Timer Counter **$000E, $000F**

$000E	Bit 15	14	13	12	11	10	9	Bit 8	TCNT (High)
$000F	Bit 7	6	5	4	3	2	1	Bit 0	TCNT (Low)

TCNT resets to $0000. In normal modes, TCNT is a read-only register.

9.3.6 Timer Control Register 1

The bits of this register specify the action taken as a result of a successful OCx compare.

TCTL1 — Timer Control 1 **$0020**

	Bit 7	6	5	4	3	2	1	Bit 0
	OM2	OL2	OM3	OL3	OM4	OL4	OM5	OL5
RESET:	0	0	0	0	0	0	0	0

OM[2:5] — Output Mode

OL[2:5] — Output Level

These control bit pairs are encoded to specify the action taken after a successful OCx compare. OC5 functions only if the I4/O5 bit in the PACTL register is clear. Refer to Table 9–3 for the coding.

Table 9–3. Timer Output Compare Actions

OMx	OLx	Action Taken on Successful Compare
0	0	Timer disconnected from output pin logic
0	1	Toggle OCx output line
1	0	Clear OCx output line to 0
1	1	Set OCx output line to 1

9.3.7 Timer Interrupt Mask Register 1

Use this 8-bit register to enable or inhibit the timer input capture and output compare interrupts.

TMSK1 — Timer Interrupt Mask 1 **$0022**

	Bit 7	6	5	4	3	2	1	Bit 0
	OC1I	OC2I	OC3I	OC4I	I4/O5I	IC1I	IC2I	IC3I
RESET:	0	0	0	0	0	0	0	0

OC1I–OC4I — Output Compare x Interrupt Enable

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

I4/O5I — Input Capture 4/Output Compare 5 Interrupt Enable

When I4/O5 in PACTL is one, I4/O5I is the input capture 4 interrupt enable bit. When I4/O5 in PACTL is zero, I4/O5I is the output compare 5 interrupt enable bit.

IC1I–IC3I — Input Capture x Interrupt Enable

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

NOTE

Bits in TMSK1 correspond bit for bit with flag bits in TFLG1. Ones in TMSK1 enable the corresponding interrupt sources.

9.3.8 Timer Interrupt Flag Register 1

Bits in this register indicate when timer system events have occurred. Coupled with the bits of TMSK1, the bits of TFLG1 allow the timer subsystem to operate in either a polled or interrupt driven system. Each bit of TFLG1 corresponds to a bit in TMSK1 in the same position.

TFLG1 — Timer Interrupt Flag 1 **$0023**

	Bit 7	6	5	4	3	2	1	Bit 0
	OC1F	OC2F	OC3F	OC4F	I4/O5F	IC1F	IC2F	IC3F
RESET:	0	0	0	0	0	0	0	0

Clear flags by writing a one to the corresponding bit position(s).

OC1F–OC4F — Output Compare x Flag
Set each time the counter matches output compare x value

I4/O5F — Input Capture 4/Output Compare 5 Flag
Set by IC4 or OC5, depending on the function enabled by I4/O5 bit in PACTL

IC1F–IC3F — Input Capture x Flag
Set each time a selected active edge is detected on the ICx input line

9.3.9 Timer Interrupt Mask Register 2

Use this 8-bit register to enable or inhibit timer overflow and real-time interrupts. The timer prescaler control bits are included in this register.

TMSK2 — Timer Interrupt Mask 2 **$0024**

	Bit 7	6	5	4	3	2	1	Bit 0
	TOI	RTII	PAOVI	PAII	—	—	PR1	PR0
RESET:	0	0	0	0	0	0	0	0

TOI — Timer Overflow Interrupt Enable
0 = TOF interrupts disabled
1 = Interrupt requested when TOF is set to one

RTII — Real-Time Interrupt Enable
Refer to **9.4 Real-Time Interrupt**.

PAOVI — Pulse Accumulator Overflow Interrupt Enable
Refer to **9.6.3 Pulse Accumulator Status and Interrupt Bits**.

PAII — Pulse Accumulator Input Edge Interrupt Enable
Refer to **9.6.3 Pulse Accumulator Status and Interrupt Bits**.

Bits [3:2] — Not implemented
Always read zero

PR[1:0] — Timer Prescaler Select

These bits are used to select the prescaler divide-by ratio. In normal modes, PR[1:0] can only be written once, and the write must be within 64 cycles after reset. Refer to Tables 9–1 and 9–4 for specific timing values.

Table 9–4. Timer Prescale

PR[1:0]	Prescaler
0 0	1
0 1	4
1 0	8
1 1	16

NOTE

Bits in TMSK2 correspond bit for bit with flag bits in TFLG2. Ones in TMSK2 enable the corresponding interrupt sources.

9.3.10 Timer Interrupt Flag Register 2

Bits in this register indicate when certain timer system events have occurred. Coupled with the four high-order bits of TMSK2, the bits of TFLG2 allow the timer subsystem to operate in either a polled or interrupt driven system. Each bit of TFLG2 corresponds to a bit in TMSK2 in the same position.

TFLG2 — Timer Interrupt Flag 2 **$0025**

	Bit 7	6	5	4	3	2	1	Bit 0
	TOF	RTIF	PAOVF	PAIF	—	—	—	—
RESET:	0	0	0	0	0	0	0	0

Clear flags by writing a one to the corresponding bit position(s).

TOF — Timer Overflow Interrupt Flag
Set when TCNT changes from $FFFF to $0000

RTIF — Real-Time (Periodic) Interrupt Flag
Refer to **9.4 Real-Time Interrupt.**

PAOVF — Pulse Accumulator Overflow Interrupt Flag
Refer to **9.6 Pulse Accumulator.**

PAIF — Pulse Accumulator Input Edge Interrupt Flag
Refer to **9.6 Pulse Accumulator.**

Bits [3:0] — Not implemented
Always read zero

9.4 Real-Time Interrupt

The real-time interrupt (RTI) feature, used to generate hardware interrupts at a fixed periodic rate, is controlled and configured by two bits (RTR1 and RTR0) in the pulse accumulator control (PACTL) register. The RTII bit in the TMSK2 register enables the interrupt capability. The four different rates available are a product of the MCU oscillator frequency and the value of bits RTR[1:0]. Refer to Table 9–5, which shows the periodic real-time interrupt rates.

Table 9–5. RTI Rates

RTR[1:0]	E = 1 MHz	E = 2 MHz	E = 3 MHz	E = X MHz
0 0	2.731 ms	4.096 ms	8.192 ms	$(E/2^{13})$
0 1	5.461 ms	8.192 ms	16.384 ms	$(E/2^{14})$
1 0	10.923 ms	16.384 ms	32.768 ms	$(E/2^{15})$
1 1	21.845 ms	32.768 ms	65.536 ms	$(E/2^{16})$

The clock source for the RTI function is a free-running clock that cannot be stopped or interrupted except by reset. This clock causes the time between successive RTI timeouts to be a constant that is independent of the software latencies associated with flag clearing and service. For this reason, an RTI period starts from the previous timeout, not from when RTIF is cleared.

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 RTI period elapses before the RTIF flag is set for the first time. Refer to the TMSK2, TFLG2, and PACTL registers.

9.4.1 Timer Interrupt Mask Register 2

This register contains the real-time interrupt enable bits.

TMSK2 — Timer Interrupt Mask Register 2 **$0024**

	Bit 7	6	5	4	3	2	1	Bit 0
	TOI	RTII	PAOVI	PAII	—	—	PR1	PR0
RESET:	0	0	0	0	0	0	0	0

TOI — Timer Overflow Interrupt Enable
0 = TOF interrupts disabled
1 = Interrupt requested when TOF is set to one

RTII — Real-Time Interrupt Enable
0 = RTIF interrupts disabled
1 = Interrupt requested when RTIF set to one

PAOVI — Pulse Accumulator Overflow Interrupt Enable
Refer to **9.6 Pulse Accumulator**.

PAII — Pulse Accumulator Input Edge
Refer to **9.6 Pulse Accumulator**.

Bits [3:2] — Not implemented
Always read zero

PR[1:0] — Timer Prescaler Select
Refer to Table 9–4.

NOTE

Bits in TMSK2 correspond bit for bit with flag bits in TFLG2. Ones in TMSK2 enable the corresponding interrupt sources.

9.4.2 Timer Interrupt Flag Register 2

Bits of this register indicate the occurrence of timer system events. Coupled with the four high-order bits of TMSK2, the bits of TFLG2 allow the timer subsystem to operate in either a polled or interrupt driven system. Each bit of TFLG2 corresponds to a bit in TMSK2 in the same position.

TFLG2 — Timer Interrupt Flag 2 **$0025**

	Bit 7	6	5	4	3	2	1	Bit 0
	TOF	RTIF	PAOVF	PAIF	—	—	—	—
RESET:	0	0	0	0	0	0	0	0

Clear flags by writing a one to the corresponding bit position(s).

TOF — Timer Overflow Interrupt Flag
Set when TCNT changes from $FFFF to $0000. To clear TOF, write a byte to TFLG2 with bit 7 set.

RTIF — Real-Time Interrupt Flag
The RTIF status bit is automatically set to one at the end of every RTI period. To clear RTIF, write a byte to TFLG2 with bit 6 set.

PAOVF — Pulse Accumulator Overflow Interrupt Flag
Refer to **9.6 Pulse Accumulator**.

PAIF — Pulse Accumulator Input Edge Interrupt Flag
Refer to **9.6 Pulse Accumulator**.

Bits [3:0] — Not implemented
Always read zero

9.4.3 Pulse Accumulator Control Register

Bits RTR[1:0] of this register select the rate for the RTI system. The remaining bits control the pulse accumulator and IC4/OC5 functions.

PACTL — Pulse Accumulator Control **$0026**

	Bit 7	6	5	4	3	2	1	Bit 0
	—	PAEN	PAMOD	PEDGE	—	I4/O5	RTR1	RTR0
RESET:	0	0	0	0	0	0	0	0

Bits 7 and 3 — Not implemented
Always read zero

PAEN — Pulse Accumulator System Enable
Refer to **9.6 Pulse Accumulator.**

PAMOD — Pulse Accumulator Mode
Refer to **9.6 Pulse Accumulator.**

PEDGE — Pulse Accumulator Edge Control
Refer to **9.6 Pulse Accumulator.**

I4/O5 — Input Capture 4/Output Compare 5
Refer to **9.6 Pulse Accumulator.**

RTR[1:0] — RTI Interrupt Rate Select
These two bits determine the rate at which the RTI system requests interrupts. The RTI system is driven by an E divided by 2^{13} rate clock that is compensated so it is independent of the timer prescaler. These two control bits select an additional division factor. Refer to Table 9–5.

9.5 Computer Operating Properly Watchdog Function

The clocking chain for the COP function, tapped off of the main timer divider chain, is only superficially related to the main timer system. The CR[1:0] bits in the OPTION register and the NOCOP bit in the CONFIG register determine the status of the COP function. One additional register, COPRST, is used to arm and clear the COP watchdog reset system. Refer to **SECTION 5 RESETS AND INTERRUPTS** for a more detailed discussion of the COP function.

9.6 Pulse Accumulator

The M68HC11 N-series MCUs have an 8-bit counter that can be configured to operate either as a simple event counter, or for gated time accumulation, depending on the state of the PAMOD bit in the PACTL register. Refer to the pulse accumulator block diagram, Figure 9–3.

In the event counting mode, the 8-bit counter is incremented by pulses on an external pin (PAI). The maximum clocking rate for the external event counting mode is the E clock divided by two. In gated time accumulation mode, a free-running E-clock ÷ 64 signal drives the 8-bit counter, but only while the external PAI pin is activated. Refer to Table 9–6. The pulse accumulator counter can be read or written at any time.

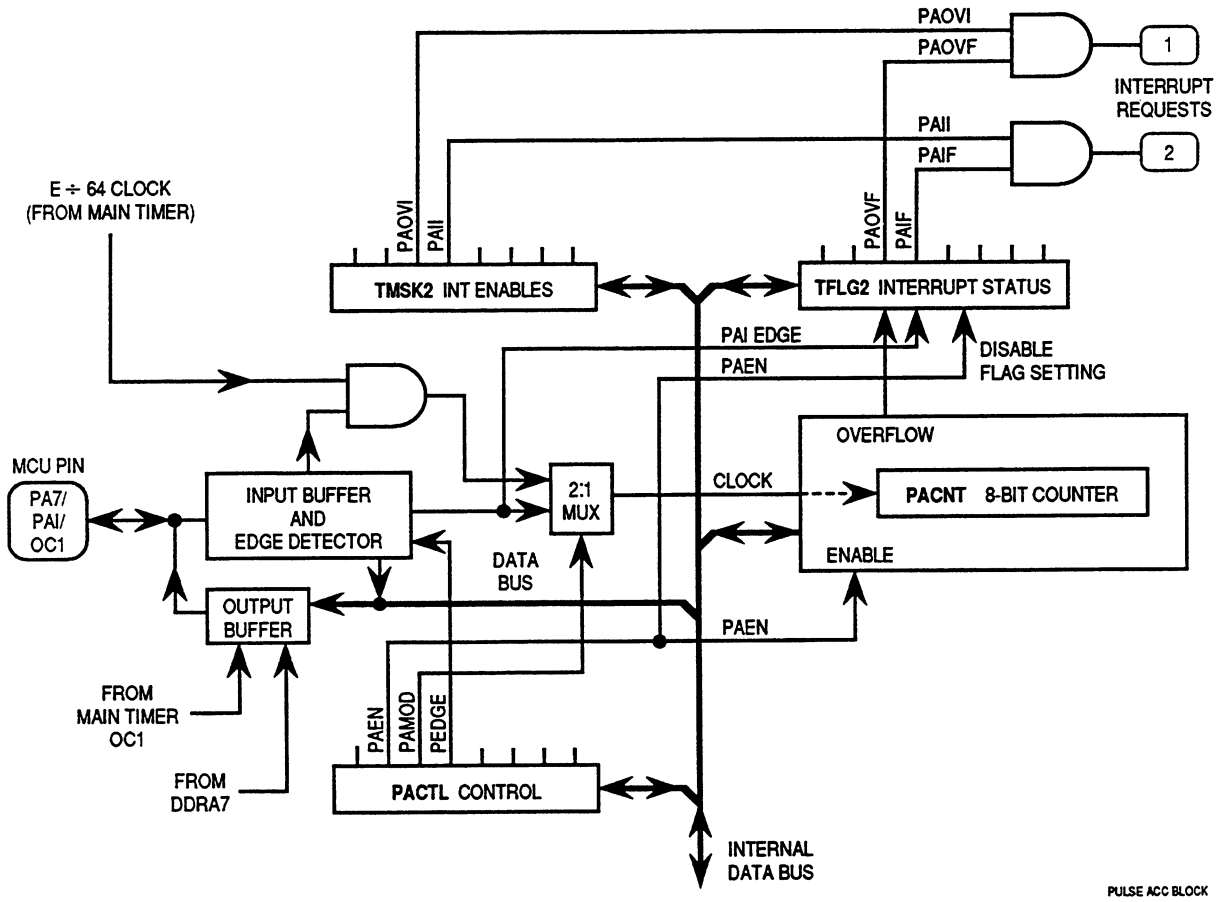


Figure 9–3. Pulse Accumulator

Table 9–6. Pulse Accumulator Timing

Crystal Frequency	E Clock	Cycle Time	E ÷ 64	PACNT Overflow
4.0 MHz	1.0 MHz	1000 ns	64 µs	16.384 ms
8.0 MHz	2.0 MHz	500 ns	32 µs	8.192 ms
12.0 MHz	3.0 MHz	333 ns	21.33 µs	5.461 ms
16.0 MHz	4.0 MHz	250 ns	16.0 µs	4.096 ms

Pulse accumulator control bits are also located within two timer registers, TMSK2 and TFLG2, as described in the following paragraphs.

9.6.1 Pulse Accumulator Control Register

Four of this register's bits control an 8-bit pulse accumulator system. Another bit enables either the OC5 function or the IC4 function, while two other bits select the rate for the real-time interrupt system.

PACTL — Pulse Accumulator Control **$0026**

	Bit 7	6	5	4	3	2	1	Bit 0
	—	PAEN	PAMOD	PEDGE	—	I4/O5	RTR1	RTR0
RESET:	0	0	0	0	0	0	0	0

Bits 7 and 3 — Not implemented
Always read zero

PAEN — Pulse Accumulator System Enable
0 = Pulse accumulator disabled
1 = Pulse accumulator enabled

PAMOD — Pulse Accumulator Mode
0 = Event counter
1 = Gated time accumulation

PEDGE — Pulse Accumulator Edge Control
This bit has different meanings depending on the state of the PAMOD bit, as shown in Table 9–7.

Table 9–7. Pulse Accumulator Edge Control

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.

I4/O5 — Input Capture 4/Output Compare 5
0 = Output compare 5 function enable (No IC4)
1 = Input capture 4 function enable (No OC5)

RTR[1:0] — RTI Interrupt Rate Selects
Refer to **9.4 Real-Time Interrupt.**

9.6.2 Pulse Accumulator Count Register

This 8-bit read/write register contains the count of external input events at the PAI input, or the accumulated count. The PACNT is readable even if PAI is not active in gated time accumulation mode. The counter is not affected by reset and can be read or written at any time. Counting is synchronized to the internal PH2 clock so that incrementing and reading occur during opposite half cycles.

PACNT — Pulse Accumulator Count **$0027**

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

9.6.3 Pulse Accumulator Status and Interrupt Bits

The pulse accumulator control bits, PAOVI and PAII, PAOVF and PAIF, are located within timer registers TMSK2 and TFLG2.

PAOVI and PAOVF — Pulse Accumulator Overflow Interrupt Enable and Flag
The PAOVF status bit is set each time the pulse accumulator count rolls over from $FF to $00. To clear this status bit, write a one in the corresponding data bit position (bit 5) of the TFLG2 register. The PAOVI control bit allows configuring the pulse accumulator overflow for polled or interrupt-driven operation and does not affect the state of PAOVF. When PAOVI is zero, pulse accumulator overflow interrupts are inhibited, and the system operates in a polled mode, which requires that PAOVF be polled by user software to determine when an overflow has occurred. When the PAOVI control bit is set, a hardware interrupt request is generated each time PAOVF is set. Before leaving the interrupt service routine, software must clear PAOVF by writing to the TFLG2 register.

PAII and PAIF — Pulse Accumulator Input Edge Interrupt Enable and Flag
The PAIF status bit is automatically set each time a selected edge is detected at the PA7/PAI/OC1 pin. To clear this status bit, write to the TFLG2 register with a one in the corresponding data bit position (bit 4). The PAII control bit allows configuring the pulse accumulator input edge detect for polled or interrupt-driven operation but does not affect setting or clearing the PAIF bit. When PAII is zero, pulse accumulator input interrupts are inhibited, and the system operates in a polled mode. In this mode, the PAIF bit must be polled by user software to determine when an edge has occurred. When the PAII control bit is set, a hardware interrupt

request is generated each time PAIF is set. Before leaving the interrupt service routine, software must clear PAIF by writing to the TFLG register.

TMSK2 — Timer Interrupt Mask 2 Register **$0024**

	Bit 7	6	5	4	3	2	1	Bit 0
	TOI	RTII	PAOVI	PAII	—	—	PR1	PR0
RESET:	0	0	0	0	0	0	0	0

TFLG2 — Timer Interrupt Flag 2 Register **$0025**

	Bit 7	6	5	4	3	2	1	Bit 0
	TOF	RTIF	PAOVF	PAIF	—	—	—	—
RESET:	0	0	0	0	0	0	0	0

9.7 Pulse-Width Modulation Timer

The M68HC11 N-series MCUs contain a six-channel PWM timer that is composed of a two-channel 12-bit modulator and a four-channel 8-bit modulator. Each of the modulators can create independent continuous waveforms with software-selectable duty rates from 0% to 100%.

Each of the four 8-bit channels has a counter, a period register, and a duty register. For 8-bit channels, the waveform output is the result of a match between the period register and the value in the counter. The duty register changes the state of the output during the period to determine the duty.

The two 12-bit channels have only a counter and a duty register. The period of the 12-bit channels is determined by bits in the PWSIZ register. The period is dependent upon whether the user configures a 12-bit channel for 8-, 10-, or 12-bit counts. The waveform output is the result of a match between the duty register and the value in the counter.

The following diagram shows the clock system used by both the two-channel and four-channel modulators.

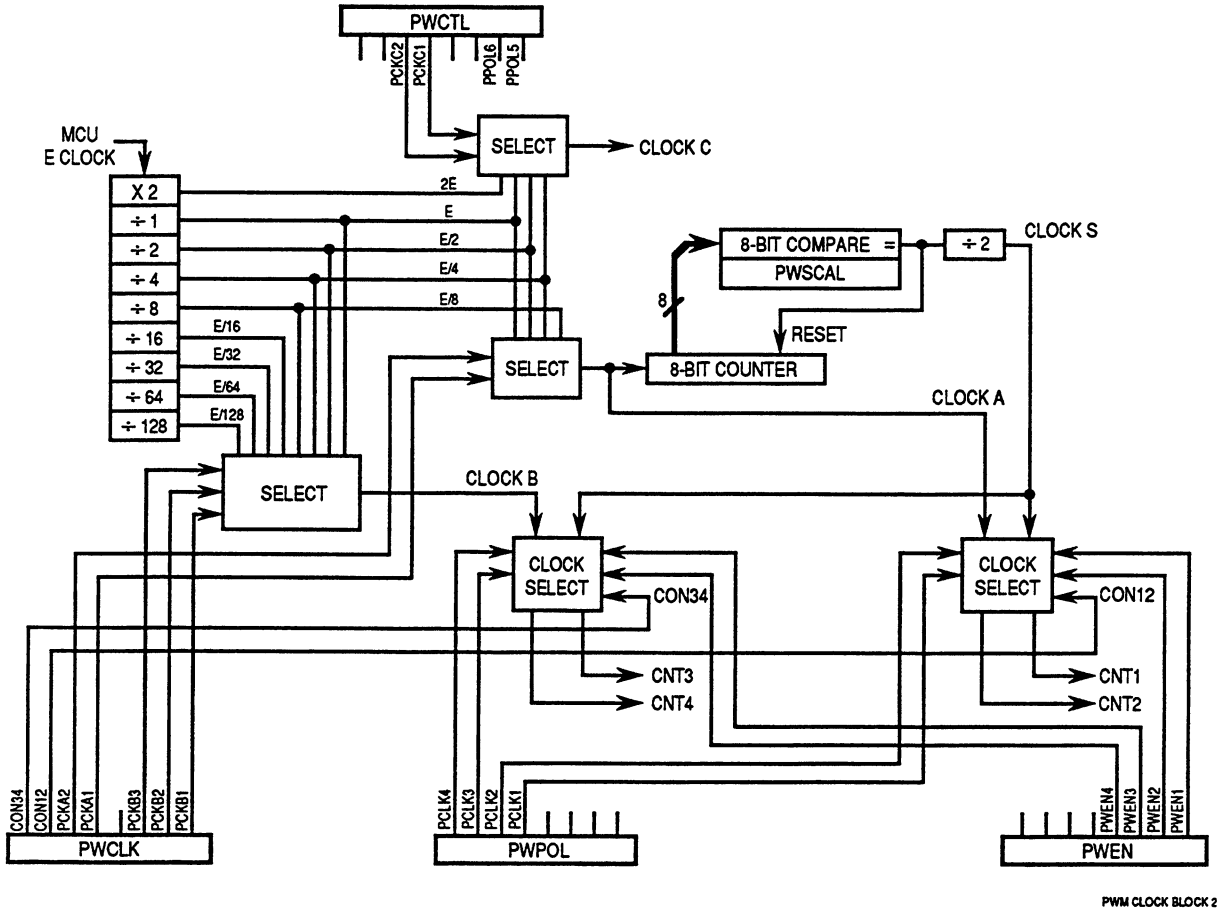


Figure 9–4. Pulse-Width Modulation System Clock Source

9.7.1 Four-Channel Modulator

The four-channel modulator provides up to four pulse-width modulated waveforms. Each channel has its own counter. Pairs of counters can be concatenated to create 16-bit PWM outputs based on 16-bit counts. Three clock sources (A, B, and S) give the PWM a wide range of frequencies.

Four control registers configure the 8-bit PWM outputs — PWCLK, PWPOL, PWSCAL, and PWEN. The PWCLK register selects the prescale value for PWM clock sources and enables the 16-bit counters. The PWPOL register determines each channel's polarity and selects the clock source for each channel. The PWSCAL register derives a user-scaled clock, based on the A clock source, and the PWEN register enables the PWM channels.

Each channel has a separate 8-bit counter, period register, and duty cycle register. The period and duty cycle registers are double buffered so that if they are changed while the channel is enabled, the change does not take effect until the counter rolls over or the channel is disabled.

With channels configured for 8-bit mode and E = 4 MHz, PWM signals of 40 kHz (1% duty cycle resolution) to less than 10 Hz (approximately 0.4% duty cycle resolution) can be produced. By configuring the channels for 16-bit mode with E = 4 MHz, PWM periods greater than one minute are possible.

In 16-bit mode, duty cycle resolution of almost 15 parts per million can be achieved (at a PWM frequency of about 60 Hz). In the same system, a PWM frequency of 1 kHz corresponds to a duty cycle resolution of 0.025%. The following block diagram shows the four 8-bit channels of the M68HC11 N-series PWM system.

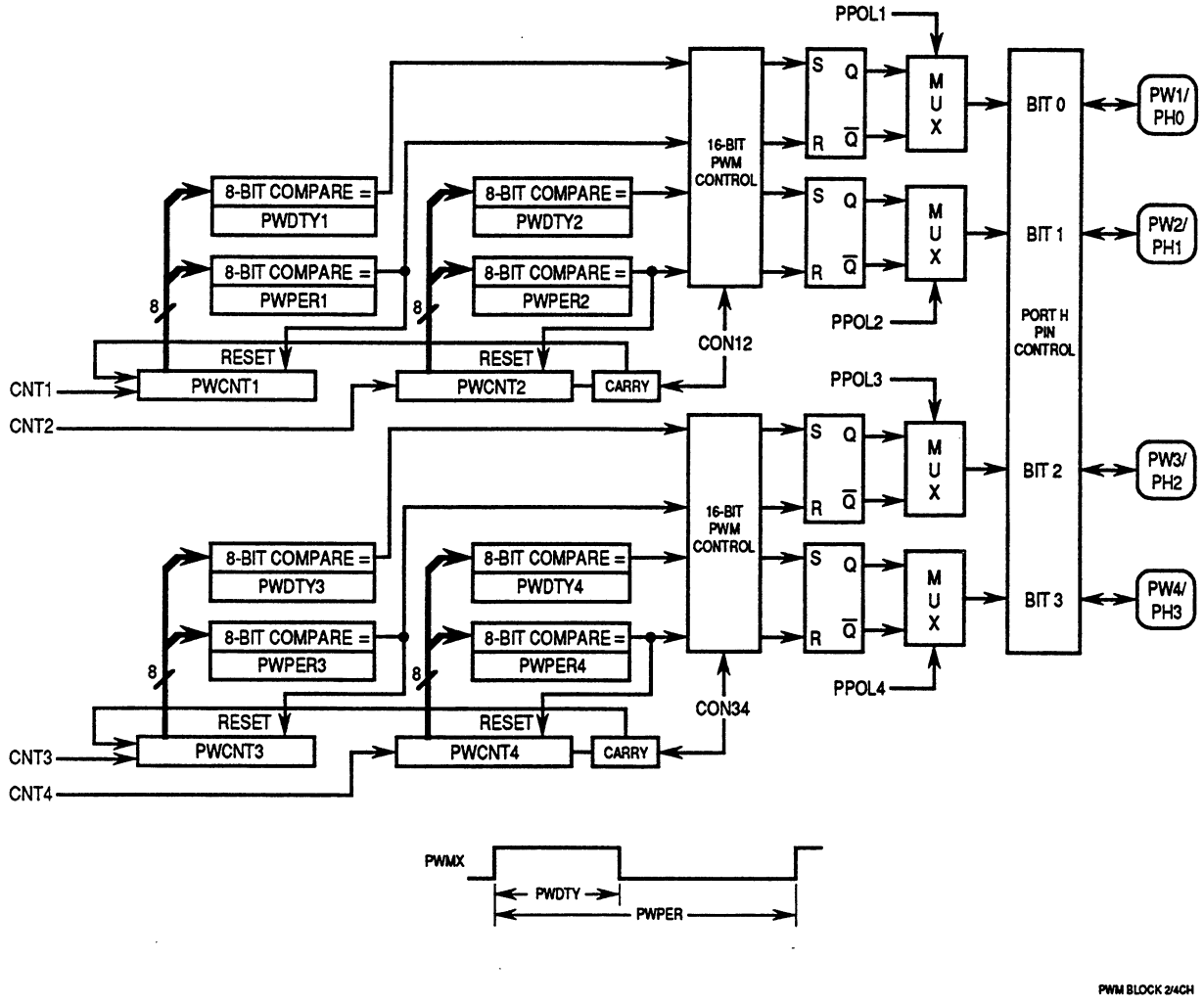


Figure 9–5. Four 8-Bit Channels of the PWM System

9.7.2 Two-Channel Modulator

PWM channels 5 and 6 share a 12-bit counter. The period of each PWM signal begins when the counter rolls over. Each channel contains a duty register that causes the output to change state when the content of the duty register matches the value of the counter. A control bit for each channel allows polarity selection. Also, the period for each channel can be configured for 8-, 10-, or 12-bit counts.

Each duty register is double buffered. When a channel is active, writes to the duty register are buffered until the counter rolls over or the channel is disabled. At this time the new value takes effect. This design requires the output to be either the old duty waveform or the new duty waveform. If the channel is not enabled, writes to the duty register go directly to both the latches and the buffer. Refer to Figure 9–6.

Two control registers, PWSIZ and PWCTL, configure the outputs of channels 5 and 6. When the enable bit is set to one, the pulse modulated signal is available at the associated port H line. Channel 5 is port H bit 4. Channel 6 is port H bit 5. The following block diagram shows the two 12-bit channels of the M68HC11 N-series PWM system.

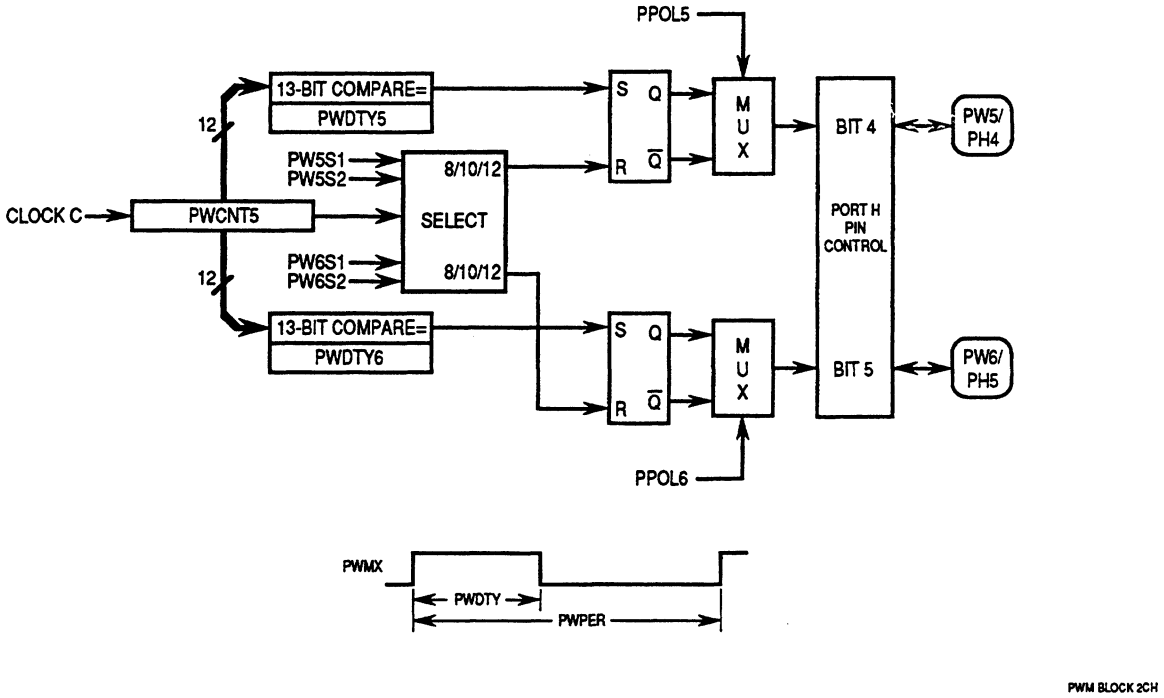


Figure 9–6. Two 12-Bit Channels of the PWM System

9.7.3 PWM Clock Select

The three clocks available to the four 8-bit channels are clock A, clock B, and clock S (scaled). Clock A can be software selected to be E, E/2, E/4, or E/8. Clock B can be software selected to be E, E/2, E/4, ..., E/128. The scaled clock (clock S) uses clock A as an input and divides it with a reloadable counter. The rates available are software selectable to be clock A divided by two down to clock A divided by 512. Each 8-bit PWM timer channel has the capability of selecting either one of two clocks. PWM channels 1 and 2 can select either clock A or clock S. Channels 3 and 4 can select either clock B or clock S. PWM channels 5 and 6 must use clock C. The PWM clock source block diagram shows the clock sources and how the scaled clock is generated. Refer to Figure 9–4.

Clock A is an input to an 8-bit counter which is then compared to a user programmable scale value. When they match, this circuit has an output that is divided by two and the counter is reset.

The PWCLK, PWSCAL, and PWPOL registers are associated with the clock select circuit for the four 8-bit channels. The PWCTL register is associated with the clock select circuit for the two 12-bit channels.

PWCLK — Pulse-Width Modulation Timer Clock Control **$0060**

	Bit 7	6	5	4	3	2	1	Bit 0
	CON34	CON12	PCKA2	PCKA1	—	PCKB3	PCKB2	PCKB1
RESET:	0	0	0	0	0	0	0	0

CON34 — Concatenate Channels 3 and 4
Channel 3 is high-order byte, and channel 4 (port H, bit 3) is output.
0 = Channels 3 and 4 are separate 8-bit PWMs.
1 = Channels 3 and 4 are concatenated to create one 16-bit PWM channel.

CON12 — Concatenate Channels 1 and 2
Channel 1 is high-order byte, and channel 2 (port H, bit 1) is output.
0 = Channels 1 and 2 are separate 8-bit PWMs
1 = Channels 1 and 2 are concatenated to create one 16-bit PWM channel.

PCKA[2:1] — Prescaler for Clock A (See also PWSCAL register)
Determines the rate of clock A. Refer to Table 9–8.

Table 9–8. Clock A Prescaler

PCKA[2:1]	Value of Clock A
0 0	E
0 1	E/2
1 0	E/4
1 1	E/8

Bit 3 — Not implemented
Always reads zero

PCKB[3:1] — Prescaler for Clock B
Determines the rate for clock B. Refer to Table 9–9.

Table 9–9. Clock B Prescaler

PCKB[3:1]	Value of Clock B
000	E
001	E/2
010	E/4
011	E/8
100	E/16
101	E/32
110	E/64
111	E/128

PWSCAL — PWM Timer Prescaler for Channels 1–4 **$0062**

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

Scaled clock S is generated by dividing clock A by twice the value in PWSCAL. When PWSCAL = $00, clock A is divided by 256 then by two to generate clock S.

PWPOL — PWM Timer Polarity and Clock Select for Channels 1–4 **$0061**

	Bit 7	6	5	4	3	2	1	Bit 0
	PCLK4	PCLK3	PCLK2	PCLK1	PPOL4	PPOL3	PPOL2	PPOL1
RESET:	0	0	0	0	0	0	0	0

Note that while PWPOL can be written at any time, if a clock select is changed during a PWM output, a stretched or truncated signal may be generated.

PCLK4 — PWM Channel 4 Clock Select
0 = Clock B is source
1 = Clock S is source

PCLK3 — PWM Channel 3 Clock Select
0 = Clock B is source
1 = Clock S is source

PCLK2 — PWM Channel 2 Clock Select
0 = Clock A is source
1 = Clock S is source

PCLK1 — PWM Channel 1 Clock Select
0 = Clock A is source
1 = Clock S is source

PPOL[4:1] — PWM Channel x Polarity
Refer to **9.7.9 Polarity Select.**

PWCTL — PWM Timer Polarity and Clock Control for Channels 5 and 6 **$0050**

	Bit 7	6	5	4	3	2	1	Bit 0
	—	—	PCKC2	PCKC1	—	—	PPOL6	PPOL5
RESET:	0	0	0	0	0	0	0	0

Bits [7:6] — Not implemented
Always read zero

PCKC[2:1] — Prescaler for Clock C
Determine rate of clock C

Table 9–10. Clock C Prescaler

PCKC[2:1]	Value of Clock C
0 0	2E
0 1	E
1 0	E/2
1 1	E/4

Bits [3:2] — Not implemented
Always read zero

PPOL[6:5] — Pulse-Width Polarity for Channels 5 and 6
Refer to **9.7.9 Polarity Select.**

9.7.4 Counters

Each 8-bit channel has its own counter. The two 12-bit channels share a single counter. All of these counters can be read at any time without affecting the count or the operation of the associated PWM channel. A write to a counter, which can be done before the counter is enabled, causes the counter to be reset to $00. The counter for channels 5 and 6 resets to $0000.

PWCNT1–4 — PWM Channel 1 to 4 Counter **$0064–$0067**

$0064	Bit 7	6	5	4	3	2	1	Bit 0	PWCNT1
$0065	Bit 7	6	5	4	3	2	1	Bit 0	PWCNT2
$0066	Bit 7	6	5	4	3	2	1	Bit 0	PWCNT3
$0067	Bit 7	6	5	4	3	2	1	Bit 0	PWCNT4
RESET:	0	0	0	0	0	0	0	0	

Resets to $00.

PWCNT5 — PWM Channel 5 and 6 Counter **$0056, $0057**

$0056	—	—	—	—	Bit 11	10	9	Bit 8	PWCNT5 (High)
$0057	Bit 7	6	5	4	3	2	1	Bit 0	PWCNT5 (Low)
RESET:	0	0	0	0	0	0	0	0	

Resets to $0000.

9.7.5 Period Registers

There is one period register for each of the four 8-bit channels. The value in this register determines the period of the associated PWM timer channel. If a period register is written while its channel is enabled, the new value will not take effect until the existing period value causes the counter to reset. The new period is then latched and used for the next period. A new period can be started immediately by writing the new period value to PWPERx and then writing $00 to the counter (PWCNTx). Reads of this register return the most recent value written.

PWPER1–4 — Pulse-Width Modulation Timer Period 1 to 4 **$0068–$006B**

$0068	Bit 7	6	5	4	3	2	1	Bit 0	PWPER1
$0069	Bit 7	6	5	4	3	2	1	Bit 0	PWPER2
$006A	Bit 7	6	5	4	3	2	1	Bit 0	PWPER3
$006B	Bit 7	6	5	4	3	2	1	Bit 0	PWPER4
RESET:	1	1	1	1	1	1	1	1	

PWPER1–4

Determines period of associated PWM channel.

9.7.6 Period Size and Enable (Channels 5 and 6)

For channels 5 and 6, both period size and channel enable are controlled by the PWSIZ register. The period for channels 5 and 6 is determined by the value of the size bits for the associated channel. When the value of the size bits is **not** equal to zero, a PWM size is selected and the associated channel is enabled. This forces the associated port H pin to be an output regardless of the state of the DDRH bit.

PWSIZ — PWM Size Select for Channels 5 and 6 **$0051**

	Bit 7	6	5	4	3	2	1	Bit 0
	—	—	—	—	PW6S2	PW6S1	PW5S2	PW5S1
RESET:	0	0	0	0	0	0	0	0

Bits [7:4] — Not implemented
Always read zero

PW6S[2:1] — PWM Channel 6 Size Select
Refer to Table 9–11.

PW5S[2:1] — PWM Channel 5 Size Select
Refer to Table 9–11.

Table 9–11. PWM Size Select for Channels 5 and 6

PWxS[2:1]	State	Period Size	Counts
0 0	Disabled	—	—
0 1	Enabled	8 bits	256
1 0	Enabled	10 bits	1024
1 1	Enabled	12 bits	4096

9.7.7 Duty Registers

There is one 8-bit duty register for each of the four 8-bit channels. There is one 13-bit duty register for each of the two 12-bit channels. The value in these registers determines the duty of the associated PWM timer channel. The duty value is compared to the counter and if it is equal, a match occurs and the output goes to the state defined by the associated polarity bit. If the register is written while the channel is enabled, then the new value is held in a buffer until the counter rolls over or the channel is disabled. Reads of this register return the most recent value written. Note that the duty registers for the two 12-bit channels use 13 bits. This allows duty cycle resolutions of 100% to be specified.

PWDTY1–4 — Pulse-Width Modulation Timer Duty Cycle 1 to 4 **$006C–$006F**

	Bit 7	6	5	4	3	2	1	Bit 0	
$006C	Bit 7	6	5	4	3	2	1	Bit 0	PWDTY1
$006D	Bit 7	6	5	4	3	2	1	Bit 0	PWDTY2
$006E	Bit 7	6	5	4	3	2	1	Bit 0	PWDTY3
$006F	Bit 7	6	5	4	3	2	1	Bit 0	PWDTY4
RESET:	1	1	1	1	1	1	1	1	

PWDTY1–4
Determines duty cycle of associated PWM channel.

PWTDY5 — PWM Channel 5 Duty **$0052, $0053**

$0052	—	—	—	Bit 12	11	10	9	Bit 8	PWTDY5 (High)
$0053	Bit 7	6	5	4	3	2	1	Bit 0	PWTDY5 (Low)

Resets to $1FFF.

PWDTY5 (High/Low)
Determines duty cycle of PWM channel 5.

PWTDY6 — PWM Channel 6 Duty **$0054, $0055**

$0054	—	—	—	Bit 12	11	10	9	Bit 8	PWTDY6 (High)
$0055	Bit 7	6	5	4	3	2	1	Bit 0	PWTDY6 (Low)

Resets to $1FFF.

PWDTY6 (High/Low)
Determines duty cycle of PWM channel 6.

9.7.8 PWM Control Registers

There are several control bits associated with the PWM timer subsystem. Each 8-bit timer has an enable bit to start its waveform output. Writing any of these PWENx bits to one causes the associated port H line to become an output regardless of the state of the associated DDR bit. This does not change the state of the DDR bit and when PWENx returns to zero the DDR bit again controls I/O state. On the front end of the PWM timer the clock to the PWM circuit is enabled by the PWENx bit being high. There is a synchronizing circuit to guarantee that the clock will only be enabled or disabled at an edge.

PWEN contains 4 PWM enable bits — one for each of the four 8-bit channels. When an enable bit is set to one, the pulse modulated signal becomes available at the associated port H pin only after its clock source starts a new cycle.

PWEN — Pulse-Width Modulation Timer Enable **$0063**

	Bit 7	6	5	4	3	2	1	Bit 0
	TPWSL	DISCP	—	—	PWEN4	PWEN3	PWEN2	PWEN1
RESET:	0	0	0	0	0	0	0	0

TPWSL — PWM Scaled Clock Test Bit (Test)
When TPWSL is one, clock S from the PWM timer is output to PWSCAL register. Writing to the PWSCAL register still functions normally.
0 = Normal operation
1 = Clock S output to PWSCAL register (Test only)

DISCP — Disable Compare Scaled E Clock (Test)
0 = Normal operation
1 = Match of period does not reset associated count register (Test only)

Bits [5:4] — Not implemented
Always read zero

PWEN[4:1] — PWMx Enable
0 = Channel disabled
1 = Channel enabled

9.7.9 Polarity Select

Each channel has a polarity bit that allows starting a cycle with either a high or low signal. This is shown in the block diagrams, Figures 9–5 and 9–6, as a mux select of either Q output or the $\overline{Q}$ output of the PWM output flip-flop. When a PPOLx bit is set, the associated PWM channel output is high at the beginning of the clock cycle, then goes low when the duty count is reached. Note that PWPOL register contains the polarity bits for the four 8-bit channels and PWCTL register contains the polarity bits for the two 12-bit channels.

PWPOL — PWM Timer Polarity and Clock Select for Channels 1–4 **$0061**

	Bit 7	6	5	4	3	2	1	Bit 0
	PCLK4	PCLK3	PCLK2	PCLK1	PPOL4	PPOL3	PPOL2	PPOL1
RESET:	0	0	0	0	0	0	0	0

PCLK[4:1] — PWM Channel 4–1 Clock Select
Refer to **9.7.3 PWM Clock Select**.

PPOL[4:1] — PWM Channel x Polarity
0 = PWM channel x output is low at the beginning of the clock cycle and goes high when duty count is reached
1 = PWM channel x output is high at the beginning of the clock cycle and goes low when duty count is reached

PWCTL — PWM Timer Polarity and Clock Control for Channels 5 and 6 **$0050**

	Bit 7	6	5	4	3	2	1	Bit 0
	—	—	PCKC2	PCKC1	—	—	PPOL6	PPOL5
RESET:	0	0	0	0	0	0	0	0

Bits [7:6] — Not implemented
Always read zero

PCKC[2:1] — Prescaler for Clock C
Refer to **9.7.3 PWM Clock Select**.

Bits [3:2] — Not implemented
Always read zero

PPOL6 — PWM Channel 6 Polarity
0 = PWM channel 6 output is low at start of clock cycle, then goes high when duty count is reached
1 = PWM channel 6 output is high at start of clock cycle, then goes low when duty count is reached

PPOL5 — PWM Channel 5 Polarity
0 = PWM channel 5 output is low at start of clock cycle, then goes high when duty count is reached
1 = PWM channel 5 output is high at start of clock cycle, then goes low when duty count is reached

9.7.10 16-Bit PWM Function

The PWCLK register contains two control bits, each of which is used to concatenate a pair of PWM channels into one 16-bit channel. Channels 3 and 4 are concatenated with the CON34 bit, and channels 1 and 2 are concatenated with the CON12 bit.

When the 16-bit concatenated mode is selected, the clock source is determined by the low order channel. Channel 2 is the low order channel when channels 1 and 2 are concatenated. Channel 4 is the low order channel when channels 3 and 4 are concatenated. Port H pins that correspond to channels 1 and 3 can be used for general-purpose I/O when 16-bit PWM mode is selected.

Channel 1 registers are the high order byte of the double-byte channel when channels 1 and 2 are concatenated. Channel 3 registers are the high order byte of the double-byte channel when channels 3 and 4 are concatenated. Reads of the high order byte cause the low order byte to be latched for one cycle to guarantee that double byte reads are accurate. Writes to the lower byte of the counter cause reset of the entire counter. Writes to the upper bytes of the counter have no effect.

PWCLK — Pulse-Width Modulation Timer Clock Select **$0060**

	CON34	CON12	PCKA2	PCKA1	—	PCKB3	PCKB2	PCKB1
RESET:	0	0	0	0	0	0	0	0

CON34 — Concatenate Channels 3 and 4
Channel 3 is high-order byte, and channel 4 (port H, bit 3) is output.
0 = Channels 3 and 4 are separate 8-bit PWM channels
1 = Channels 3 and 4 are concatenated to create one 16-bit PWM channel

CON12 — Concatenate Channels 1 and 2
Channel 1 is high-order byte, and channel 2 (port H, bit 1) is output.
0 = Channels 1 and 2 are separate 8-bit PWMs
1 = Channels 1 and 2 are concatenated to create one 16-bit PWM channel.

PCKA[2:1] — Prescaler for Clock A
Refer to Table 9–9.

Bit 3 — Not implemented
Always reads zero

PCKB[3:1] — Prescaler for Clock B
Refer to Table 9–10.

9.7.11 PWM Boundary Cases

Certain values written to PWM control registers, counters, etc. can cause outputs that are not what the user might expect. These are referred to as boundary cases. Boundary cases occur when the user specifies a value that is either a maximum or a minimum. This value combined with other conditions causes unexpected behavior of the PWM system.

9.7.11.1 Boundary Cases for Channels 1–4

The following conditions cause the corresponding output to always be high:

PWDTYx = $00, PWPERx > $00, and PPOLx = 0
PWDTYx ≥ PWPERx, and PPOLx = 1
PWPERx = $00 and PPOLx = 1

The following conditions cause the corresponding output to always be low:

PWDTYx = $00, PWPERx > $00, and PPOLx = 1
PWDTYx ≥ PWPERx, and PPOLx = 0
PWPERx = $00 and PPOLx = 0

9.7.11.2 Boundary Cases for Channels 5 and 6

The following conditions cause the corresponding output to always be high:

PWDTYx = $0000 and PPOLx = 0
PWxS[2:1] = 0:1, PWDTYx ≥ $0100 and PPOLx = 1
PWxS[2:1] = 1:0, PWDTYx ≥ $0400, and PPOLx = 1
PWxS[2:1] = 1:1, PWDTYx ≥ $1000, and PPOLx = 1

The following conditions cause the corresponding output to always be low:

PWDTYx = $0000 and PPOLx = 1
PWxS[2:1] = 0:1, PWDTYx ≥ $0100 and PPOLx = 0
PWxS[2:1] = 1:0, PWDTYx ≥ $0400 and PPOLx = 0
PWxS[2:1] = 1:1, PWDTYx ≥ $1000 and PPOLx = 0

SECTION 10
ANALOG-TO-DIGITAL CONVERTER

The analog-to-digital (A/D) system, a successive approximation converter, uses an all-capacitive charge redistribution technique to convert analog signals to digital values.

10.1 Overview

The A/D system is a 12-channel, 8-bit, multiplexed-input converter. The AV_{DD} pin is used to input supply voltage to the A/D converter. This allows the supply voltage to be bypassed independently. The converter does not require external sample and hold circuits because of the type of charge redistribution technique used. A/D converter timing can be synchronized to the system E clock, or to an internal resistor capacitor (RC) oscillator. The A/D converter system consists of four functional blocks: multiplexer, analog converter, digital control, and result storage. Refer to Figure 10–1.

10.1.1 Multiplexer

The multiplexer selects one of 16 inputs for conversion. Input selection is controlled by the value of bits CD–CA in the ADCTL register. The eight port E pins and four of the port G pins are fixed-direction analog inputs to the multiplexer, and internal analog signal lines are routed to it.

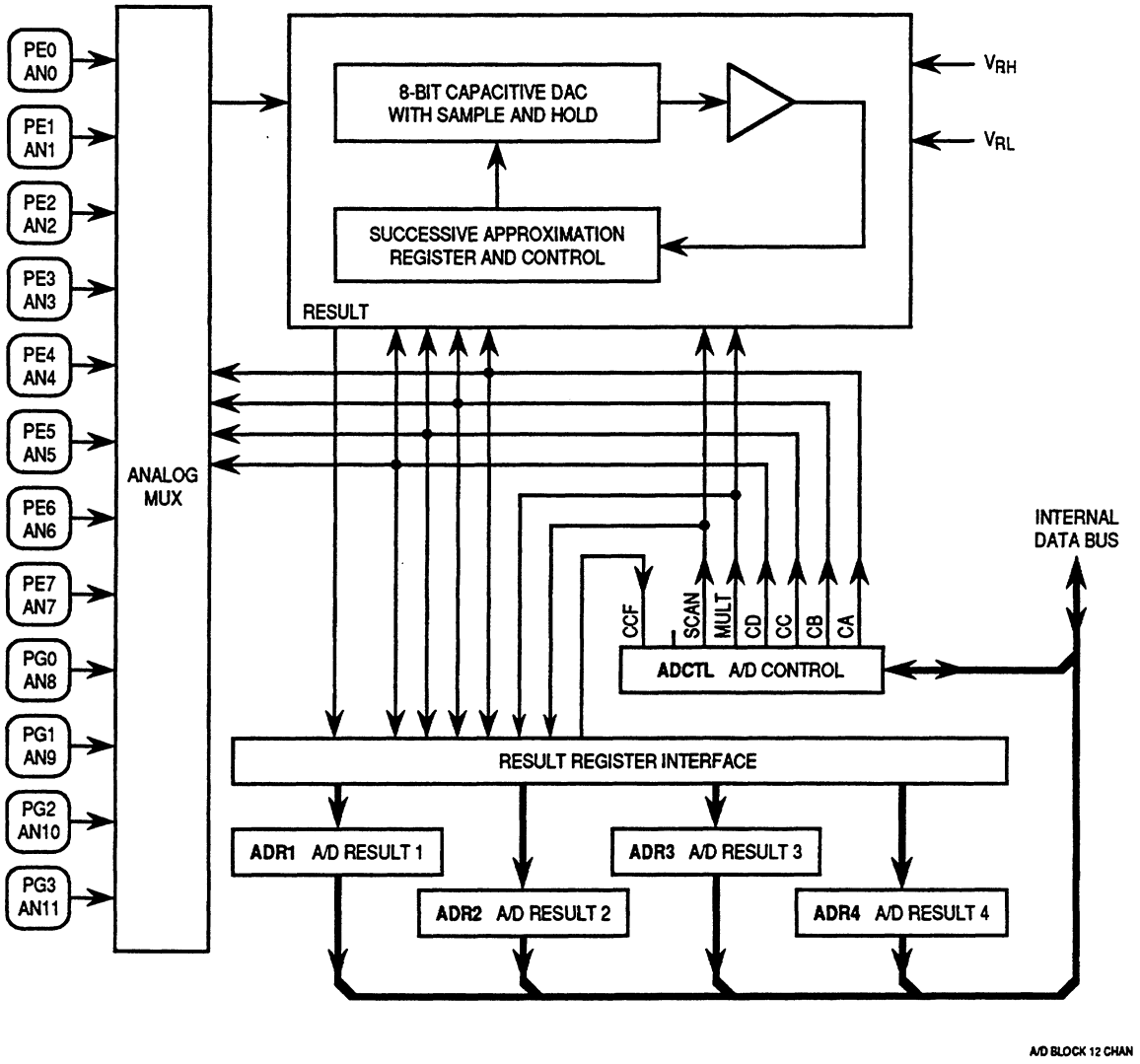
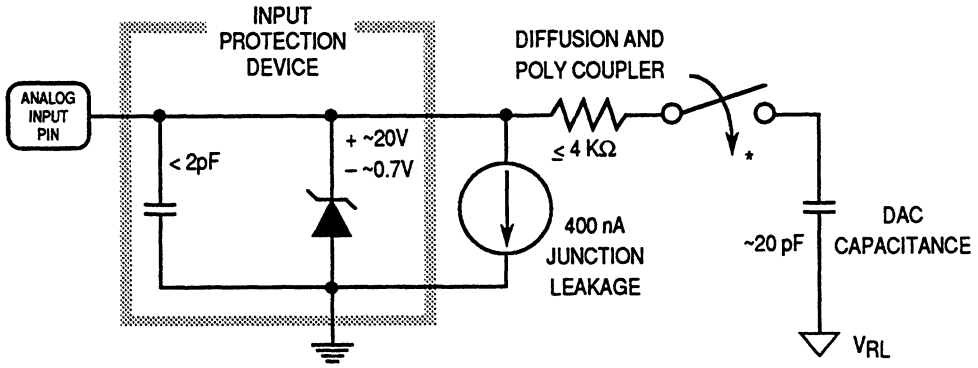


Figure 10–1. A/D Converter Block Diagram

Port E pins can also be used as digital inputs. Digital reads of port E pins are not recommended during the sample portion of an A/D conversion cycle, when the gate signal to the N-channel input gate is on. Because no P-channel devices are directly connected to either input pins or reference voltage pins, voltages above V_{DD} do not cause a latchup problem, although current should be limited according to maximum ratings. Refer to Figure 10–2, which is a functional diagram of an input pin.



*THIS ANALOG SWITCH IS CLOSED ONLY DURING THE 12-CYCLE SAMPLE TIME.

ANALOG INPUT PIN

Figure 10–2. Electrical Model of an A/D Input Pin (Sample Mode)

10.1.2 Analog Converter

Conversion of an analog input selected by the multiplexer occurs in this block. It contains a digital-to-analog capacitor (DAC) array, a comparator, and a successive approximation register (SAR). Each conversion is a sequence of eight comparison operations, beginning with the most significant bit (MSB). Each comparison determines the value of a bit in the successive approximation register.

The DAC array performs two functions. It acts as a sample and hold circuit during the entire conversion sequence, and provides comparison voltage to the comparator during each successive comparison.

The result of each successive comparison is stored in the SAR. When a conversion sequence is complete, the contents of the SAR are transferred to the appropriate result register.

A charge pump provides switching voltage to the gates of analog switches in the multiplexer. Charge pump output must stabilize between 7 and 8 volts within up to 100 µs before the converter can be used. The charge pump is enabled by the ADPU bit in the OPTION register.

Power is provided to the A/D converter system through the AV_{DD} and AV_{SS} pins.

10.1.3 Digital Control

All A/D converter operations are controlled by bits in register ADCTL. In addition to selecting the analog input to be converted, ADCTL bits indicate conversion status, and control whether single or continuous conversions are performed. Finally, the ADCTL bits determine whether conversions are performed on single or multiple channels.

10.1.4 Result Registers

Four 8-bit registers (ADR1 – ADR4) store conversion results. Each of these registers can be accessed by the processor in the CPU. The conversion complete flag (CCF) indicates when valid data is present in the result registers. The result registers are written during a portion of the system clock cycle when reads do not occur, so there is no conflict.

10.1.5 A/D Converter Clocks

The CSEL bit in the OPTION register selects whether the A/D converter uses the system E clock or an internal RC oscillator for synchronization. When E-clock frequency is below 750 kHz, charge leakage in the capacitor array can cause errors, and the internal oscillator should be used. When the RC clock is used, additional errors can occur because the comparator is sensitive to the additional system clock noise.

10.1.6 Conversion Sequence

A/D converter operations are performed in sequences of four conversions each. A conversion sequence can repeat continuously or stop after one iteration. The conversion complete flag (CCF) is set after the fourth conversion in a sequence to show the availability of data in the result registers. Figure 10–3 shows the timing of a typical sequence. Synchronization is referenced to the system E clock.

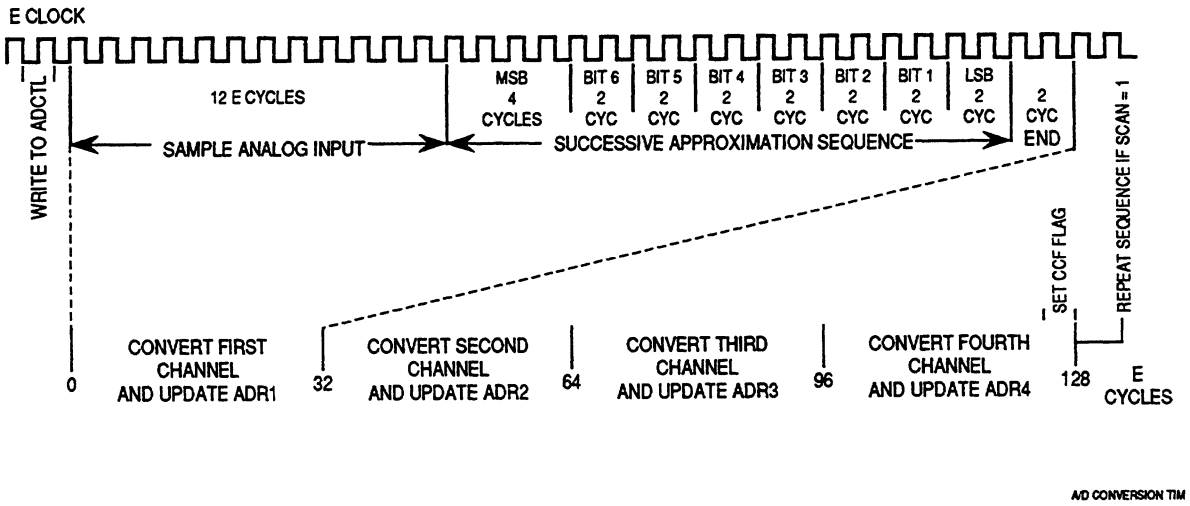


Figure 10–3. A/D Conversion Sequence

10.2 A/D Converter Power-Up Operation and Clock Select

Bit 7 of the OPTION register controls A/D converter power-up operation. Clearing ADPU removes power from and disables the A/D converter system. Setting ADPU enables the A/D converter system. Stabilization of the analog bias voltages requires a delay of as much as 100 µs after turning on the A/D converter. When the A/D converter 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 times, which minimizes noise errors. The internal RC oscillator is asynchronous to the MCU clock, so noise affects A/D converter results, which lowers accuracy slightly while CSEL = 1.

OPTION — System Configuration Options **$0039**

	Bit 7	6	5	4	3	2	1	Bit 0
	ADPU	CSEL	IRQE*	DLY*	CME	FCME	CR1*	CR0*
RESET:	0	0	0	1	0	0	0	0

*Can be written only once in first 64 cycles out of reset in normal modes, or at any time in special modes

ADPU — A/D Power Up
0 = A/D powered down
1 = A/D powered up

CSEL — Clock Select
0 = A/D and EEPROM use system E clock
1 = A/D and EEPROM use internal RC clock

IRQE — Configure $\overline{IRQ}$ for Edge-Sensitive Only Operation
Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY.**

DLY — Enable Oscillator Startup Delay
Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY.**

CME — Clock Monitor Enable
Refer to **SECTION 5 RESETS AND INTERRUPTS.**

FCME — Force Clock Monitor Enable
Refer to **SECTION 5 RESETS AND INTERRUPTS.**

CR[1:0] — COP Timer Rate Select Bits
Refer to **SECTION 5 RESETS AND INTERRUPTS** and **SECTION 9 TIMING SYSTEM.**

10.3 Conversion Process

The A/D conversion sequence begins one E-clock cycle after a write to the A/D control/status register, ADCTL. The bits in ADCTL select the channel and the mode of conversion.

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 of this type, the source of each analog input should use V_{RH} as the supply voltage and be referenced to V_{RL}.

10.4 Channel Assignments

The multiplexer allows the 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 internal reference points or test functions, and four channels are reserved. Refer to Table 10–1.

Table 10–1. A/D Converter Channel Assignments

Channel Number	Channel Signal	Result in ADRx if MULT = 1
1	AN0	ADR1
2	AN1	ADR2
3	AN2	ADR3
4	AN3	ADR4
5	AN4	ADR1
6	AN5	ADR2
7	AN6	ADR3
8	AN7	ADR4
9	AN8	ADR1
10	AN9	ADR2
11	AN10	ADR3
12	AN11	ADR4
13	V_{RH}*	ADR1
14	V_{RL}*	ADR2
15	$(V_{RH})/2$*	ADR3
16	Reserved*	ADR4

*Used for factory testing

10.5 Single-Channel Operation

There are two types of single-channel operation. When SCAN = 0, the first type, the single selected channel is converted four consecutive times. The first result is stored in A/D result register 1 (ADR1), and the fourth result is stored in 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 type of single-channel operation, 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 overwriting ADR2, and so on.

10.6 Multiple-Channel Operation

There are two types of multiple-channel operation. When SCAN = 0, the first type, a selected group of four channels is converted one time each. The first result is stored in A/D result register 1 (ADR1), and the fourth result is stored in 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 type of multiple-channel operation, 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 overwriting ADR2, and so on.

10.7 Operation in STOP and WAIT Modes

If a conversion sequence is in progress 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 is resampled and the conversion sequence is 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 is necessary to allow a stabilization period when leaving the STOP mode. If the STOP mode is exited with a delay (DLY = 1), there is 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 = 0), allow 10 ms for the A/D circuitry to stabilize to avoid invalid results.

10.8 A/D Control/Status Registers

All bits in this register can be read or written, except bit 7, which is a read-only status indicator, and bit 6, which always reads as zero. Write to ADCTL to initiate a conversion. To quit a conversion in progress, write to this register and a new conversion sequence begins immediately.

ADCTL — A/D Control/Status **$0030**

	Bit 7	6	5	4	3	2	1	Bit 0
	CCF	—	SCAN	MULT	CD	CC	CB	CA
RESET:	0	0	U	U	U	U	U	U

CCF — Conversions Complete Flag

A read-only status indicator, this bit is set when all four A/D result registers contain valid conversion results. Each time the ADCTL register is overwritten, this bit is automatically cleared to zero and a conversion sequence is started. In the continuous mode, CCF is set at the end of the first conversion sequence.

Bit 6 — Not implemented

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 updated as data becomes available.

MULT — Multiple Channel/Single Channel Control

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

NOTE

When the multiple-channel continuous scan mode is used, extra care is needed in the design of circuitry driving the A/D inputs. The charge on the capacitive DAC array before 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 µs 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 errors in accuracy. Refer to *M68HC11 Reference Manual* (M68HC11RM/AD) for further information.

CD–CA — Channel Selects D–A

Refer to Table 10–2. 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 is to be converted.

Table 10–2. A/D Converter Channel Selection

Channel Select Control Bits CD:CC:CB:CA	Channel Signal	Result in ADRx if MULT = 1
0000	AN0	ADR1
0001	AN1	ADR2
0010	AN2	ADR3
0011	AN3	ADR4
0100	AN4	ADR1
0101	AN5	ADR2
0110	AN6	ADR3
0111	AN7	ADR4
1000	AN8	ADR1
1001	AN9	ADR2
1010	AN10	ADR3
1011	AN11	ADR4
1100	V_{RH}*	ADR1
1101	V_{RL}*	ADR2
1110	$(V_{RH})/2$*	ADR3
1111	Reserved*	ADR4

*Used for factory testing

10.9 A/D Converter Result Registers

These read-only registers hold an 8-bit conversion result. Writes to these registers have no effect. Data in the A/D converter result registers is valid when the CCF flag in the ADCTL register is set, indicating a conversion sequence is complete. If conversion results are needed sooner, refer to Figure 10–3, which shows the A/D conversion sequence diagram.

ADR1–ADR4 — A/D Result Registers 1–4 **$0031–$0034**

$0031	Bit 7	6	5	4	3	2	1	Bit 0	ADR1
$0032	Bit 7	6	5	4	3	2	1	Bit 0	ADR2
$0033	Bit 7	6	5	4	3	2	1	Bit 0	ADR3
$0034	Bit 7	6	5	4	3	2	1	Bit 0	ADR4

SECTION 11
DIGITAL-TO-ANALOG CONVERTER

The digital-to-analog (D/A) system converts digital values to analog voltages. It has two independent output channels.

11.1 Overview

The M68HC11 N-series D/A converter system has two independent output channels, two input data registers, and a control register. The system outputs an analog voltage between V_{SS} and V_{DD} based on a digital input value between $00 and $FF. The output of each channel is independent and monotonic. Figure 11–1 shows the D/A system block diagram. Figure 11–2 shows the D/A system output voltage related to digital input values.

The maximum output current is 1.0 µA at 5 volts. Worst case settling time is 5 µs after a write to the data register. The output load for each channel should have no resistive paths to ground. During STOP mode the D/A outputs become open to prevent additional current drain.

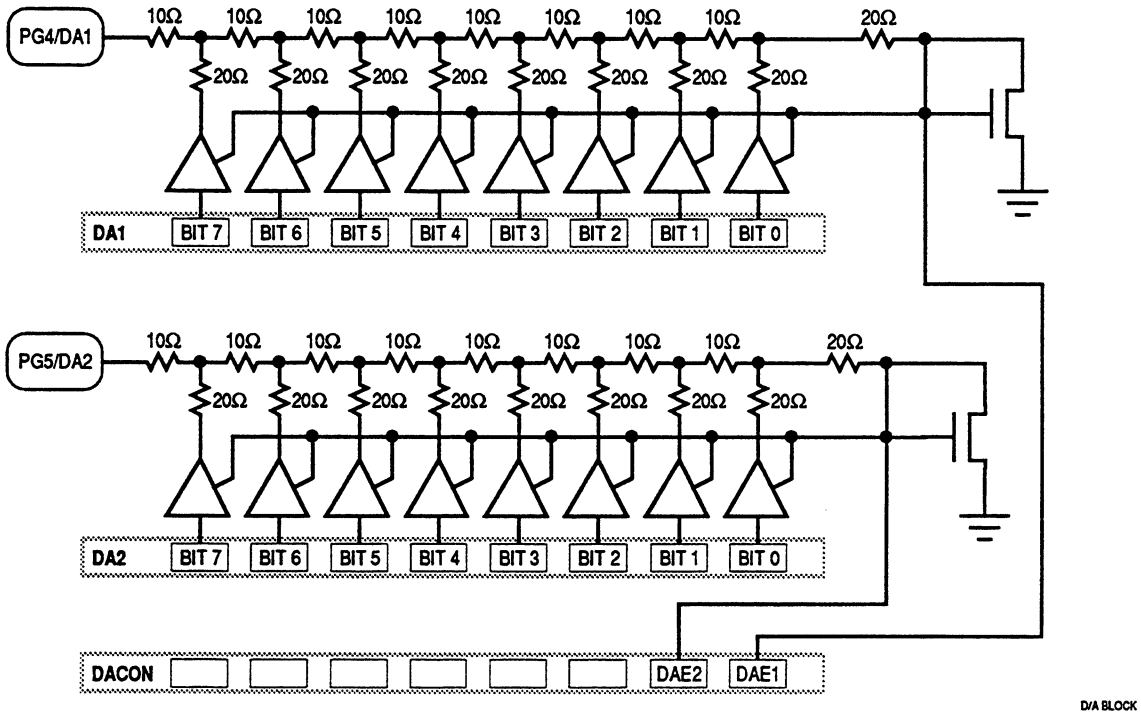


Figure 11–1. Digital-to-Analog Converter Block Diagram

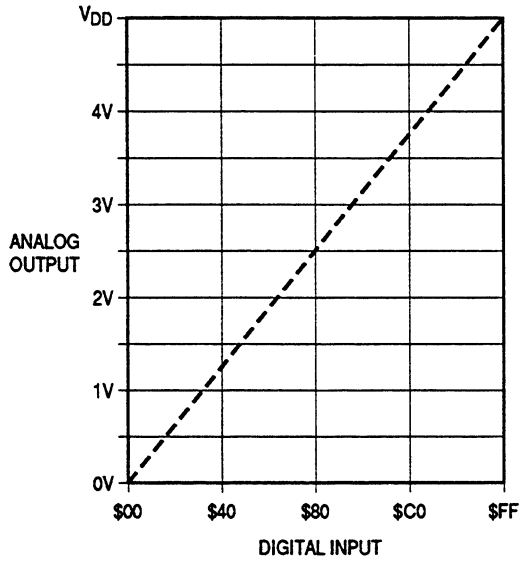


Figure 11–2. Digital Input Related to Analog Output

11.2 D/A Converter Control Register

The D/A converter system control register is used to enable the D/A output channels. When a channel is enabled, the associated port G pin becomes an output even though port G pins 4 and 5 are normally input only. Care should be taken not to enable a D/A channel while its associated port G pin is being used as a general-purpose input. When a D/A channel is disabled, the associated port G pin becomes input only. There are no DDR bits corresponding to port G pins 4 and 5.

DACON — Digital-to-Analog System Control **$004D**

	Bit 7	6	5	4	3	2	1	Bit 0
	—	—	—	—	—	—	DAE2	DAE1
RESET:	0	0	0	0	0	0	0	0

Bits [7:2] — Not implemented
Always read zero

DAE2 — D/A Channel 2 Enable
0 = D/A channel 2 disabled
1 = D/A channel 2 enabled

DAE1 — D/A Channel 1 Enable
0 = D/A channel 1 disabled
1 = D/A channel 1 enabled

11.3 D/A Data Registers

The D/A system data registers are used to input the digital values that are converted to analog voltages. Values in this register are converted such that digital values ranging from $00 to $FF result in output voltages ranging in equal increments from V_{SS} to V_{DD}. Refer to Figure 11–2. A value written to a data register while its channel is disabled has no meaning or effect but will be the first value converted when the channel is enabled.

DA1 — Digital-to-Analog System Channel 1 Data Register **$004E**

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

Bits [7:0] — Digital Input Data for D/A Channel 1
Resets to $00

DA2 — Digital-to-Analog System Channel 2 Data Register **$004E**

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

Bits [7:0] — Digital Input Data for D/A Channel 2
Resets to $00

SECTION 12
ARITHMETIC LOGIC UNIT

The M68HC11 N-series arithmetic logic unit (ALU) performs 16-bit integer multiplication and division as a coprocessor. It performs signed or unsigned multiplication, with or without accumulated product, as well as signed or unsigned division. Because the arithmetic operations are independently executed, the CPU is free to perform other operations. This reduces CPU overhead and increases system performance.

12.1 Overview

The ALU consists of three data registers, one control register, and one status register. The three data registers are AREG, BREG, and CREG. AREG is a 16-bit register that contains the value of the multiplicand or the divisor. BREG is a 16-bit register that contains the value of the multiplier or the remainder. CREG is a 32-bit register that contains the value of the product, accumulated product, numerator before division, or quotient after division. CREG is considered as two 16-bit registers CL (CREG low) and CH (CREG high). ALU registers are mapped in the peripheral register area and can be read or written by the CPU. Figure 12–1 shows the coprocessor structure and which registers are involved during different operations. Figure 12–2 shows the ALU data format.

Arithmetic operations are started when the lower byte of the register indicated in Figure 12–1 is written. Any operation can be started at any time by setting the TRG bit in ALUC register. FDIV can only be started by setting the TRG bit.

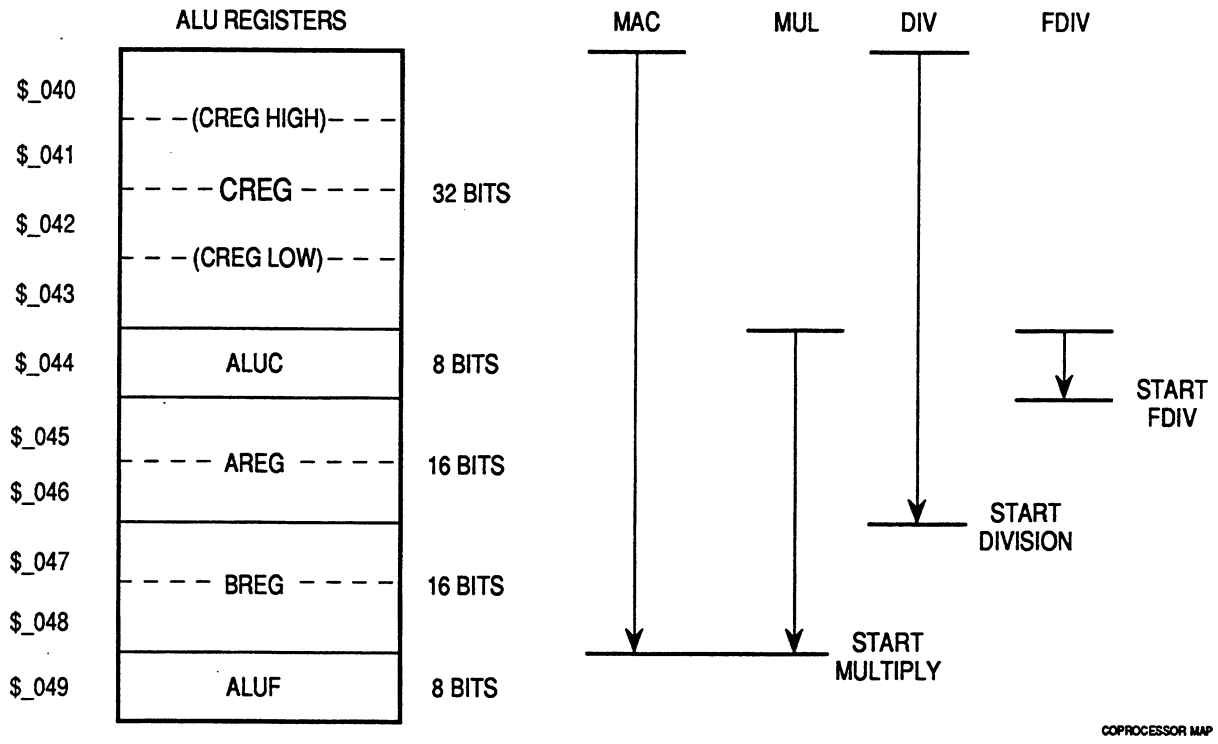


Figure 12–1. ALU Register Map

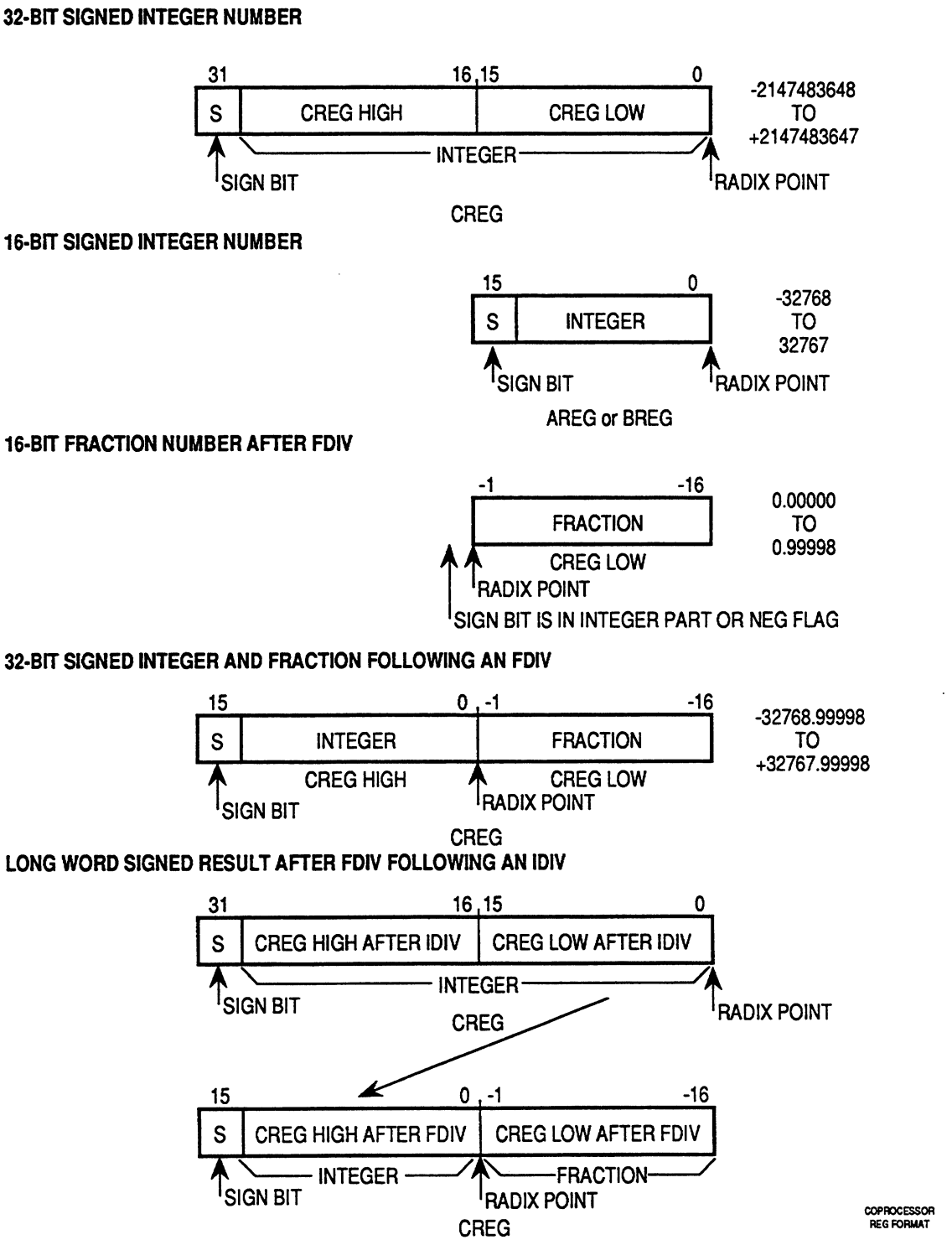


Figure 12–2. ALU Operations

12.2 ALU Data Format

The signed data which is used for both input and output consists of two's complement binary integers. Table 12–1 shows the ranges that are allowed. Table 12–2 shows expression of the numbers. Table 12–3 shows how fractional numbers are represented. Figure 12–2 shows a complete description of ALU data operations.

Table 12–1. Number Ranges for ALU Registers

Register	Size	Unsigned	Signed
AREG	16 bits	0 to 65535	–32,768 to +32,767
BREG	16 bits	0 to 65535	–32,768 to +32,767
CREG	32 bits	0 to 4,294,967,295	–2,147,483,648 to +2,147,483,647

Table 12–2. Signed Number Expression

Decimal	16-Bit Hexadecimal	32-Bit Hexadecimal
+2,147,483,647	—	$7FFF FFFF
•	—	•
•	—	•
+32,767	$7FFF	$0000 7FFF
•	•	•
•	•	•
+1	$0001	$0000 0001
0	$0000	$0000 0000
–1	$FFFF	$FFFF FFFF
–2	$FFFE	$FFFF FFFE
•	•	•
•	•	•
–32,768	$8000	$FFFF 8000
•	—	•
•	—	•
–2,147,483,647	—	$8000 0000

In the case of fractional division, the number held in CL register does not contain the sign bit. A fractional quotient stored in CL has its radix point to the left of bit 15 (DCC bit in ALUC register = 1). The sign bit will either be in the integer portion of the quotient (held in CH) or indicated by NEG = 1 in ALUF register. Signed data is in two's complement binary format. Table 12–3 shows how these numbers are expressed.

Table 12–3. Fractional Number Expression

Decimal	16-Bit Hexadecimal
+0.99998	$FFFF
•	•
•	•
+0.5	$8000
+0.25	$4000
+0.125	$2000
+0.0625	$1000
+0.03125	$0800
+0.015625	$0400
•	•
•	•
+0.0000153	$0001
–0.99998	$0001
•	•
•	•
–0.5	$8000
–0.25	$C000
–0.125	$E000
–0.0625	$F000
–0.03125	$F800
–0.015625	$FC00
•	•
•	•
–0.0000153	$FFFF

12.3 Arithmetic Operations

The ALU performs 16-bit integer multiplication and division and signed or unsigned multiplication, with or without accumulated product, as well as signed or unsigned division. Below are the times in which the ALU performs its functions.

Signed or unsigned Multiplication without accumulated product:

AREG * BREG → CREG (product)
20 E clock cycles (5 µs at 4 MHz)

Signed or unsigned Multiplication with accumulated product:

AREG * BREG + CREG → CREG
20 E clock cycles (5 µs at 4 MHz)

Signed or unsigned division:

CREG ÷ AREG → CREG (quotient), BREG (remainder)
Unsigned — 33 E clock cycles (8.25 µs at 4 MHz)
Signed — 35 E clock cycles (8.75 µs at 4 MHz)

Signed or unsigned fractional division:

CREG (CL) → CREG (CH)
BREG ÷ AREG → CREG (CL) (fractional quotient), BREG (remainder)
Unsigned — 17 E clock cycles (4.25 µs at 4 MHz)
Signed — 18 E clock cycles (4.5 µs at 4 MHz)

12.3.1 Fractional Division

The ALU can perform signed or unsigned 16-bit fractional divide (FDIV) function. Before this division, the BREG may contain the remainder of previous integer division (IDIV). The remainder in BREG must be less than the divisor in AREG. This ALU function performs an integer divide followed by a fractional divide. The quotient may have a radix point to the left of bit 15 in CL register. To begin an FDIV operation, set the TRG bit in ALUC register. Refer to Table 12–4.

Since the previous contents of CL moved to CH, the CREG contains an integer plus fractional value which has a radix point at the center of CREG after this FDIV. In the case of signed concatenated fractional division, the previous division should be performed with DCC bit in ALUC register set. Final division should be performed with DCC bit clear for the sign adjustment of the concatenated quotient. Refer to the following.

Single division cases:
IDIV (DCC = 0)
FDIV (DCC = 0)

CREG ÷ AREG → CREG (quotient), BREG (remainder)

Concatenated signed division cases:
IDIV (DCC = 1) → FDIV (DCC = 0)
IDIV (DCC = 1) → FDIV (DCC = 1) → FDIV (DCC = 0)
IDIV (DCC = 1) → FDIV (DCC = 1).....FDIV (DCC = 1) → FDIV (DCC = 0)

CREG (CL) → CREG (CH)
BREG ÷ AREG → CREG (CL) (fractional quotient), BREG (remainder)

12.3.2 Multiplication

The ALU can perform signed or unsigned multiplication with or without accumulated product. During a multiply operation, AREG (high/low) and BREG (high/low) hold the values of the multiplier and multiplicand. Since multiplication is commutative, it does not matter which register holds the multiplier or multiplicand. However, AREG must be written first because writing to the lower byte of BREG starts the multiply process. Refer to Table 12–4. When the ACF bit in ALUF register is set, reads of BREG return the last multiplication result. When multiplying using the accumulated product method, CREG holds the value of the last multiply plus the value of the previous product. The value in CREG is lost when read following a multiply operation. For signed multiplication, SIG bit in ALUC must be set. For accumulated product multiplication, MAC bit in ALUC must be set. Refer to the following.

Multiply without accumulated product:
UMUL (SIG = 0, DIV = 0, MAC = 0)
SMUL (SIG = 1, DIV = 0, MAC = 0)

AREG * BREG → CREG (product)

Multiply with accumulated product:
UMUL (SIG = 0, DIV = 0, MAC = 1) → UMUL (SIG = 0, DIV = 0, MAC = 1)
SMUL (SIG = 1, DIV = 0, MAC = 1) → SMUL (SIG = 1, DIV = 0, MAC = 1)

AREG * BREG + CREG → CREG

12.4 ALU Registers

The following paragraphs explain the functions of ALU data, control, and status registers. All data registers containing integer values have an implied fixed radix point at the right of bit 0.

12.4.1 ALUC Arithmetic Logic Unit Control Register

Control of the ALU is provided by the ALUC register. ALUC can be read and written at any time. Refer to Table 12–4 for ALUC control bits and their relation to each arithmetic function.

Table 12–4. ALUC Control Bits

SIG	DIV	MAC	DCC	FUNCTION	TRIGGERS
0	0	0	X	Unsigned MUL	Write to BREG low or set TRG
1	0	0	X	Signed MUL	Write to BREG low or set TRG
0	0	1	X	Unsigned MAC	Write to BREG low or set TRG
1	0	1	X	Signed MAC	Write to BREG low or set TRG
0	1	0	X	Unsigned IDIV	Write to AREG low or set TRG
1	1	0	0	Signed IDIV	Write to AREG low or set TRG
1	1	0	1	Signed IDIV DCC	Write to AREG low or set TRG
0	1	1	X	Unsigned FDIV	Set TRG
1	1	1	0	Signed FDIV	Set TRG
1	1	1	1	Signed FDIV DCC	Set TRG

ALUC — Arithmetic Logic Unit Control **$0044**

	Bit 7	6	5	4	3	2	1	Bit 0
	SIG	DIV	MAC	DCC	TRG	—	—	—
RESET:	0	0	0	0	0	0	0	0

SIG — Signed Number Enable
- 0 = AREG, BREG, and CREG contents are unsigned numbers
- 1 = AREG, BREG, and CREG contents are signed numbers

DIV — Division Enable
- 0 = Division disabled
- 1 = Division enabled

MAC — Multiply with Accumulated Product Enable
- 0 = Multiply with accumulated product disabled
- 1 = Multiply with accumulated product enabled. During a MAC operation, the accumulated product is held in CREG and is added to subsequent multiplications.

DCC — Division Compensation for Concatenated Quotient Enable
- 0 = DCC disabled
- 1 = DCC enabled. For signed IDIV and signed FDIV, causes CREG to contain both integer result **plus** fractional amount to right of radix point. Radix point is to the left of bit 15 in CREG. Refer to Figure 12–2.

TRG — Function Start Trigger Bit
Always reads zero
- 0 = No effect
- 1 = Writing this bit to one starts the function

Bits [2:0] — Not implemented
Always read zero

12.4.2 CREG Data Register C

CREG, considered as two 16-bit registers (CH and CL), holds the product or accumulated product after multiplication, or the numerator before division and the quotient after division. There is an implied fixed radix point at the right of bit zero.

During fractional division, the fractional number in CL does not include the sign bit. The quotient is placed in CL and the remainder is placed in BREG. The previous contents of CL are moved into CH after the fractional division. The signed data that is used for both input or output is two's complement binary integer format.

CREG — Data Register C **$0040–$0043**

$0040	Bit 31	30	29	28	27	26	25	Bit24	CREG (High)
$0041	Bit 23	22	21	20	19	18	17	Bit 16	CREG (Mid-High)
$0042	Bit 15	14	13	12	11	10	9	Bit 8	CREG (Mid-Low)
$0043	Bit 7	6	5	4	3	2	1	Bit 0	CREG (Low)

CREG reset value is indeterminate.

12.4.3 AREG ALU Data Register A

AREG holds the value of the multiplicand or the divisor. There is an implied fixed radix point at the right of bit 0.

AREG — Data Register A **$0045, $0046**

	Bit 7	6	5	4	3	2	1	Bit 0	
$0045	Bit 15	14	13	12	11	10	9	Bit 8	AREG (High)
$0046	Bit 7	6	5	4	3	2	1	Bit 0	AREG (Low)

AREG reset value is indeterminate.

12.4.4 BREG ALU Data Register B

BREG holds the multiplier or remainder after division. There is an implied fixed radix point at the right of bit 0.

BREG — Data Register B **$0047, $0048**

	Bit 7	6	5	4	3	2	1	Bit 0	
$0047	Bit 15	14	13	12	11	10	9	Bit 8	BREG (High)
$0048	Bit 7	6	5	4	3	2	1	Bit 0	BREG (Low)

BREG reset value is indeterminate.

12.4.5 ALUF ALU Status Register

The ALUF register indicates the status of the operation just performed.

ALUF — Arithmetic Logic Unit Status Flag Register **$0049**

	Bit 7	6	5	4	3	2	1	Bit 0
	NEG	RZF	—	—	—	OVF	DZF	ACF
RESET:	0	0	0	0	0	0	0	0

NEG — Negative Result

Read only bit. Writes to this bit do not affect the value. Valid until next write to AREG or until start of next arithmetic operation.

0 = Result is positive value

1 = Result is negative value

RZF — Remainder Zero

Read only bit. Writes to this bit do not affect the value. Valid until next write to AREG or until start of next arithmetic operation.

0 = Remainder in BREG after FDIV or IDIV is not zero

1 = Remainder in BREG after FDIV or IDIV is zero

Bits [5:3] — Not implemented
Always read zero

OVF — Overflow Flag
Cleared automatically by write to this register with bit 2 set.
0 = Overflow from MSB bit on CREG not detected
1 = Overflow from MSB bit on CREG detected

DZF — Divide by Zero Flag
Cleared automatically by a write to this register with bit 1 set.
0 = Divide by zero condition not detected
1 = Divide by zero condition detected

ACF — Arithmetic Completion Flag
Cleared automatically by write to this register with bit 0 set.
0 = Arithmetic operation not completed
1 = Arithmetic operation completed

APPENDIX A
ELECTRICAL CHARACTERISTICS

This appendix contains electrical parameters for M68HC11 N-series microcontrollers.

Table A–1. Maximum Ratings

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 MC68HC(7)11N4C MC68HC(7)11N4V	T_A	T_L to T_H −40 to +85 −40 to +105	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Current Drain per Pin* Excluding V_{DD}, V_{SS}, AV_{DD}, V_{RH}, and V_{RL}	I_D	25	mA

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

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) 80-Pin Plastic Quad Flat Pack 80-Pin Windowed Ceramic Quad Flat Pack	Θ_{JA}	 50 50	 °C/W °C/W
Total Power Dissipation (Note 1)	P_D	$P_{INT} + P_{I/O}$ $K / (T_J + 273°C)$	W
Device Internal Power Dissipation	P_{INT}	$I_{DD} \times V_{DD}$	W
I/O Pin Power Dissipation (Note 2)	$P_{I/O}$	User-determined	W
A Constant (Note 3)	K	$P_D \times (T_A + 273°C) + \Theta_{JA} \times P_D^2$	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

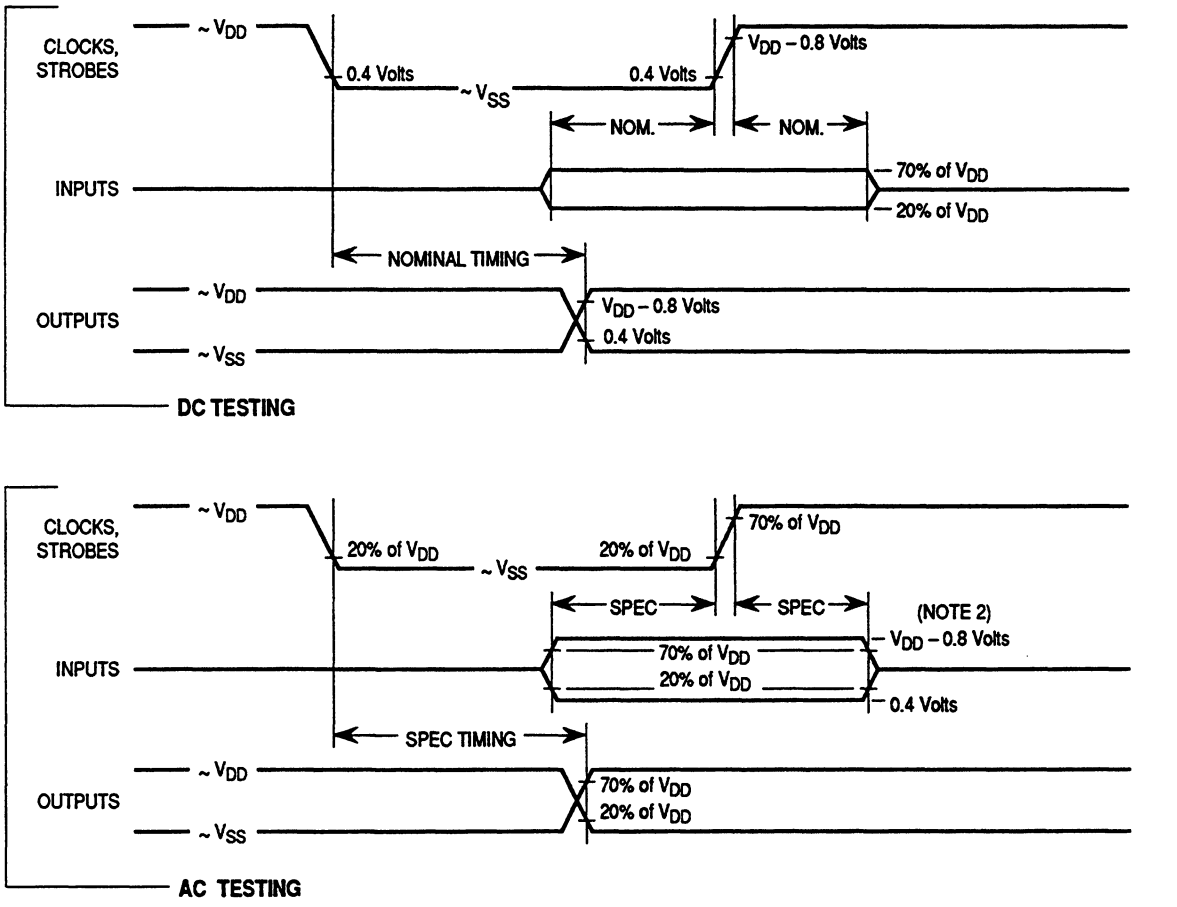
V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted

Characteristic	Symbol	Min	Max	Unit
Output Voltage (Note 1) All Outputs except XTAL	V_{OL}	—	0.1	V
All Outputs Except XTAL, RESET, and MODA I_{Load} = ± 10.0 µA	V_{OH}	V_{DD} – 0.1	—	V
Output High Voltage (Note 1) All Outputs Except XTAL, 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 RESET	V_{IH}	0.7 x V_{DD}	V_{DD} + 0.3	V
RESET		0.8 x V_{DD}	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 Ports A, B, C, D, F, G, H, V_{in} = V_{IH} or V_{IL} MODA/LIR, RESET	I_{OZ}	—	±10	µA
Input Leakage Current (Note 2)	I_{in}			
V_{in} = V_{DD} or V_{SS} IRQ, XIRQ on standard devices		—	±1	µA
V_{in} = V_{DD} or V_{SS} MODB/V_{STBY}, XIRQ on EPROM devices		—	±10	µA
Input Current with Pull-up Resistors V_{in} = V_{IL} Ports B, F, G, and H	I_{ipr}	100	500	µA
RAM Standby Voltage Power down	V_{SB}	2.0	V_{DD}	V
RAM Standby Current Power down	I_{SB}	—	10	µA
Input Capacitance PE[7:0], PG[7:0], IRQ, XIRQ, EXTAL	C_{in}	—	8	pF
Ports A, B, C, D, F, H, MODA/LIR, RESET		—	12	pF
Output Load Capacitance All Outputs Except PD[4:1], XOUT, XTAL, MODA/LIR	C_L	—	90	pF
PD[4:1]		—	200	pF
XOUT		—	30	pF

Characteristic	Symbol	2 MHz	3 MHz	4 MHz	Unit
Maximum Total Supply Current (Note 3)					
RUN:	I_{DD}				
Single-Chip Mode		27	32	40	mA
Expanded Mode		35	42	50	mA
WAIT: (All Peripheral Functions Shut Down)	W_{IDD}				
Single-Chip Mode		10	15	20	mA
Expanded Mode		12	17	22	mA
STOP:	S_{IDD}				
No Clocks, Single-Chip Mode		50	50	50	µA
Maximum Power Dissipation	P_D				
Single-Chip Mode		149	176	220	mW
Expanded Mode		193	231	275	mW

NOTES:

1. V_{OH} specification for 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; t_{cyc} = 250 ns for 4 MHz rating; $V_{IL} \le 0.2$ V; $V_{IH} \ge V_{DD} - 0.2$ V; No dc loads.



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 V_{DD} – 0.8 volts while timing measurements are taken at the 20% and 70% of V_{DD} points.

TEST METHODS 2

Figure A–1. Test Methods

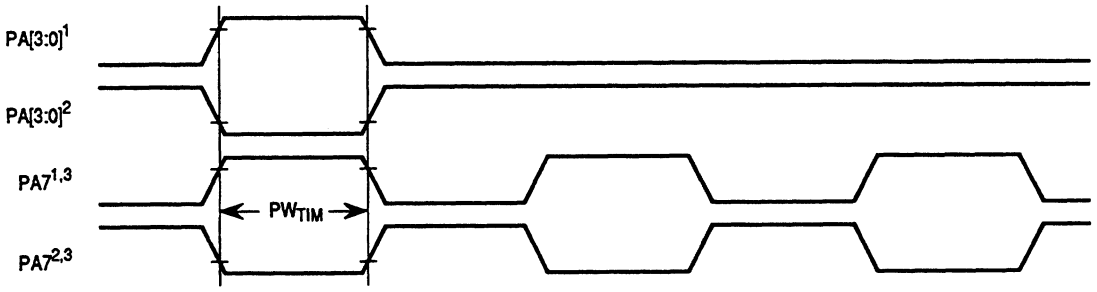
Table A–4. Control Timing

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

Characteristic	Symbol	2.0 MHz		3.0 MHz		4.0 MHz		Unit
		Min	Max	Min	Max	Min	Max	
Frequency of Operation	f_o	dc	2.0	dc	3.0	dc	4.0	MHz
E-Clock Period	t_{cyc}	500	—	333	—	250	—	ns
Crystal Frequency	f_{XTAL}	—	8.0	—	12.0	—	16.0	MHz
External Oscillator Frequency	$4 f_o$	dc	8.0	dc	12.0	dc	16.0	MHz
Processor Control Setup Time $t_{PCSU} = 1/4\ t_{cyc} + 50$ ns	t_{PCSU}	175	—	133	—	112	—	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} 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}	520	—	353	—	270	—	ns
Wait Recovery Startup Time	t_{WRS}	—	4	—	4	—	4	t_{cyc}
Timer Pulse Width, Input Capture Pulse Accumulator Input $PW_{TIM} = t_{cyc} + 20$ ns	PW_{TIM}	520	—	353	—	270	—	ns

NOTES:

1. 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 **SECTION 5 RESETS AND INTERRUPTS** for further detail.
2. All timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted.



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

TIMER INPUTS TIM

Figure A–2. Timer Inputs

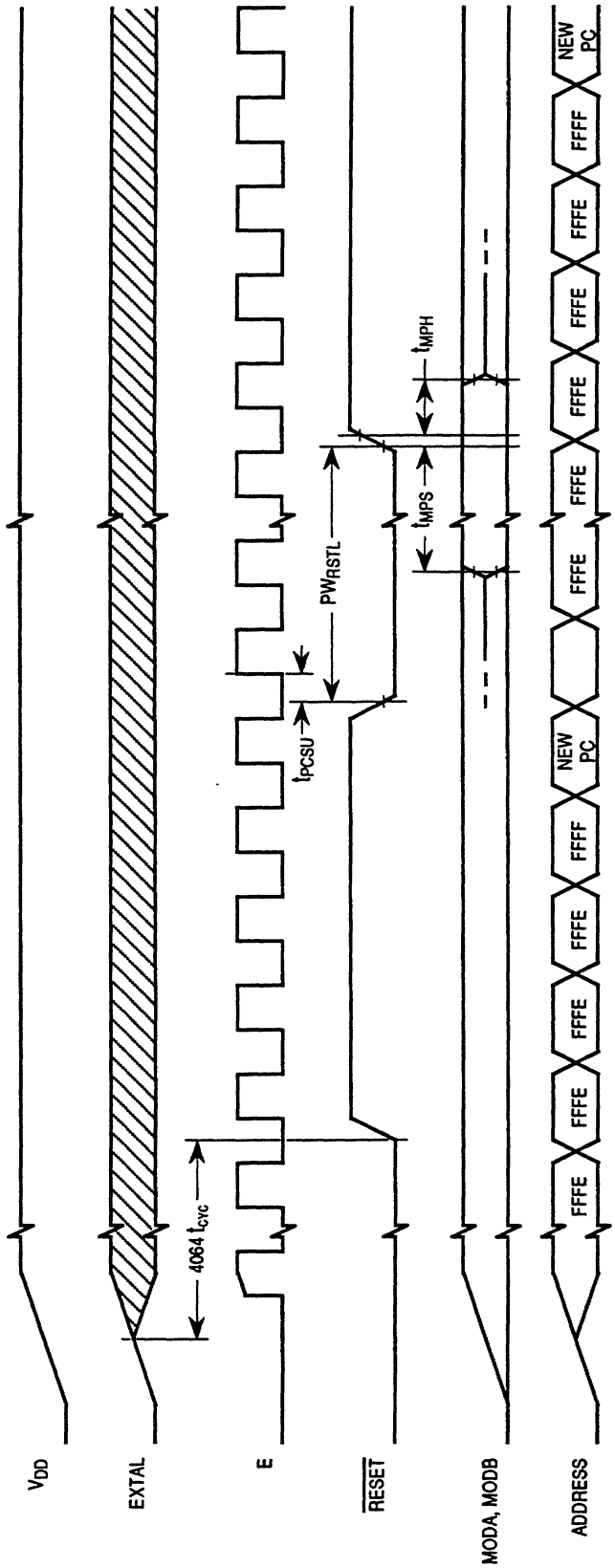
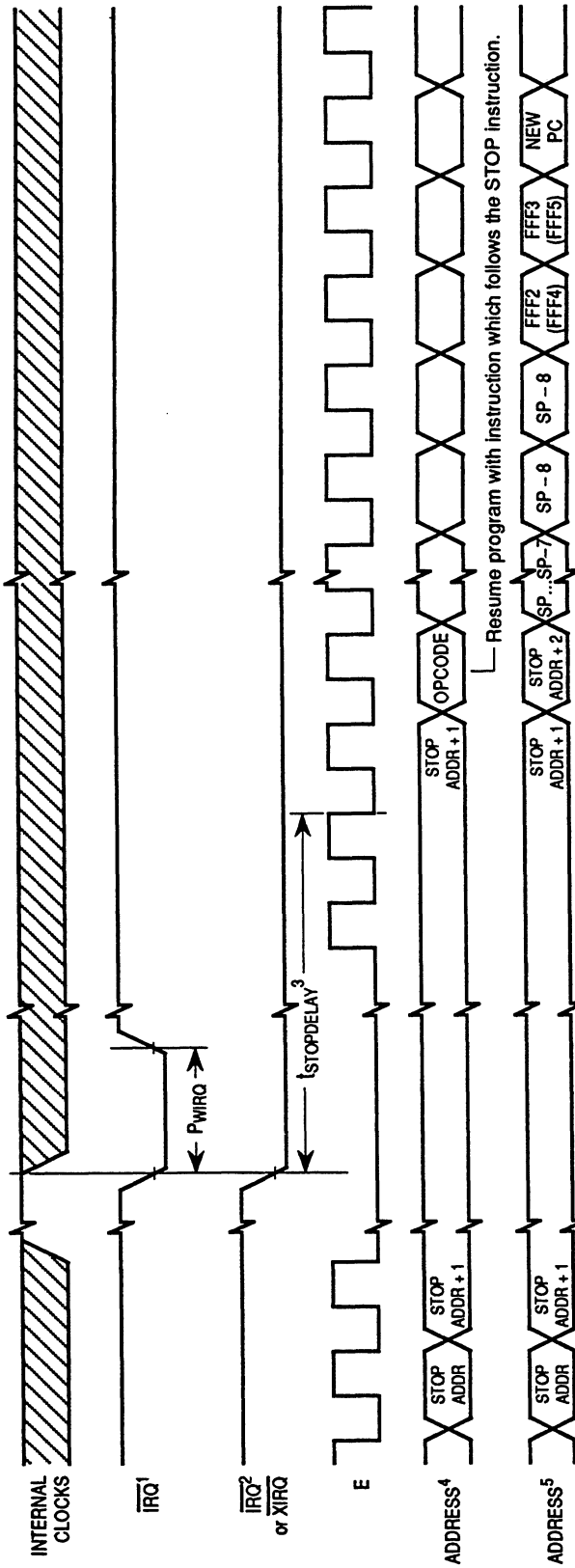


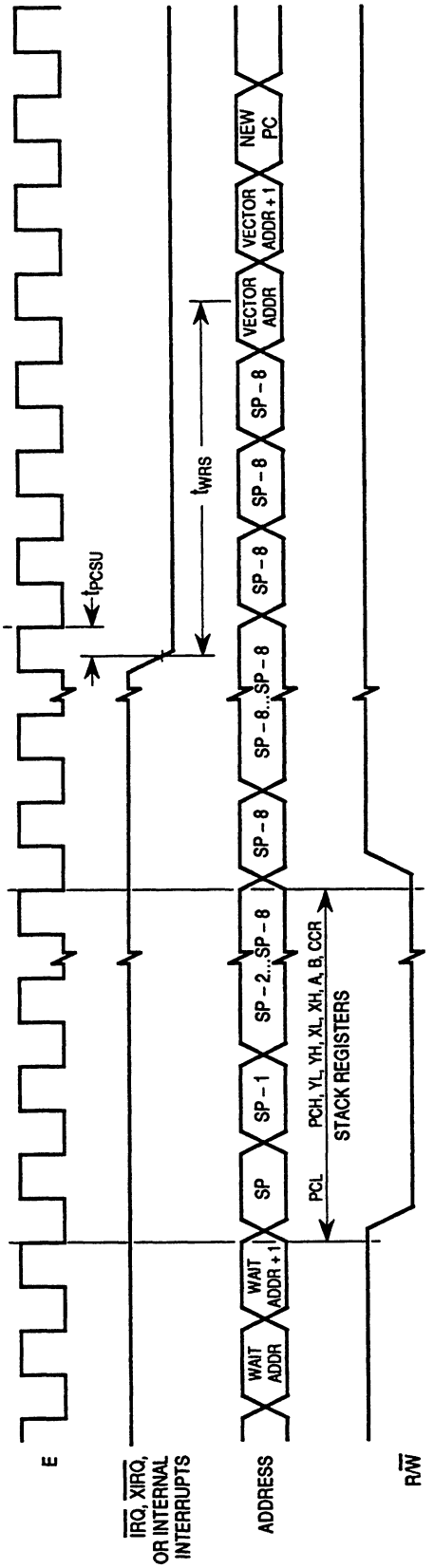
Figure A–3. POR External Reset Timing Diagram



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

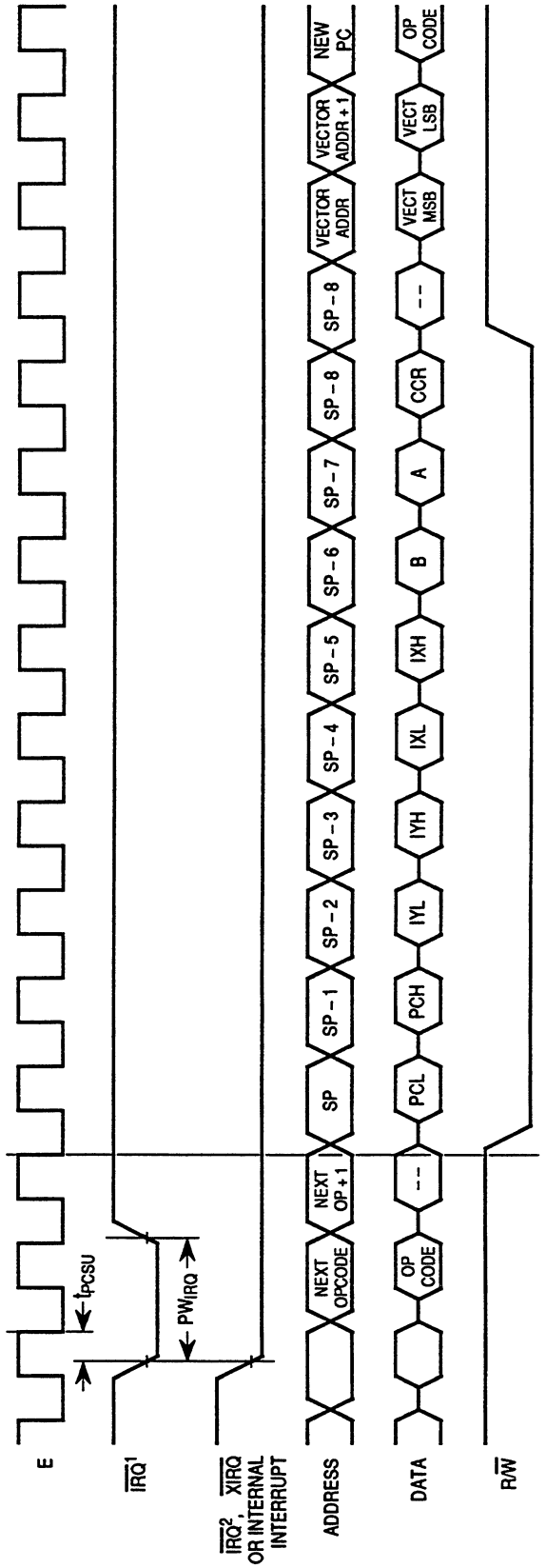
Figure A–4. STOP Recovery Timing Diagram



NOTE: $\overline{\text{RESET}}$ also causes recovery from WAIT.

WAIT RECOVERY TIM

Figure A–5. WAIT Recovery from Interrupt Timing Diagram



NOTES:
1. Edge sensitive $\overline{IRQ}$ pin (IRQE bit = 1)
2. Level sensitive $\overline{IRQ}$ pin (IRQE bit = 0)

INTERRUPT TIM

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	2.0 MHz		3.0 MHz		4.0 MHz		Unit
		Min	Max	Min	Max	Min	Max	
Frequency of Operation (E-Clock Frequency)	f_o	dc	2.0	dc	3.0	dc	4.0	MHz
E-Clock Period	t_{cyc}	500	—	333	—	250	—	ns
Peripheral Data Setup Time (MCU Read of Ports A, B, C, D, E, F, G, and H)	t_{PDSU}	100	—	100	—	100	—	ns
Peripheral Data Hold Time (MCU Read of Ports A, B, C, D, E, F, G, and H)	t_{PDH}	50	—	50	—	50	—	ns
Delay Time, Peripheral Data Write (MCU Write to Ports A, B, G, and H)	t_{PWD}	—	200	—	200	—	200	ns
(MCU Write to Ports C, D, and F; t_{PWD} = 1/4 t_{cyc} + 100 ns)		—	225	—	183	—	162	ns

NOTES:

1. Port C and D timing is valid for active drive (CWOM and DWOM bits not set in OPT2 and SPCR registers respectively).
2. All timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted.

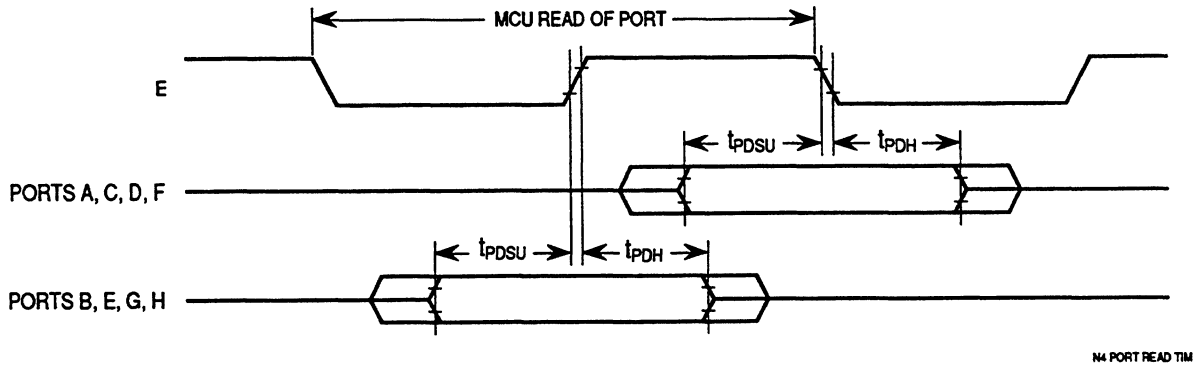


Figure A–7. Port Read Timing Diagram

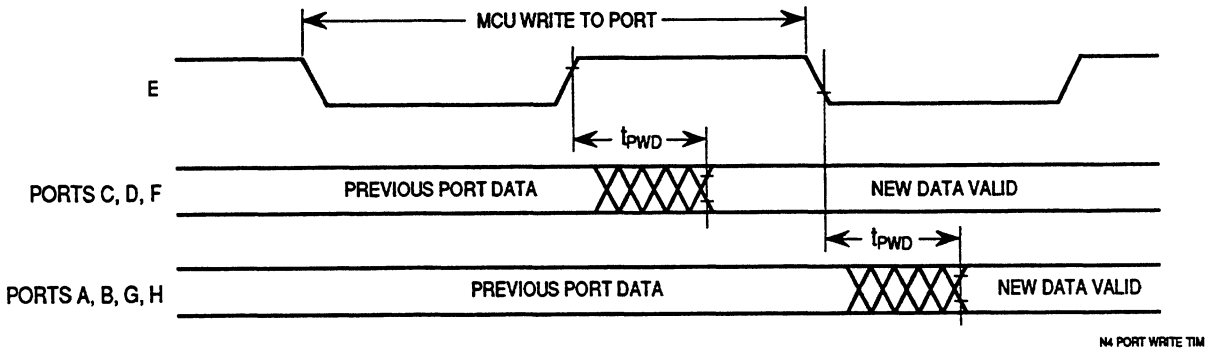


Figure A–8. Port Write Timing Diagram

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 ≤ 4.0 MHz, unless otherwise noted

Characteristic	Parameter	Min	Absolute	2.0 MHz	3.0 MHz	4.0 MHz	Unit
				Max	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	± 1	LSB
Zero Error	Difference Between the Output of an Ideal and an Actual for Zero Input Voltage	—	—	± 1/2	± 1	± 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	± 1	LSB
Total Unadjusted Error	Maximum Sum of Non-Linearity, Zero Error, and Full-Scale Error	—	—	± 1/2	± 1 1/2	± 1 1/2	LSB
Quantization Error	Uncertainty Because of Converter Resolution	—	—	± 1/2	± 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	± 2	LSB
Conversion Range	Analog Input Voltage Range	V_{RL}	—	V_{RH}	V_{RH}	V_{RH}	V
V_{RH}	Maximum Analog Reference Voltage (Note 2)	V_{RL}	—	V_{DD} + 0.1	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_{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: E Clock Internal RC Oscillator	— —	32 —	— t_{cyc} + 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	FF	Hex
Sample Acquisition Time	Analog Input Acquisition Sampling Time: E Clock Internal RC Oscillator	— —	12 —	— 12	— 12	— 12	t_{cyc} µs
Sample/Hold Capacitance	Input Capacitance during Sample PE[7:0]	—	20 (Typ)	—	—	—	pF
Input Leakage	Input Leakage on A/D Pins PE[7:0] V_{RL}, V_{RH}	— —	— —	400 1.0	400 1.0	400 1.0	nA µA

NOTES:

1. For f_{op} < 2 MHz, source impedances should equal approximately 10 kΩ. For f_{op} ≥ 2 MHz, source impedances should equal approximately 5 kΩ – 10 kΩ. 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%.

Table A–7. Digital-to-Analog Converter Characteristics

V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, 750 kHz ≤ E ≤ 4.0 MHz, unless otherwise noted

Characteristic	Parameter	Min	Absolute	2.0 MHz	3.0 MHz	4.0 MHz	Unit
				Max	Max	Max	
Resolution	Number of Bits Resolved by D/A Converter	—	8	—	—	—	Bits
Non-Linearity	Maximum Deviation from the Ideal D/A Transfer Characteristics	—	—	± 1/2	± 1	± 1	LSB
Zero Error	Difference Between the Output of an Ideal and an Actual for Zero Input Voltage	—	—	± 1/2	± 1	± 1	LSB
Full Scale Error	Difference Between the Output of an Ideal and an Actual D/A for Full-Scale Input Voltage	—	—	± 1/2	± 1	± 1	LSB
Total Unadjusted Error	Maximum Sum of Non-Linearity, Zero Error, and Full-Scale Error	—	—	± 1/2	± 1 1/2	± 1 1/2	LSB
Quantization Error	Uncertainty Because of Converter Resolution	—	—	± 1/2	± 1/2	± 1/2	LSB
Absolute Accuracy	Difference Between the Equivalent of the Binary Input Code and the Actual Output Voltage, All Error Sources Included	—	—	± 1	± 2	± 2	LSB
Conversion Range	Analog Output Voltage Range	V_{SS}	—	AV_{DD}	AV_{DD}	AV_{DD}	V
Conversion Time	Total Time to Perform a Single Digital-to-Analog Conversion: E Clock Internal RC Oscillator	 — —	 32 —	 — t_{cyc} + 32	 — t_{cyc} + 32	 — t_{cyc} + 32	 t_{cyc} µs
Monotonicity	Conversion Result Never Decreases with an Increase in Output Voltage and has no Missing Codes	—	Guaranteed	—	—	—	—
Zero Input Reading	Conversion Result when Input = $00	V_{SS}	—	—	—	—	V
Full Scale Input Reading	Conversion Result when Input = $FF	—	—	V_{DD}	V_{DD}	V_{DD}	V
Output Current	Maximum Output Current at 5.0 V	—	—	1.0	1.0	1.0	µA
Settling Time	Maximum Time For Output to Stabilize	—	—	5.0	5.0	5.0	µs

Table A–8. Expansion Bus 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		4.0 MHz		Unit
			Min	Max	Min	Max	Min	Max	
	Frequency of Operation (E-Clock Frequency)	f_o	dc	2.0	dc	3.0	dc	4.0	MHz
1	Cycle Time $t_{cyc} = 1/f_o$	t_{cyc}	500	—	333	—	250	—	ns
2	Pulse Width, E Low $PW_{EL} = 1/2\ t_{cyc} - 20$ ns	PW_{EL}	230	—	147	—	105	—	ns
3	Pulse Width, E High $PW_{EH} = 1/2\ t_{cyc} - 25$ ns	PW_{EH}	225	—	142	—	100	—	ns
4A	E Clock Rise Time	t_r	—	20	—	20	—	20	ns
4B	Fall Time	t_f	—	20	—	18	—	15	ns
9	Address Hold Time $t_{AH} = 1/8\ t_{cyc} - 10$ ns	t_{AH}	53	—	32	—	21	—	ns
11	Address Delay Time $t_{AD} = 1/8\ t_{cyc} + 40$ ns	t_{AD}	—	103	—	82	—	71	ns
12	Address Valid Time to E Rise $t_{AV} = PW_{EL} - t_{AD}$	t_{AV}	128	—	65	—	34	—	ns
17	Read Data Setup Time	t_{DSR}	30	—	30	—	20	—	ns
18	Read Data Hold Time	t_{DHR}	0	—	0	—	0	—	ns
19	Write Data Delay Time	t_{DDW}	—	40	—	40	—	40	ns
21	Write Data Hold Time $t_{DHW} = 1/8\ t_{cyc}$	t_{DHW}	63	—	42	—	31	—	ns
29	MPU Address Access Time $t_{ACCA} = t_{cyc} - t_f - t_{DSR} - t_{AD}$	t_{ACCA}	348	—	203	—	144	—	ns
39	Write Data Setup Time $t_{DSW} = PW_{EH} - t_{DDW}$	t_{DSW}	185	—	102	—	60	—	ns
57	Address Valid to Data Three-State Time	t_{AVDZ}	—	10	—	10	—	10	ns

NOTES:
1. Input clocks with duty cycles other than 50% affect bus performance.
2. All timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted.

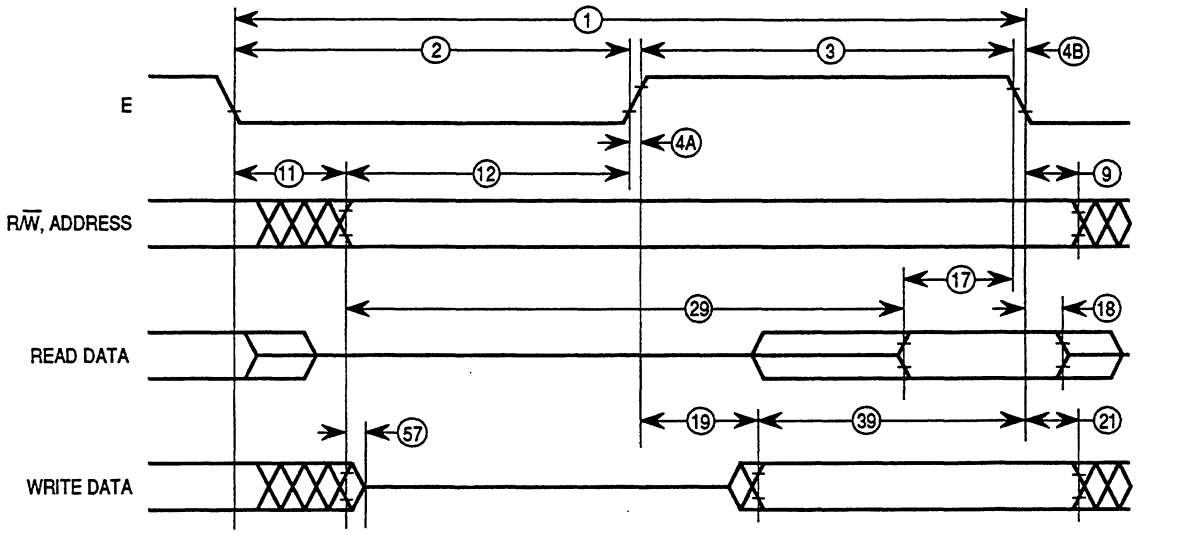


Figure A–9. Expansion Bus Timing

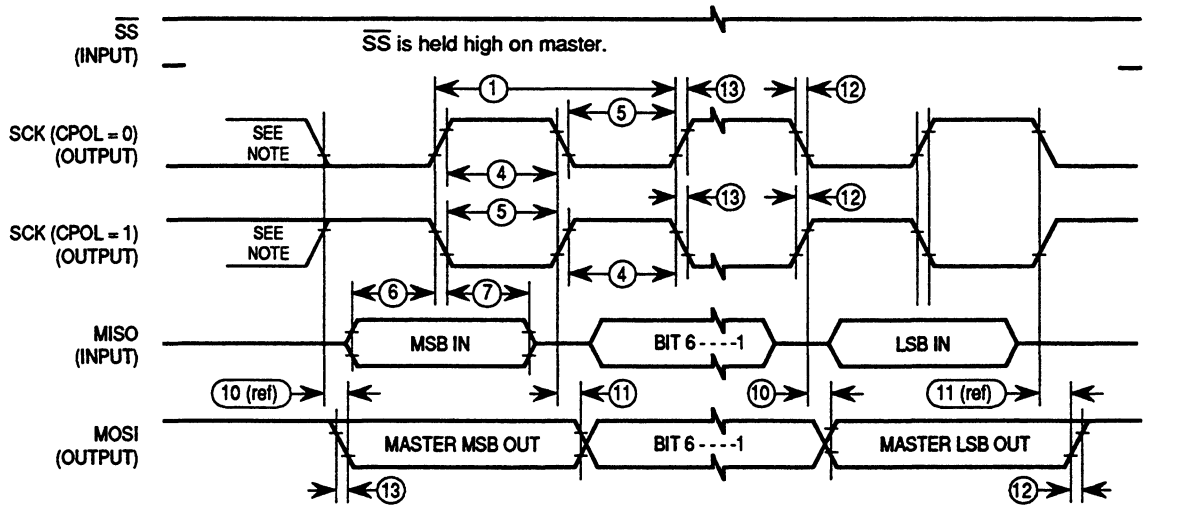
Table A–9. Serial Peripheral Interface Timing

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		4.0 MHz		Unit
			Min	Max	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	 dc dc	 0.5 4.0	f_{op} MHz
1	Cycle Time Master Slave	 $t_{cyc(m)}$ $t_{cyc(s)}$	 2.0 500	 — —	 2.0 333	 — —	 2.0 250	 — —	t_{cyc} ns
2	Enable Lead Time Master (Note 2) Slave	 $t_{lead(m)}$ $t_{lead(s)}$	 — 250	 — —	 — 240	 — —	 — 200	 — —	ns ns
3	Enable Lag Time Master (Note 2) Slave	 $t_{lag(m)}$ $t_{lag(s)}$	 — 250	 — —	 — 240	 — —	 — 200	 — —	ns ns
4	Clock (SCK) High Time Master Slave	 $t_{w(SCKH)m}$ $t_{w(SCKH)s}$	 340 190	 — —	 227 127	 — —	 130 85	 — —	ns ns
5	Clock (SCK) Low Time Master Slave	 $t_{w(SCKL)m}$ $t_{w(SCKL)s}$	 340 190	 — —	 227 127	 — —	 130 85	 — —	ns ns
6	Data Setup Time (Inputs) Master Slave	 $t_{su(m)}$ $t_{su(s)}$	 100 100	 — —	 100 100	 — —	 100 100	 — —	ns ns
7	Data Hold Time (Inputs) Master Slave	 $t_{h(m)}$ $t_{h(s)}$	 100 100	 — —	 100 100	 — —	 100 100	 — —	ns ns
8	Access Time (Time to Data Active from High-Impedance State) Slave	 t_a	 0	 120	 0	 120	 0	 120	ns
9	Disable Time (Hold Time to High-Impedance State) Slave	 t_{dis}	 —	 240	 —	 167	 —	 125	ns
10	Data Valid (After Enable Edge) (Note 3)	$t_{v(s)}$	—	240	—	167	—	125	ns
11	Data Hold Time (Outputs) (After Enable Edge)	t_{ho}	0	—	0	—	0	—	ns
12	Rise Time (20% V_{DD} to 70% V_{DD}, C_L = 200 pF) SPI Outputs (SCK, MOSI, and MISO) SPI Inputs (SCK, MOSI, MISO, and SS)	 t_{rm} t_{rs}	 — —	 100 2.0	 — —	 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 SS)	 t_{fm} t_{fs}	 — —	 100 2.0	 — —	 100 2.0	 — —	 100 2.0	 ns µs

NOTES:

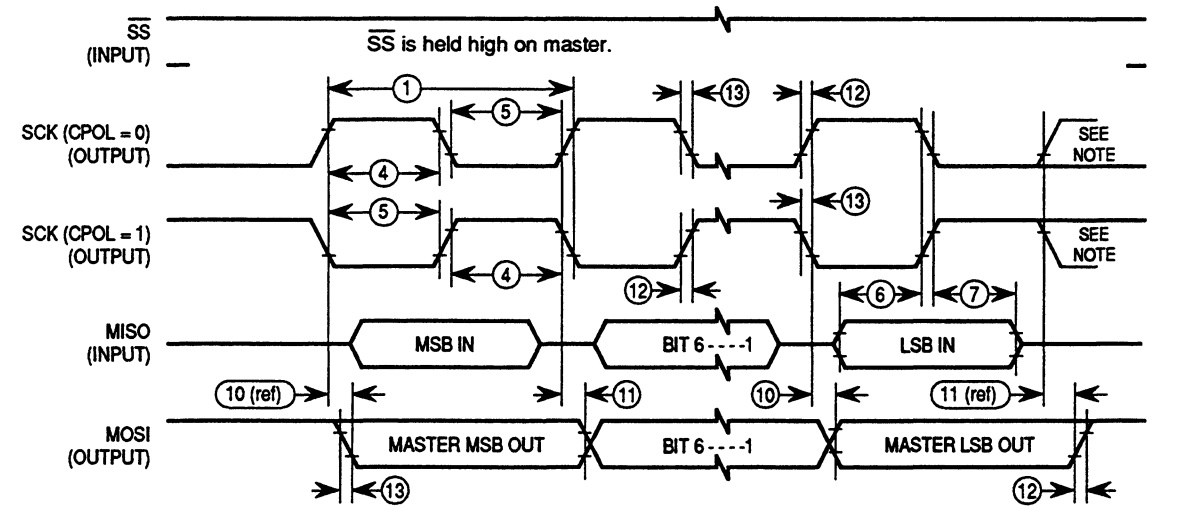
1. All timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted.
2. Signal production depends on software.
3. Assumes 200 pF load on all SPI pins.



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)

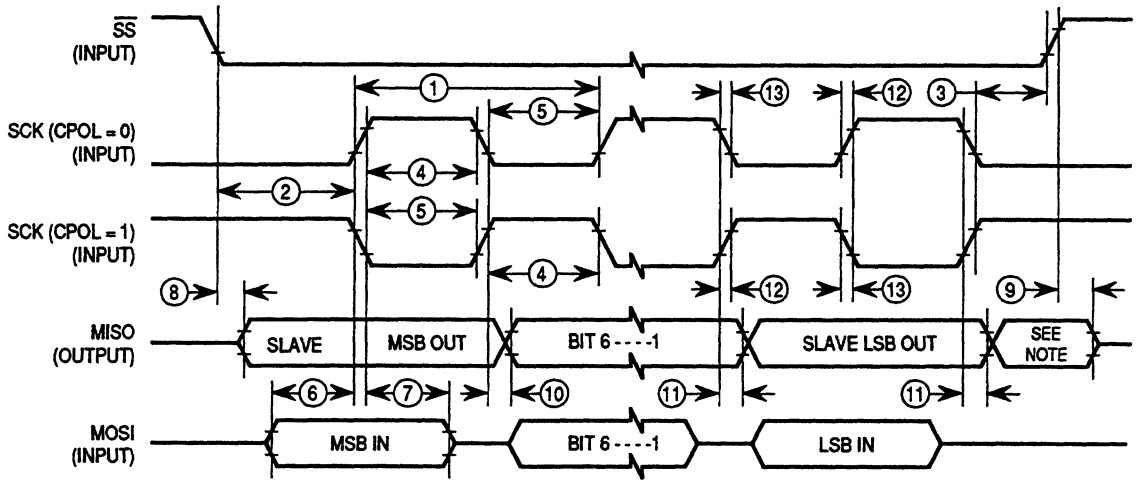


NOTE: This last clock edge is generated internally but is not seen at the SCK pin.

SPI MASTER CPHA1 TIM

b) SPI Master Timing (CPHA = 1)

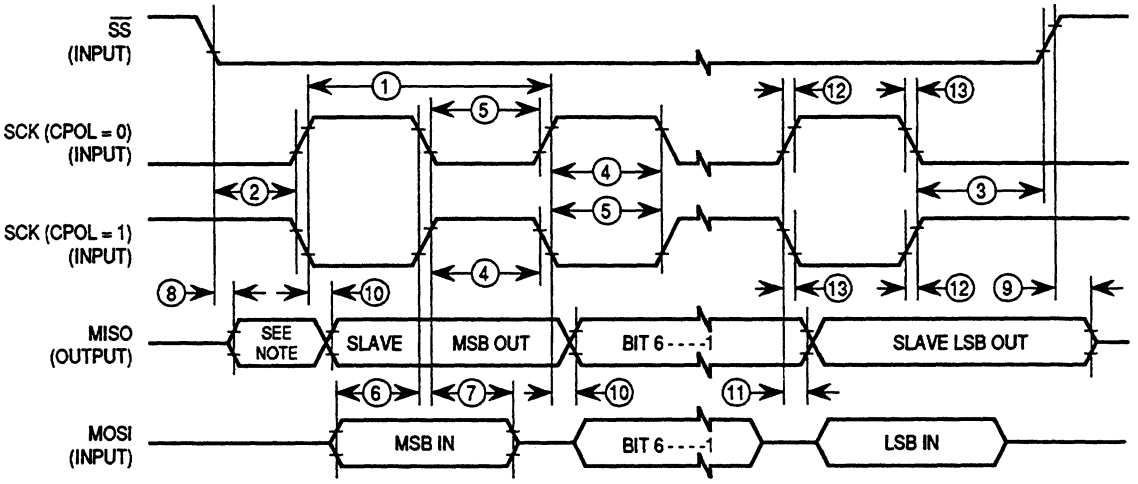
Figure A–10. SPI Timing Diagram (1 of 2)



NOTE: Not defined but normally MSB of character just received.

SPI SLAVE CPHA0 TIM

a) SPI Slave Timing (CPHA = 0)



NOTE: Not defined but normally LSB of character previously transmitted.

SPI SLAVE CPHA1 TIM

b) SPI Slave Timing (CPHA = 1)

Figure A–10. SPI Timing Diagram (2 of 2)

Table A–10. EEPROM Characteristics

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

Characteristic	Temperature Range			Unit
	–40 to 85° C	–40 to 105° C	–40 to 125° C	
Programming Time (Note 1) <1.0 MHz, RCO Enabled 1.0 to 2.0 MHz, RCO Disabled ≥ 2.0 MHz (or Anytime RCO Enabled)	10 20 10	15 Must use RCO 15	20 Must use RCO 20	ms
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

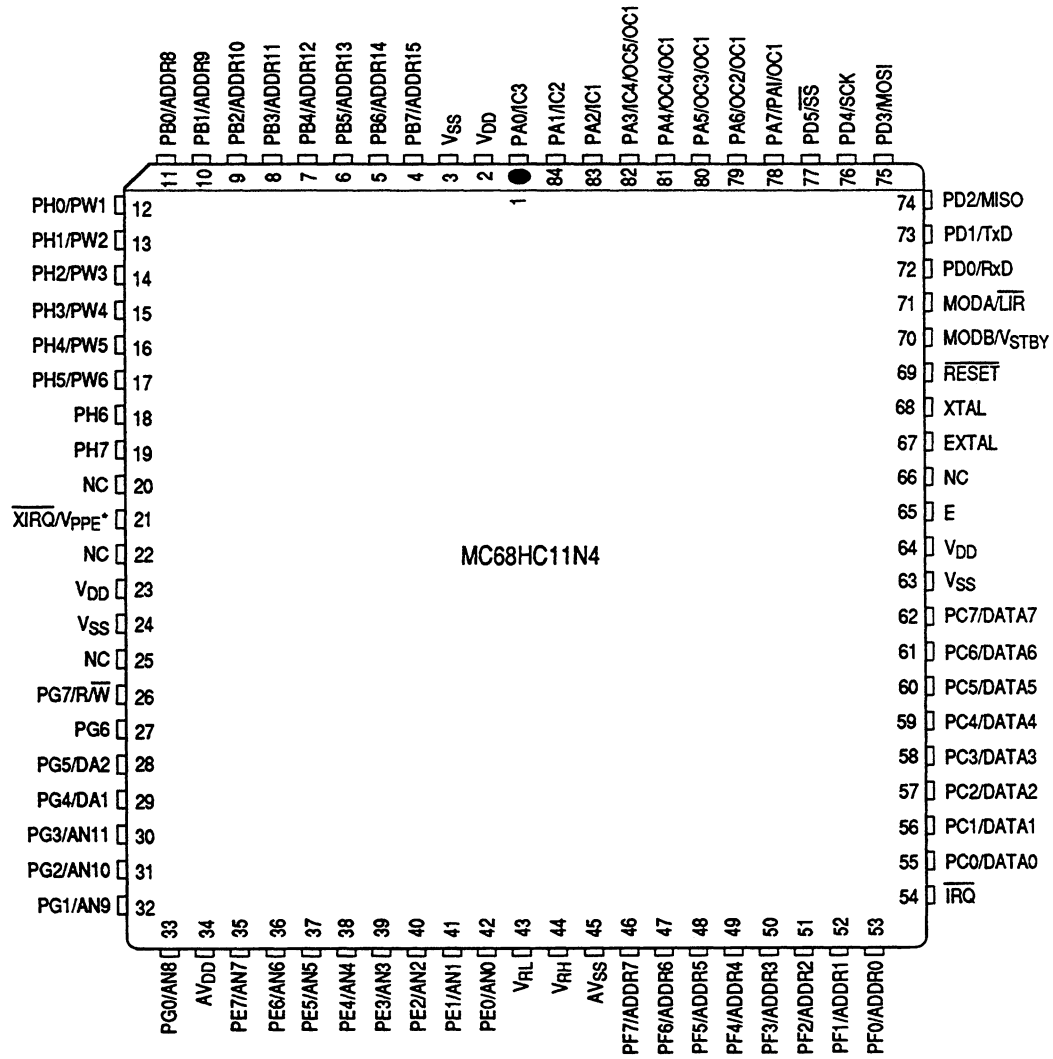
NOTES:

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.

APPENDIX B
MECHANICAL DATA AND ORDERING INFORMATION

B.1 Pin Assignments

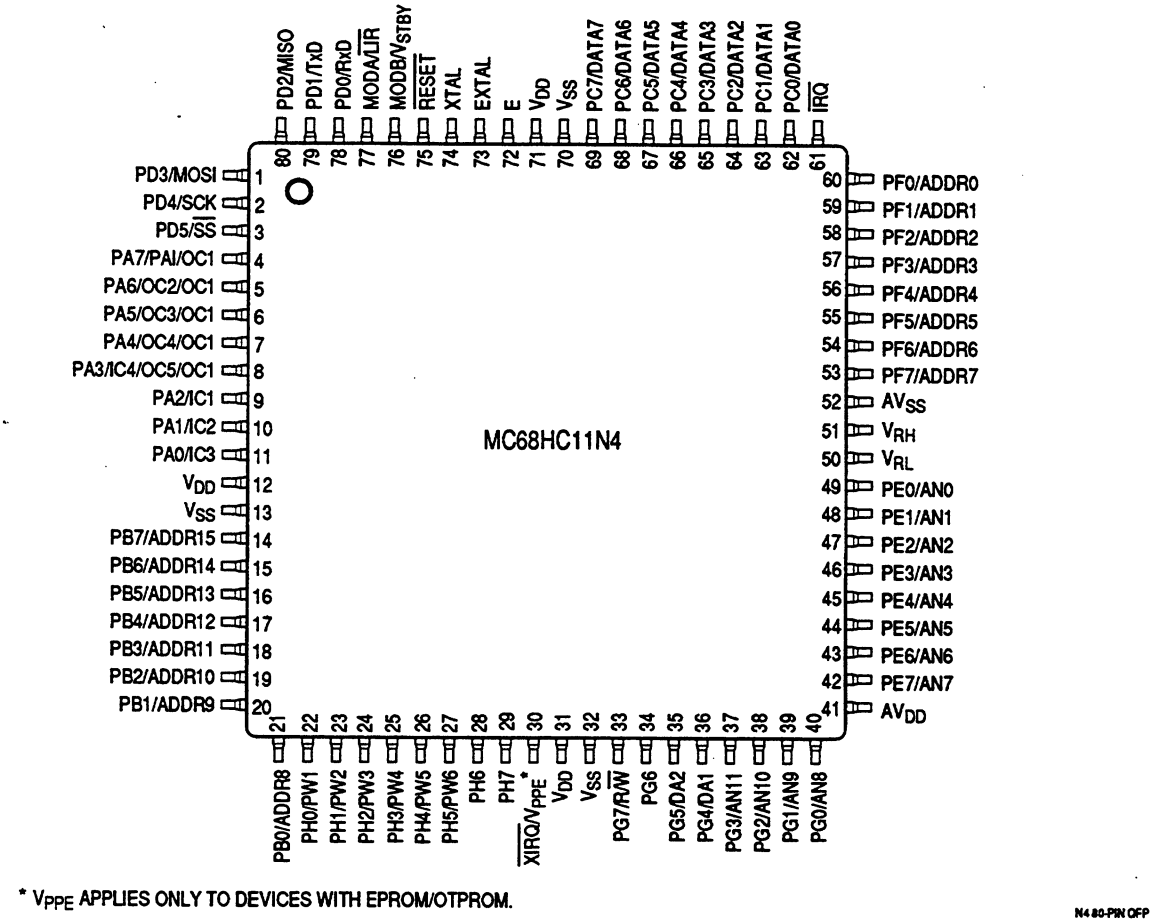
The MC68HC11N4 is available in the 80-pin plastic quad flat pack (QFP). The MC68HC711N4 is available in the 80-pin plastic QFP (OTPROM) or the 80-pin windowed ceramic QFP (EPROM). Refer to Tables B–1 and B–2 for ordering information. The 84-pin PLCC/Cerquad package option is not available at this time.



* V_{PPE} APPLIES ONLY TO DEVICES WITH EPROM/OTPROM.

N4 84-PIN PLCC

Figure B–1. M68HC11 N-Series 84-Pin PLCC/Cerquad



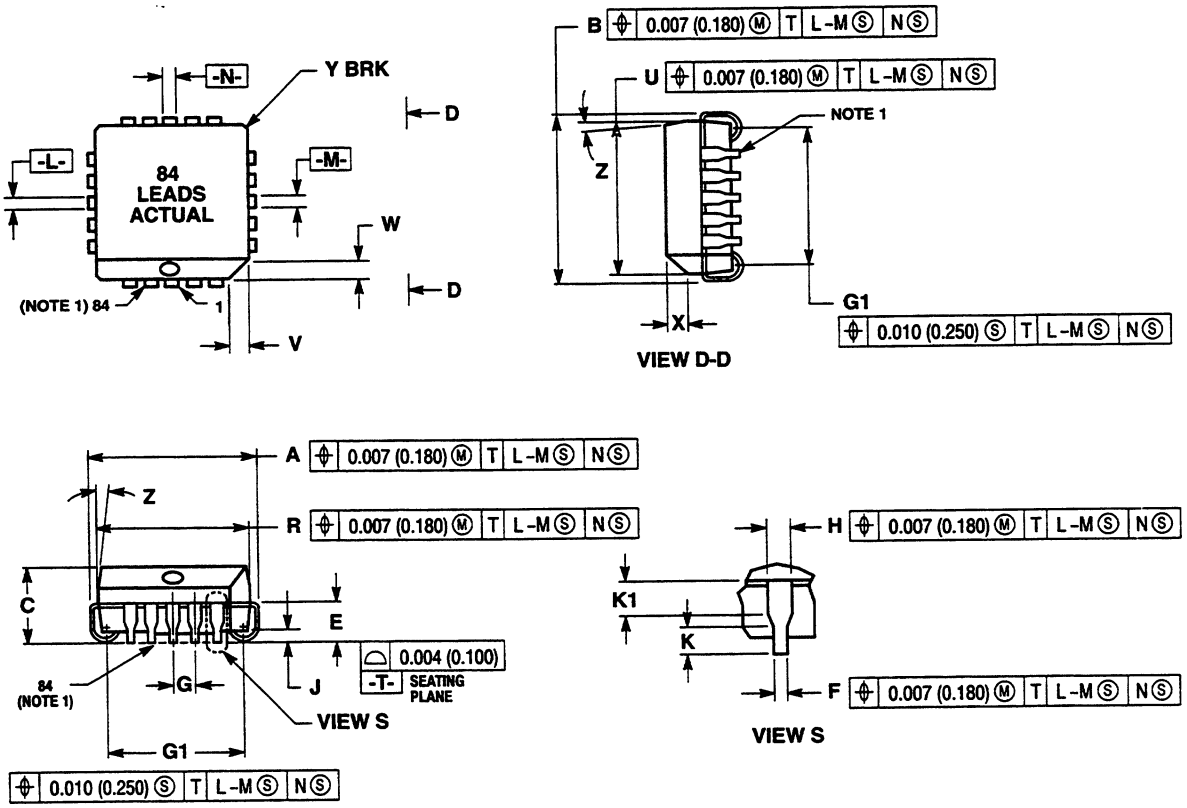
* V_{PPE} APPLIES ONLY TO DEVICES WITH EPROM/OTPROM.

N4 80-PIN QFP

Figure B–2. M68HC11 N-Series 80-Pin Quad Flat Pack

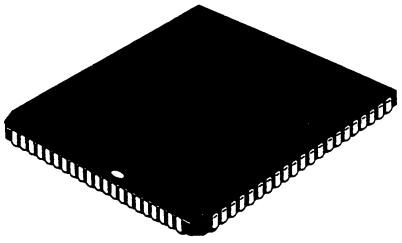
B.2 Package Dimensions

Figures B–3, B–4, and B–5 show the M68HC11 N-series MCUs in case outline.



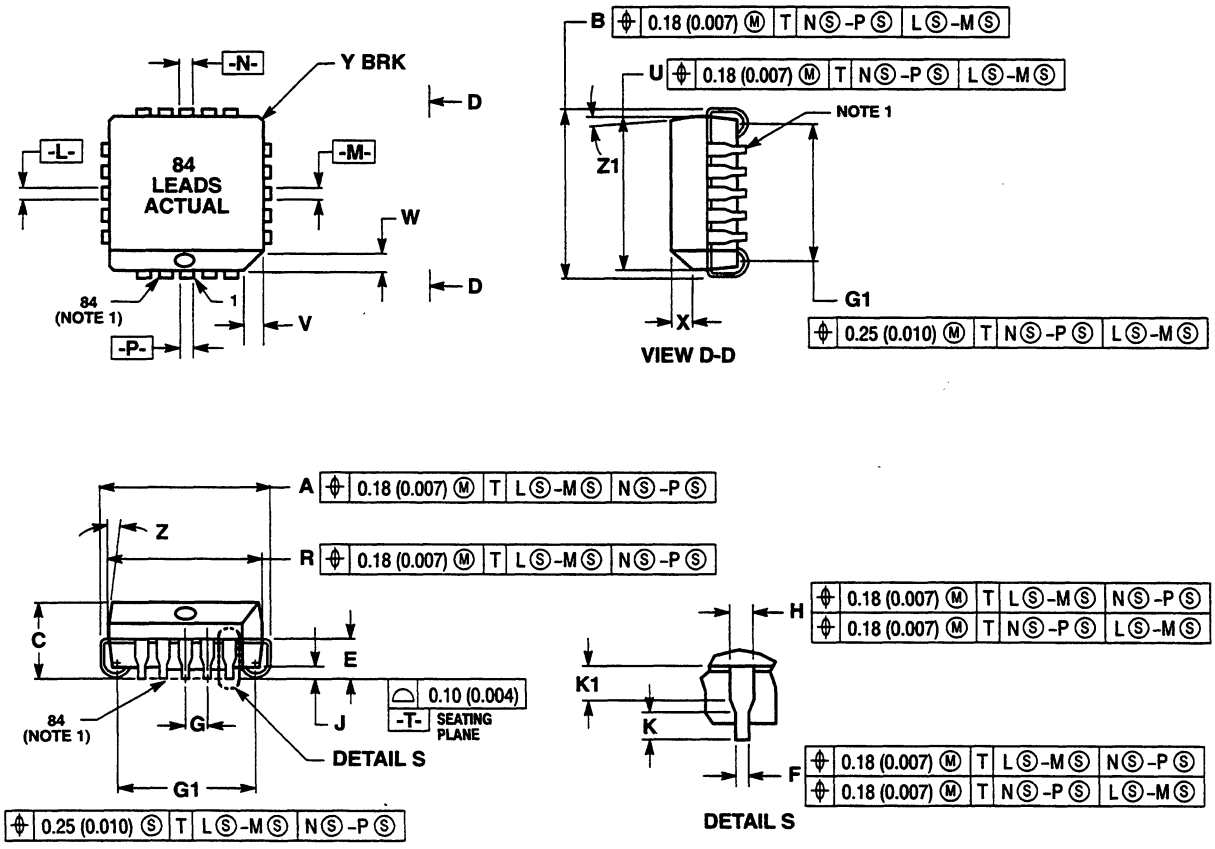
NOTES:
1. DUE TO SPACE LIMITATION, CASE 780-01 SHALL BE REPRESENTED BY A GENERAL (SMALLER) CASE OUTLINE DRAWING RATHER THAN SHOWING ALL 84 LEADS.
2. DATUMS -L-, -M-, AND -N- DETERMINED WHERE TOP OF LEAD SHOULDER EXITS PLASTIC BODY AT MOLD PARTING LINE.
3. DIM G1, TRUE POSITION TO BE MEASURED AT DATUM -T-, SEATING PLANE.
4. DIM R AND U DO NOT INCLUDE MOLD FLASH. ALLOWABLE MOLD FLASH IS .010 (.250) PER SIDE.
5. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
6. CONTROLLING DIMENSION: INCH.
7. THE PACKAGE TOP MAY BE SMALLER THAN THE PACKAGE BOTTOM BY UP TO .012 (.300). DIMENSIONS R AND U ARE DETERMINED AT THE OUTERMOST EXTREMES OF THE PLASTIC BODY EXCLUSIVE OF MOLD FLASH, TIE BAR BURRS, GATE BURRS AND INTERLEAD FLASH, BUT INCLUDING ANY MISMATCH BETWEEN THE TOP AND BOTOM OF THE PLASTIC BODY.
8. DIMENSION H DOES NOT INCLUDE DAMBAR PROTRUSION OR INTRUSION. THE DAMBAR PROTRUSION(S) SHALL NOT CAUSE THE H DIMENSION TO BE GREATER THAN .037 (.940). THE DAMBAR INTRUSION(S) SHALL NOT CAUSE THE H DIMENSION TO BE SMALLER THAN .025 (.635).

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	30.10	30.35	1.185	1.195
B	30.10	30.35	1.185	1.195
C	4.20	4.57	0.165	0.180
E	2.29	2.79	0.090	0.110
F	0.33	0.48	0.013	0.019
G	1.27 BSC		0.050 BSC	
H	0.66	0.81	0.026	0.032
J	0.51	—	0.020	—
K	0.64	—	0.025	—
R	29.21	29.36	1.150	1.156
U	29.21	29.36	1.150	1.156
V	1.07	1.21	0.042	0.048
W	1.07	1.21	0.042	0.048
X	1.07	1.42	0.042	0.056
Y	—	0.50	—	0.020
Z	2°	10°	2°	10°
G1	28.20	28.70	1.110	1.130
K1	1.02	—	0.040	—



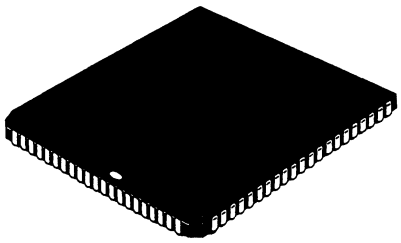
SCALE 1:1

Figure B–3. Case Outline #780-01



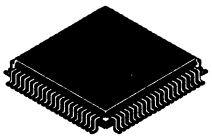
DIM	MILLIMETERS MIN	MAX	INCHES MIN	MAX
A	30.10	30.35	1.185	1.195
B	30.10	30.35	1.185	1.195
C	4.20	4.57	0.165	0.180
E	2.29	2.79	0.090	0.110
F	0.33	0.53	0.013	0.021
G	1.27 BSC		0.050 BSC	
H	0.66	0.81	0.026	0.032
J	0.51	—	0.020	—
K	0.64	—	0.025	—
R	29.21	29.36	1.150	1.156
U	29.21	29.36	1.150	1.156
V	1.07	1.21	0.042	0.048
W	1.07	1.21	0.042	0.048
X	1.07	1.42	0.042	0.056
Y	—	0.50	—	0.020
Z	2°	10°	2°	10°
G1	28.20	28.70	1.110	1.130
K1	1.02	—	0.040	—
Z1	2°	10°	2°	10°

NOTES:
1. DUE TO SPACE LIMITATION, CASE 780-01 SHALL BE REPRESENTED BY A GENERAL (SMALLER) CASE OUTLINE DRAWING RATHER THAN SHOWING ALL 84 LEADS.
2. DATUMS -L-, -M-, -N-, AND -P- DETERMINED WHERE TOP OF LEAD SHOULDER EXITS PACKAGE BODY AT GLASS PARTING LINE.
3. DIM G1, TRUE POSITION TO BE MEASURED AT DATUM -T-, SEATING PLANE.
4. DIM R AND U DO NOT INCLUDE GLASS PROTRUSION. ALLOWABLE GLASS PROTRUSION IS 0.25 (0.010) PER SIDE.
5. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
6. CONTROLLING DIMENSION: INCH.

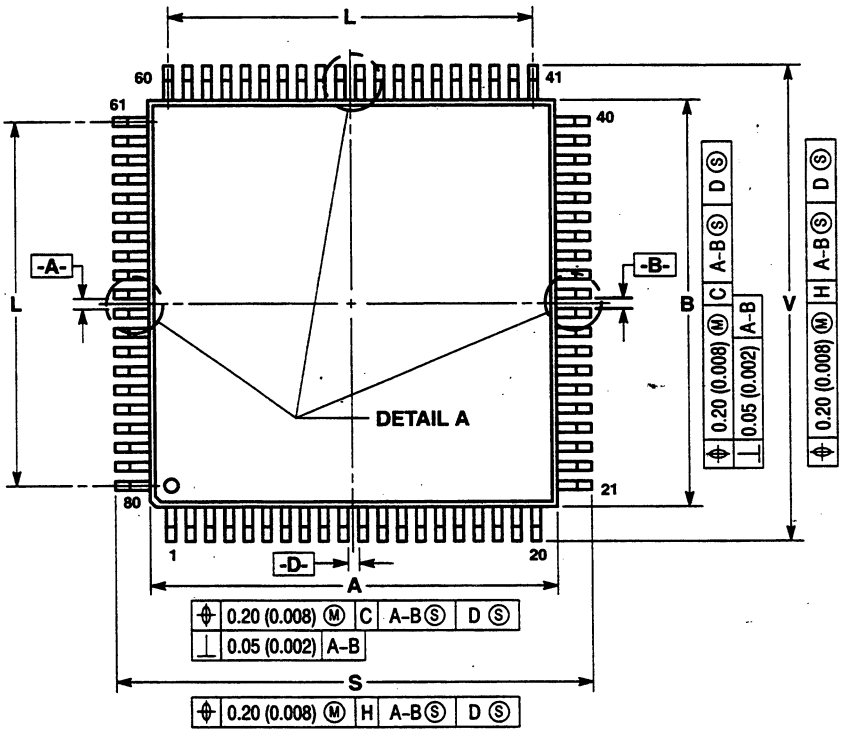


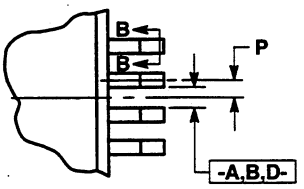
SCALE 1:1

Figure B–4. Case Outline #780A-01

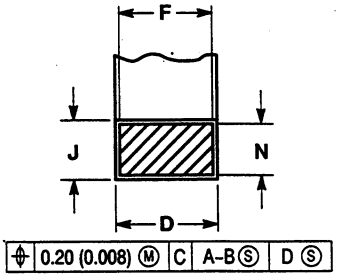


SCALE 1:1

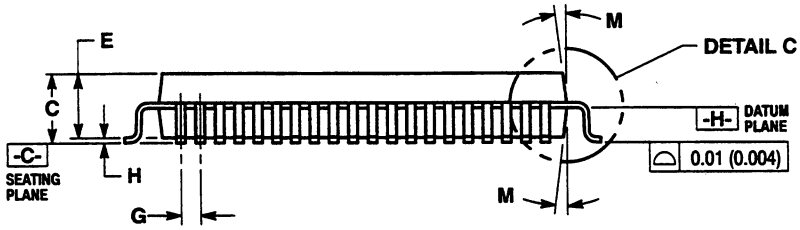


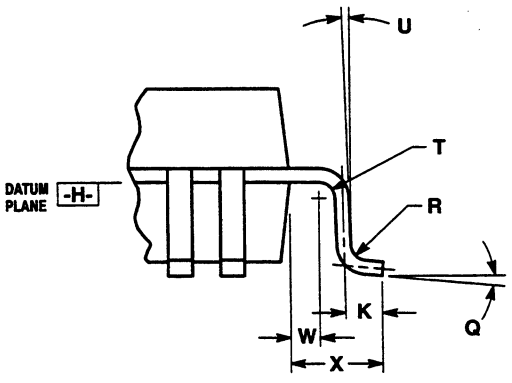


DETAIL A



SECTION B–B





DETAIL C

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

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	13.90	14.10	0.547	0.555
B	13.90	14.10	0.547	0.555
C	2.15	2.45	0.084	0.096
D	0.22	0.38	0.009	0.015
E	2.00	2.40	0.079	0.094
F	0.22	0.33	0.009	0.013
G	0.65 BSC		0.026 BSC	
H	—	0.25	—	0.010
J	0.13	0.23	0.005	0.009
K	0.65	0.95	0.026	0.037
L	12.35 BSC		0.486 BSC	
M	5°	10°	5°	10°
N	0.13	0.17	0.005	0.007
P	0.325 BSC		0.013 BSC	
Q	0°	7°	0°	7°
R	0.13	0.30	0.005	0.012
S	16.95	17.45	0.667	0.687
T	0.13	—	0.005	—
U	0°	—	0°	—
V	16.95	17.45	0.667	0.687
W	0.35	0.45	0.014	0.018
X	1.6 REF		0.06 REF	

Figure B–5. Case Outline #841B-01

B.3 Ordering Information

Use the information in Tables B–1 and B–2 to specify the appropriate device when placing an order.

Table B–1. Standard Device Ordering Information

Package	Temperature	CONFIG	Description	Frequency	MC Order Number
80-Pin Ceramic QFP (Windowed)	– 40° to + 85° C	$EF	EPROM	2 MHz	MC68HC711N4CFE2
				3 MHz	MC68HC711N4CFE3
				4 MHz	MC68HC711N4CFE4
	– 40° to + 105° C	$EF	EPROM	2 MHz	MC68HC711N4VFE2
				3 MHz	MC68HC711N4VFE3
				4 MHz	MC68HC711N4VFE4
84-Pin PLCC	– 40° to + 85° C	$EF	OTPROM	2 MHz	MC68HC711N4CFN2
				3 MHz	MC68HC711N4CFN3
				4 MHz	MC68HC711N4CFN4
	– 40° to + 105° C	$EF	OTPROM	2 MHz	MC68HC711N4VFN2
				3 MHz	MC68HC711N4VFN3
				4 MHz	MC68HC711N4VFN4
	– 40° to + 125° C	$EF	OTPROM	2 MHz	MC68HC711N4MFN2
				3 MHz	MC68HC711N4MFN3
				4 MHz	MC68HC711N4MFN4
84-Pin Ceramic Leaded Chip Carrier (Windowed)	– 40° to + 85° C	$EF	EPROM	2 MHz	MC68HC711N4CFS2
				3 MHz	MC68HC711N4CFS3
				4 MHz	MC68HC711N4CFS4
	– 40° to + 105° C	$EF	EPROM	2 MHz	MC68HC711N4VFS2
				3 MHz	MC68HC711N4VFS3
				4 MHz	MC68HC711N4VFS4
	– 40° to + 125° C	$EF	EPROM	2 MHz	MC68HC711N4MFS2
				3 MHz	MC68HC711N4MFS3
				4 MHz	MC68HC711N4MFS4

Shaded areas indicate package options not available at this time.

Table B–2. Custom ROM Device Ordering Information

Package	Temperature	Description	Frequency	MC Order Number
80-Pin Plastic QFP	– 40° to + 85° C	Custom ROM	2 MHz	MC68HC11N4CFU2
			3 MHz	MC68HC11N4CFU3
			4 MHz	MC68HC11N4CFU4
	– 40° to + 105° C	Custom ROM	2 MHz	MC68HC11N4VFU2
			3 MHz	MC68HC11N4VFU3
			4 MHz	MC68HC11N4VFU4
84-Pin PLCC	– 40° to + 85° C	Custom ROM	2 MHz	MC68HC11N4CFN2
			3 MHz	MC68HC11N4CFN3
			4 MHz	MC68HC11N4CFN4
	– 40° to + 105° C	Custom ROM	2 MHz	MC68HC11N4VFN2
			3 MHz	MC68HC11N4VFN3
			4 MHz	MC68HC11N4VFN4
	– 40° to + 125° C	Custom ROM	2 MHz	MC68HC11N4MFN2
			3 MHz	MC68HC11N4MFN3
			4 MHz	MC68HC11N4MFN4

Shaded areas indicate package options not available at this time.

APPENDIX C
DEVELOPMENT SUPPORT

C.1 M68HC11 Development Tools

The following provide a reference to development tools for the M68HC11 family of microcontrollers. Information provided is complete at the time of publication, but new systems and software are continually being developed. In addition, a growing number of third-party tools are available. Motorola publication *MCU Tool Box* (MCUTLBX/D) provides an up-to-date list of development tools.

At the time of this printing, M68HC11 N-series and M68HC11 M-series MCUs are available only in the 80-pin QFP package. The M68HC11KMNPEVS currently supports only the devices listed in the PLCC package.

New development support systems for the M68HC11 N series are currently being produced. A future version of the M68HC11KMNPEVS will support the QFP package. Contact your Motorola representative for more information.

Table C–1. M68HC11 Development Tools

Devices	Evaluation Boards	Evaluation Modules	Evaluation Systems/Kits	Programmer Boards
M68HC11 K series M68HC11 N series M68HC11 M series M68HC11 P series	—	—	M68HC11KMNPEVS	—

C.2 EVS — Evaluation System

The EVS is an economical tool for designing, debugging, and evaluating target systems based on the M68HC11.

- Monitor/debugger firmware
- One-line assembler/disassembler
- Host computer download capability
- Dual memory maps:
  — 64 Kbyte monitor map that includes 16 Kbytes of monitor EPROM
  — MC68HC711N4 user map that includes 64 Kbytes of emulation RAM
- OTPROM, EPROM, and EEPROM MCU programmer
- MCU extension I/O port for single-chip, expanded, and special-test operation modes
- RS-232C terminal and host I/O ports
- Logic analyzer connector

APPENDIX D
REGISTER INDEX

This appendix contains an alphabetical list of registers in the M68HC11 N-series MCU and their locations in this document. Bits in each register are listed beneath the register name.

— S —

— T —

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

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

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

MC68HC11N/D

1ATX31287-2 PRINTED IN USA 10/97 IMPERIAL LITHO 32472 1500 LITCSIC

Freescale Semiconductor, Inc.