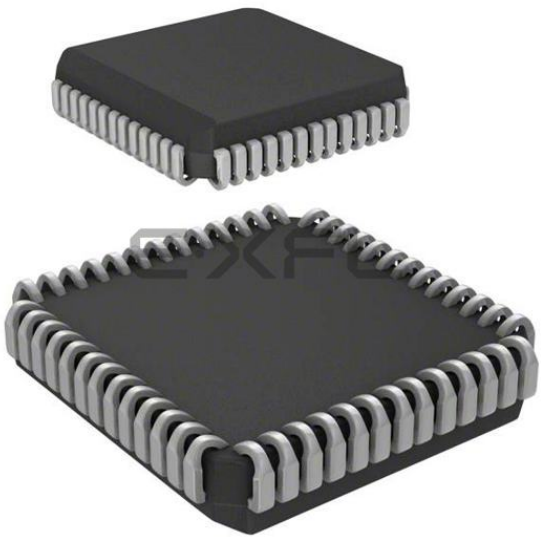


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	HC11
Core Size	8-Bit
Speed	2MHz
Connectivity	SCI, SPI
Peripherals	POR, WDT
Number of I/O	38
Program Memory Size	-
Program Memory Type	ROMless
EEPROM Size	512 x 8
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 5.5V
Data Converters	A/D 8x8b
Oscillator Type	Internal
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	52-LCC (J-Lead)
Supplier Device Package	52-PLCC (19.1x19.1)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc68l11e1fne2

Chapter 3
Analog-to-Digital (A/D) Converter

Chapter 4
Central Processor Unit (CPU)

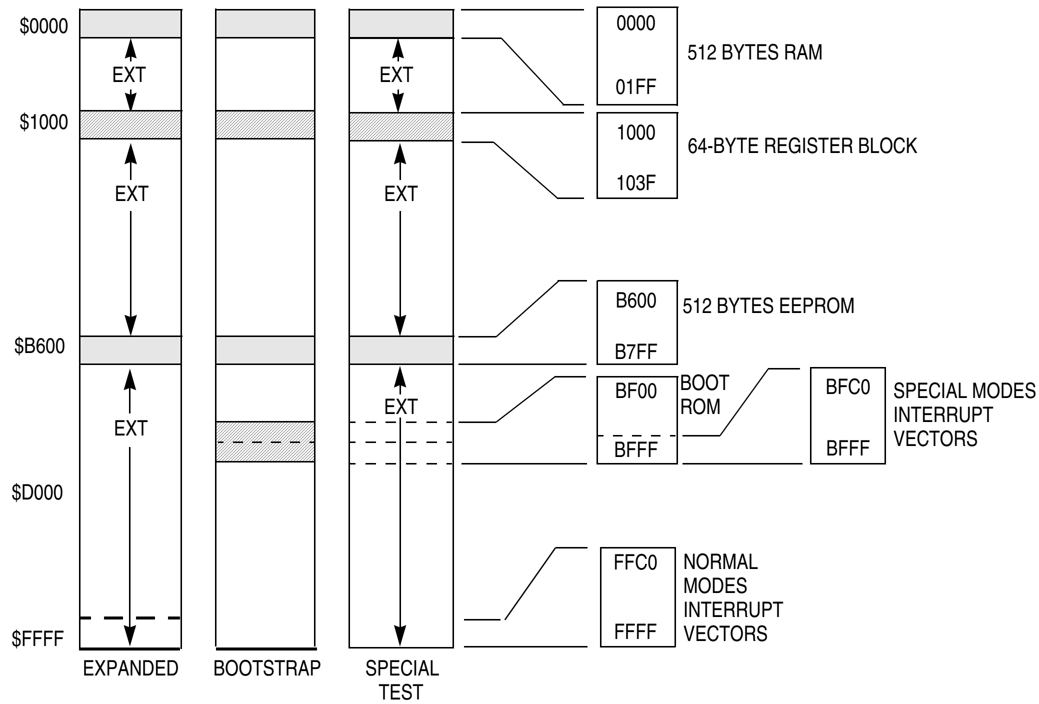


Figure 2-3. Memory Map for MC68HC11E1

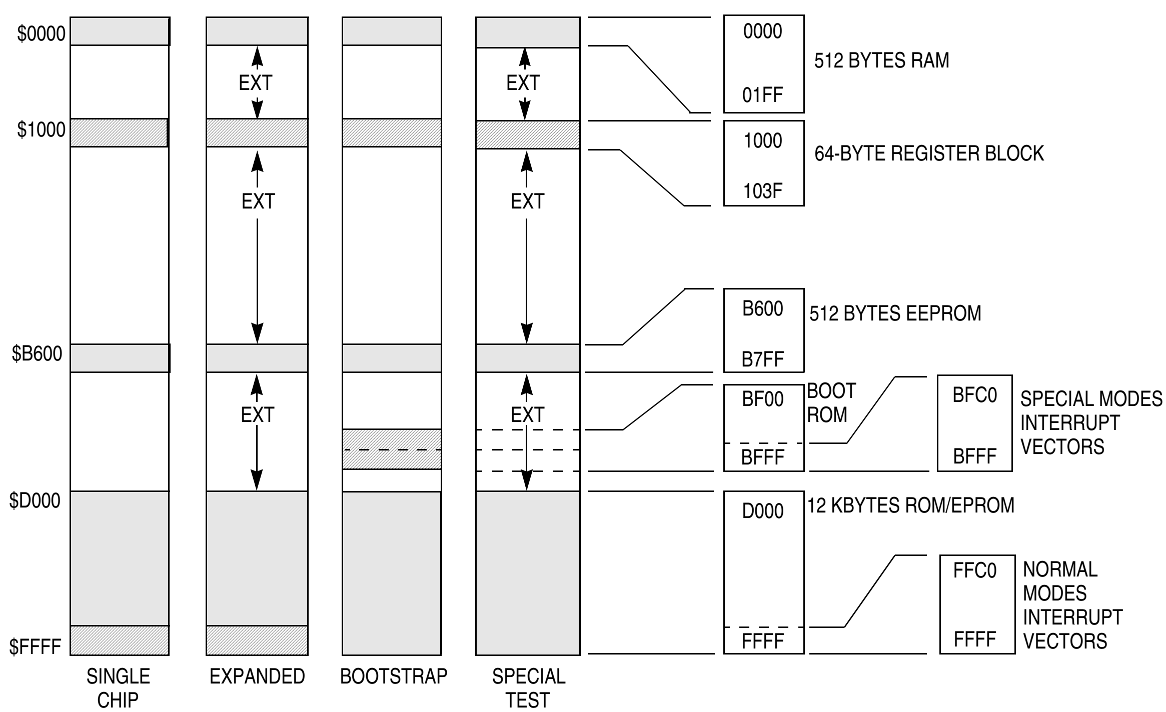


Figure 2-4. Memory Map for MC68HC(7)11E9

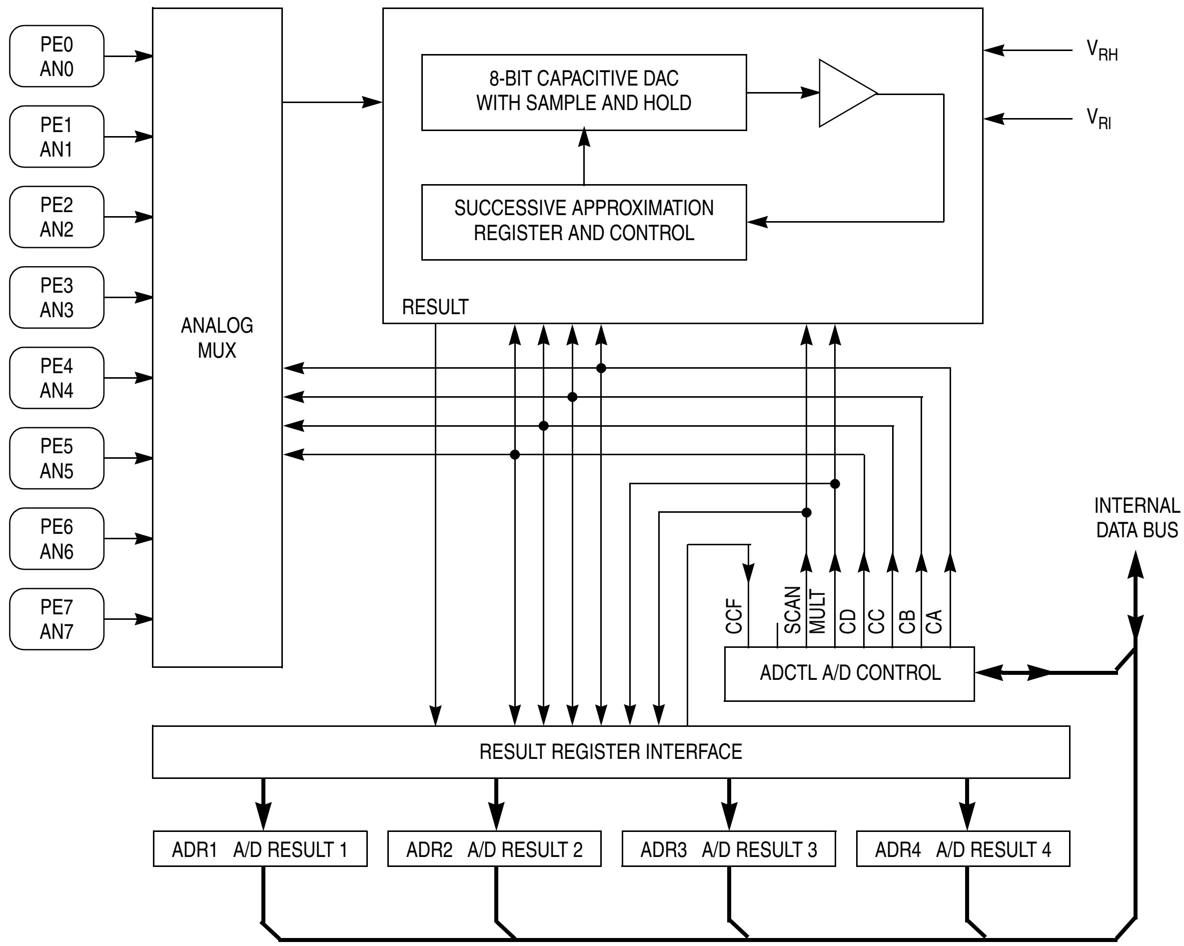


Figure 3-1. A/D Converter Block Diagram

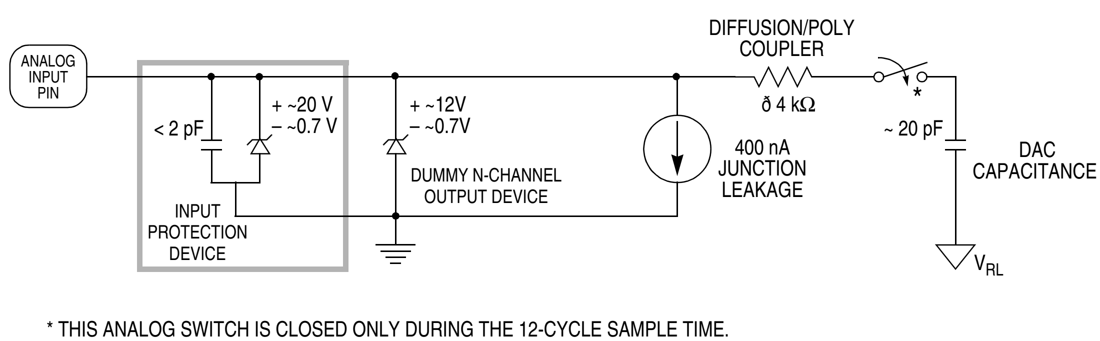


Figure 3-2. Electrical Model of an A/D Input Pin (Sample Mode)

At the end of the interrupt service routine, an return-from interrupt (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.

Certain instructions push and pull the A and B accumulators and the X and Y index registers and 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.

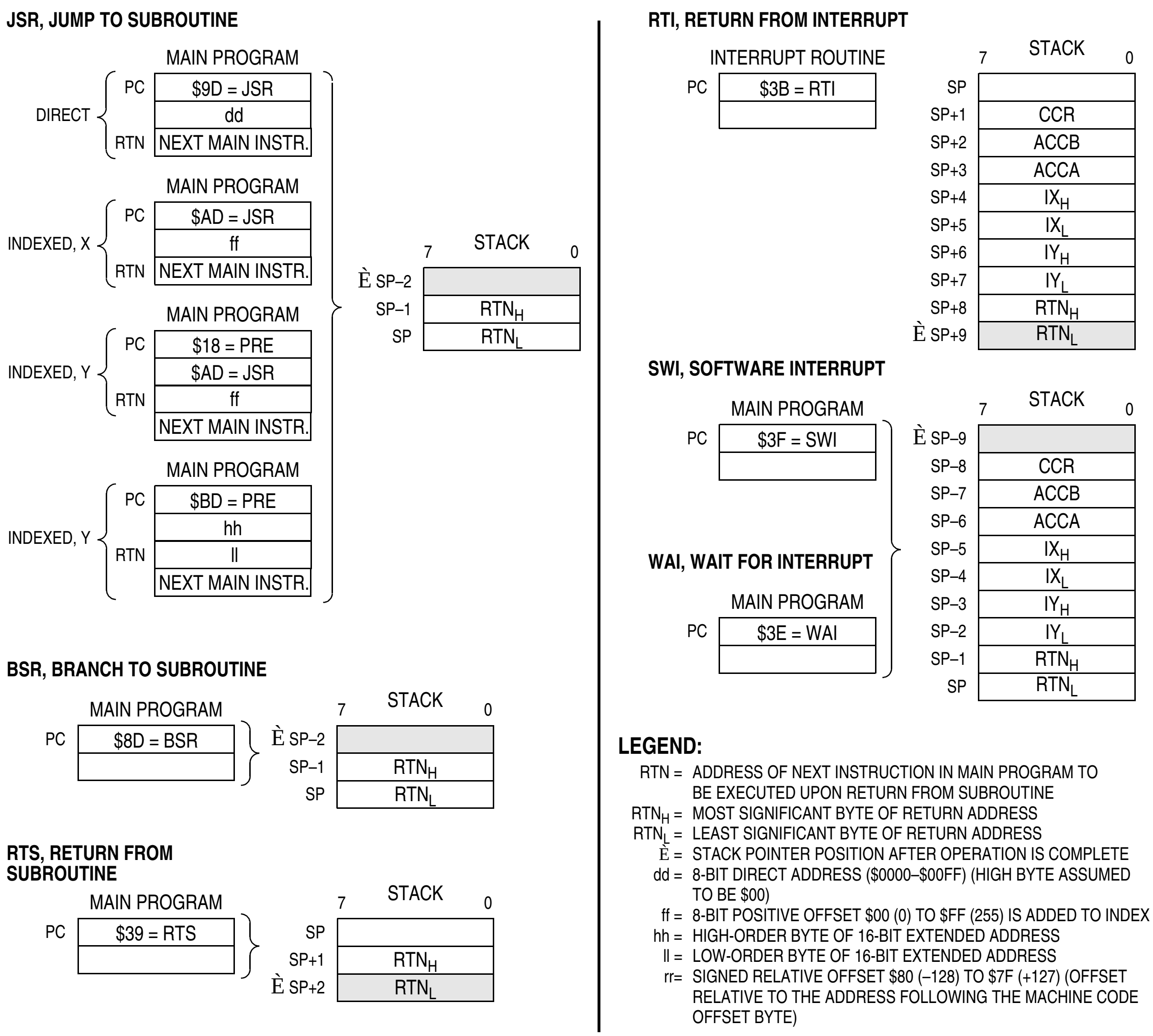


Figure 4-2. Stacking Operations

4.2.6.5 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 0 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 Chapter 5 Resets and Interrupts.

4.2.6.6 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.

4.2.6.7 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 0; 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.

4.2.6.8 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 is disabled by default.

4.3 Data Types

The M68HC11 CPU supports four data types:

1. Bit data
2. 8-bit and 16-bit signed and unsigned integers
3. 16-bit unsigned fractions
4. 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.

5.2.5 System Configuration Options Register

Address: $1039

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	ADPU	CSEL	IRQE(1)	DLY(1)	CME		CR1(1)	CR0(1)
Reset:	0	0	0	1	0	0	0	0

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

= Unimplemented

Figure 5-2. System Configuration Options Register (OPTION)

ADPU — Analog-to-Digital Converter Power-Up Bit

Refer to Chapter 3 Analog-to-Digital (A/D) Converter.

CSEL — Clock Select Bit

Refer to Chapter 3 Analog-to-Digital (A/D) Converter.

IRQE — Configure $\overline{\text{IRQ}}$ for Edge-Sensitive-Only Operation Bit

0 = $\overline{\text{IRQ}}$ is configured for level-sensitive operation.
1 = $\overline{\text{IRQ}}$ is configured for edge-sensitive-only operation.

DLY — Enable Oscillator Startup Delay Bit

Refer to Chapter 2 Operating Modes and On-Chip Memory and Chapter 3 Analog-to-Digital (A/D) Converter.

CME — Clock Monitor Enable Bit

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.

0 = Clock monitor circuit disabled
1 = Slow or stopped clocks cause reset

Bit 2 — Unimplemented

Always reads 0

CR[1:0] — COP Timer Rate Select Bit

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. See Table 5-1 for specific timeout settings.

5.3.9 Analog-to-Digital (A/D) Converter

The analog-to-digital (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 indeterminate.

5.3.10 System

The EEPROM programming controls are disabled, so the memory system is configured for normal read operation. PSEL[3:0] are initialized with the value %0110, causing the external $\overline{\text{IRQ}}$ pin to have the highest I-bit interrupt priority. The $\overline{\text{IRQ}}$ 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. MODA and MODB inputs select one of the four operating modes. After reset, writing SMOD and MDA in special modes causes the MCU to change operating modes. Refer to the description of HPRIO register in Chapter 2 Operating Modes and On-Chip Memory for a detailed description of SMOD and MDA. The DLY control bit is set to specify that an oscillator startup delay is imposed upon recovery from stop mode. The clock monitor system is disabled because CME is cleared.

5.4 Reset and Interrupt Priority

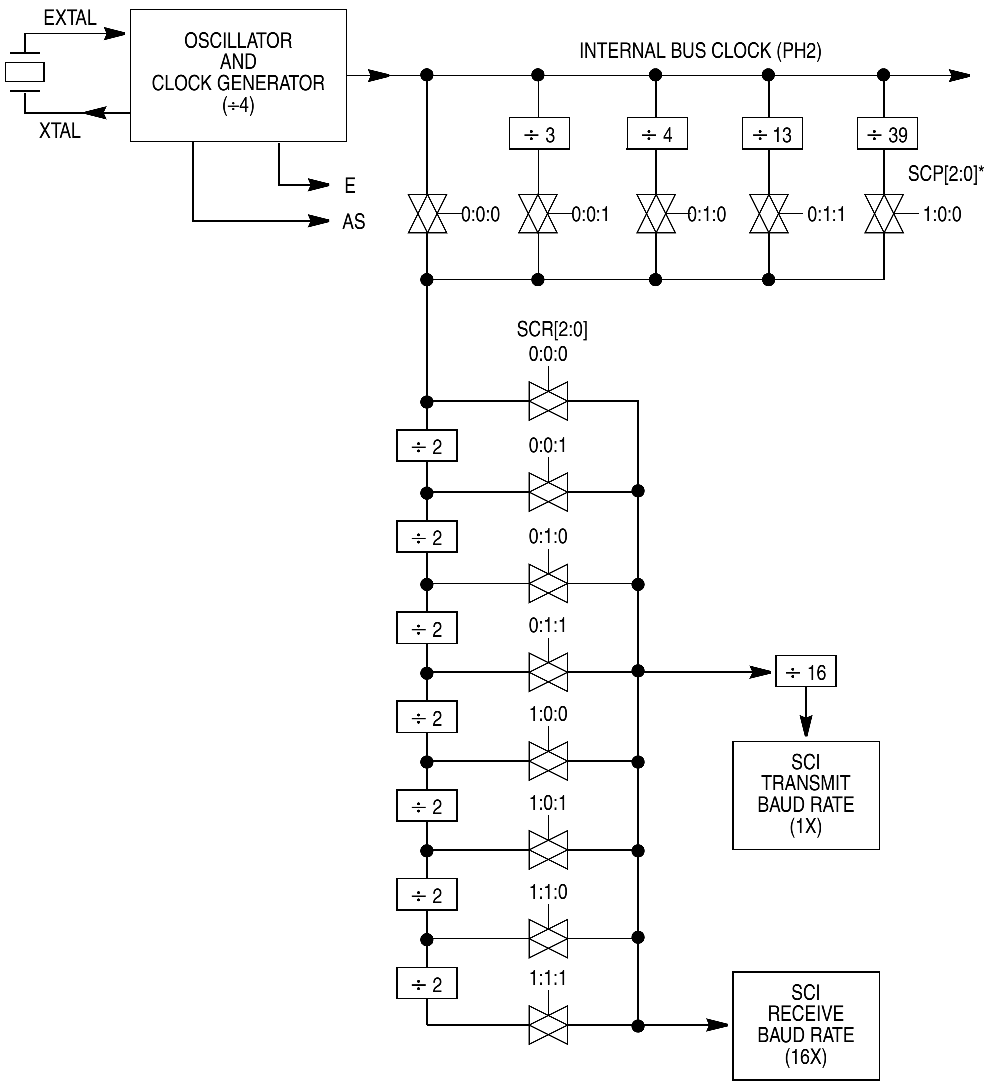
Resets and interrupts have a hardware priority that determines which reset or interrupt is serviced first when simultaneous requests occur. Any maskable interrupt can be given priority over other maskable interrupts.

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

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 this 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 5-7)



*SCP2 is present only on MC68HC(7)11E20.

Figure 7-9. MC68HC(7)11E20 SCI Baud Rate Generator Block Diagram

7.8 Status Flags and Interrupts

The SCI transmitter has two status flags. These status flags can be read by software (polled) to tell when the corresponding condition exists. Alternatively, a local interrupt enable bit can be set to enable each of these status conditions to generate interrupt requests when the corresponding condition is present. Status flags are automatically set by hardware logic conditions, but must be cleared by software, which 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.

MSTR — Master Mode Select Bit

It is customary to have an external pullup resistor on lines that are driven by open-drain devices.

0 = Slave mode
1 = Master mode

CPOL — Clock Polarity Bit

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 8.4 Clock Phase and Polarity Controls.

CPHA — Clock Phase Bit

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 8.4 Clock Phase and Polarity Controls.

SPR[1:0] — SPI Clock Rate Select Bits

These two bits select the SPI clock (SCK) rate when the device is configured as master. When the device is configured as slave, these bits have no effect. Refer to Table 8-1.

Table 8-1. SPI Clock Rates

SPR[1:0]	Divide E Clock By	Frequency at E = 1 MHz (Baud)	Frequency at E = 2 MHz (Baud)	Frequency at E = 3 MHz (Baud)	Frequency at E = 4 MHz (Baud)
0 0	2	500 kHz	1.0 MHz	1.5 MHz	2 MHz
0 1	4	250 kHz	500 kHz	750 kHz	1 MHz
1 0	16	62.5 kHz	125 kHz	187.5 kHz	250 kHz
1 1	32	31.3 kHz	62.5 kHz	93.8 kHz	125 kHz

8.7.2 Serial Peripheral Status Register

Address: $1029

	Bit 7	6	5	4	3	2	1	Bit 0
Read: / Write:	SPIF	WCOL		MODF				
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 8-4. Serial Peripheral Status Register (SPSR)

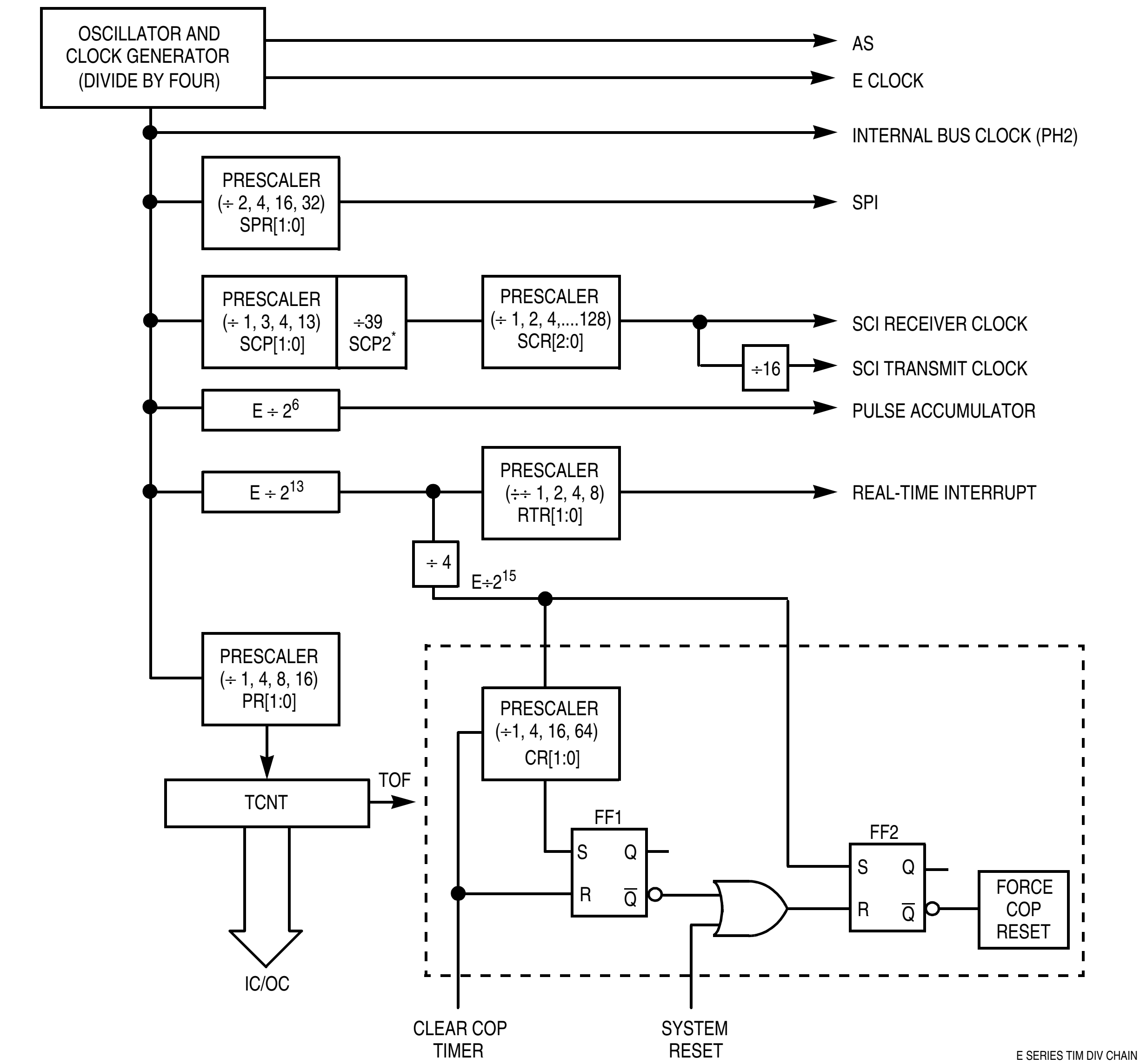
SPIF — SPI Interrupt Complete Flag

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 Bit

Clearing the WCOL bit is accomplished by reading the SPSR (with WCOL set) followed by an access of SPDR. Refer to 8.5.4 Slave Select and 8.6 SPI System Errors.

0 = No write collision
1 = Write collision



* SCP2 present on MC68HC(7)11E20 only

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	Other Rates
	1.0 MHz	2.0 MHz	3.0 MHz	(E)
	1000 ns	500 ns	333 ns	(1/E)
	Main Timer Count Rates			
0 0 1 count — overflow —	1000 ns 65.536 ms	500 ns 32.768 ms	333 ns 21.845 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	(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	(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	(E/16) ($E/2^{20}$)

9.2 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.

9.3 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.

9.4.5 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.

Register name: Timer Counter Register (High) Address: $100E

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Write:								
Reset:	0	0	0	0	0	0	0	0

Register name: Timer Counter Register (Low) Address: $100F

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 9-15. Timer Counter Register (TCNT)

9.4.6 Timer Control Register 1

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

Address: $1020

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

Figure 9-16. Timer Control Register 1 (TCTL1)

OM[2:5] — Output Mode Bits
OL[2:5] — Output Level Bits

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

10.6 Supply Currents and Power Dissipation

Characteristics[1]	Symbol	Min	Max	Unit
Run maximum total supply current[2] Single-chip mode 2 MHz 3 MHz Expanded multiplexed mode 2 MHz 3 MHz	I_{DD}	— — — —	15 27 27 35	mA
Wait maximum total supply current[2] (all peripheral functions shut down) Single-chip mode 2 MHz 3 MHz Expanded multiplexed mode 2 MHz 3 MHz	W_{IDD}	— — — —	6 15 10 20	mA
Stop maximum total supply current[2] Single-chip mode, no clocks –40°C to +85°C > +85°C to +105°C > +105°C to +125°C	S_{IDD}	— — —	25 50 100	µA
Maximum power dissipation Single-chip mode 2 MHz 3 MHz Expanded multiplexed mode 2 MHz 3 MHz	P_D	— — — —	85 150 150 195	mW

1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted
2. EXTAL is driven with a square wave, and
   t_{CYC}= 500 ns for 2 MHz rating
   t_{CYC}= 333 ns for 3 MHz rating
   $V_{IL} \leq 0.2$ V
   $V_{IH} \geq V_{DD} - 0.2$ V
   no dc loads

10.7 MC68L11E9/E20 DC Electrical Characteristics

Characteristics(1)	Symbol	Min	Max	Unit
Output voltage(2) I_{Load} = ±±10.0 µA All outputs except XTAL All outputs except XTAL, $\overline{\text{RESET}}$, and MODA	V_{OL}, V_{OH}	— V_{DD} –0.1	0.1 —	V
Output high voltage(2) I_{Load} = –0.5 mA, V_{DD} = 3.0 V I_{Load} = –0.8 mA, V_{DD} = 4.5 V All outputs except XTAL, $\overline{\text{RESET}}$, and MODA	V_{OH}	V_{DD} –0.8	—	V
Output low voltage I_{Load} = 1.6 mA, V_{DD} = 5.0 V I_{Load} = 1.0 mA, V_{DD} = 3.0 V All outputs except XTAL	V_{OL}	—	0.4	V
Input high voltage All inputs except $\overline{\text{RESET}}$ $\overline{\text{RESET}}$	V_{IH}	0.7 × V_{DD} 0.8 × V_{DD}	V_{DD} + 0.3 V_{DD} + 0.3	V
Input low voltage, all inputs	V_{IL}	V_{SS} –0.3	0.2 × V_{DD}	V
I/O ports, 3-state leakage V_{In} = V_{IH} or V_{IL} PA7, PA3, PC[7:0], PD[5:0], AS/STRA, MODA/$\overline{\text{LIR}}$, $\overline{\text{RESET}}$	I_{OZ}	—	±10	µA
Input leakage current(3) V_{In} = V_{DD} or V_{SS} PA[2:0], $\overline{\text{IRQ}}$, $\overline{\text{XIRQ}}$ MODB/V_{STBY} ($\overline{\text{XIRQ}}$ on EPROM-based devices)	I_{In}	— —	±1 ±10	µA
RAM standby voltage, power down	V_{SB}	2.0	V_{DD}	V
RAM standby current, power down	I_{SB}	—	10	µA
Input capacitance PA[2:0], PE[7:0], $\overline{\text{IRQ}}$, $\overline{\text{XIRQ}}$, EXTAL PA7, PA3, PC[7:0], PD[5:0], AS/STRA, MODA/$\overline{\text{LIR}}$, $\overline{\text{RESET}}$	I	— —	8 12	pF
Output load capacitance All outputs except PD[4:1] PD[4:1]	C_L	— —	90 100	pF

1. V_{DD} = 3.0 Vdc to 5.5 Vdc, V_{SS} = 0 Vdc, T_A = T_L to T_H, unless otherwise noted
2. V_{OH} specification for $\overline{\text{RESET}}$ and MODA is not applicable because they are open-drain pins. V_{OH} specification not applicable to ports C and D in wired-OR mode.
3. Refer to 10.13 Analog-to-Digital Converter Characteristics and 10.14 MC68L11E9/E20 Analog-to-Digital Converter Characteristics for leakage current for port E.

10.11 Peripheral Port Timing

Characteristic[1] [2]	Symbol	1.0 MHz		2.0 MHz		3.0 MHz		Unit
		Min	Max	Min	Max	Min	Max	
Frequency of operation E-clock frequency	f_o	dc	1.0	dc	2.0	dc	3.0	MHz
E-clock period	t_{CYC}	1000	—	500	—	333	—	ns
Peripheral data setup time MCU read of ports A, C, D, and E	t_{PDSU}	100	—	100	—	100	—	ns
Peripheral data hold time MCU read of ports A, C, D, and E	t_{PDH}	50	—	50	—	50	—	ns
Delay time, peripheral data write t_{PWD} = 1/4 t_{CYC}+ 100 ns MCU writes to port A MCU writes to ports B, C, and D	t_{PWD}	— —	200 350	— —	200 225	— —	200 183	ns
Port C input data setup time	t_{IS}	60	—	60	—	60	—	ns
Port C input data hold time	t_{IH}	100	—	100	—	100	—	ns
Delay time, E fall to STRB t_{DEB} = 1/4 t_{CYC}+ 100 ns	t_{DEB}	—	350	—	225	—	183	ns
Setup time, STRA asserted to E fall[3]	t_{AES}	0	—	0	—	0	—	ns
Delay time, STRA asserted to port C data output valid	t_{PCD}	—	100	—	100	—	100	ns
Hold time, STRA negated to port C data	t_{PCH}	10	—	10	—	10	—	ns
3-state hold time	t_{PCZ}	—	150	—	150	—	150	ns

1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = T_L to T_H, all timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted
2. Ports C and D timing is valid for active drive. (CWOM and DWOM bits are not set in PIOC and SPCR registers, respectively.)
3. If this setup time is met, STRB acknowledges in the next cycle. If it is not met, the response may be delayed one more cycle.

10.12 MC68L11E9/E20 Peripheral Port Timing

Characteristic[1] [2]	Symbol	1.0 MHz		2.0 MHz		Unit
		Min	Max	Min	Max	
Frequency of operation E-clock frequency	f_o	dc	1.0	dc	2.0	MHz
E-clock period	t_{CYC}	1000	—	500	—	ns
Peripheral data setup time MCU read of ports A, C, D, and E	t_{PDSU}	100	—	100	—	ns
Peripheral data hold time MCU read of ports A, C, D, and E	t_{PDH}	50	—	50	—	ns
Delay time, peripheral data write t_{PWD} = 1/4 t_{CYC}+ 150 ns MCU writes to port A MCU writes to ports B, C, and D	t_{PWD}	— —	250 400	— —	250 275	ns
Port C input data setup time	t_{IS}	60	—	60	—	ns
Port C input data hold time	t_{IH}	100	—	100	—	ns
Delay time, E fall to STRB t_{DEB} = 1/4 t_{CYC}+ 150 ns	t_{DEB}	—	400	—	275	ns
Setup time, STRA asserted to E fall[3]	t_{AES}	0	—	0	—	ns
Delay time, STRA asserted to port C data output valid	t_{PCD}	—	100	—	100	ns
Hold time, STRA negated to port C data	t_{PCH}	10	—	10	—	ns
3-state hold time	t_{PCZ}	—	150	—	150	ns

1. V_{DD} = 3.0 Vdc to 5.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, all timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted
2. Ports C and D timing is valid for active drive. (CWOM and DWOM bits are not set in PIOC and SPCR registers, respectively.)
3. If this setup time is met, STRB acknowledges in the next cycle. If it is not met, the response may be delayed one more cycle.

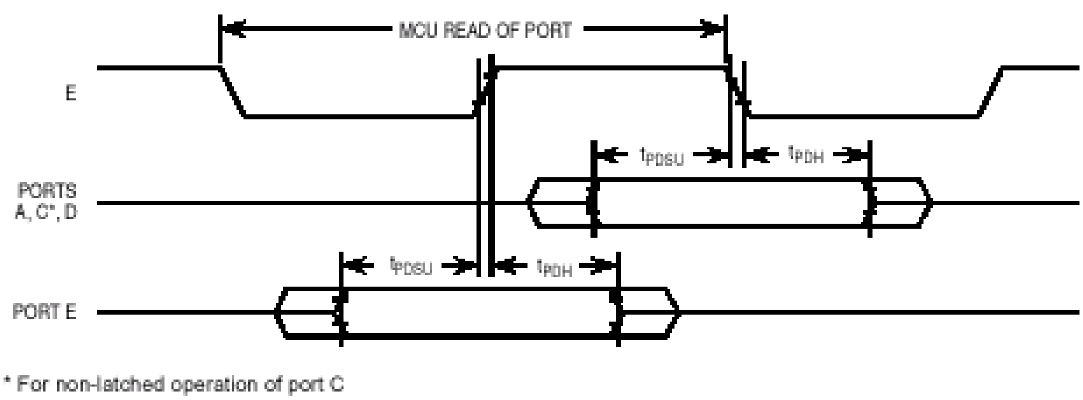


Figure 10-7. Port Read Timing Diagram

A.4 Modular Development System (MMDS11)

The M68MMDS11 modular development system (MMDS11) is an emulator system for developing embedded systems based on an M68HC11 microcontroller unit (MCU). The MMDS11 provides a bus state analyzer (BSA) and real-time memory windows. The unit's integrated development environment includes an editor, an assembler, user interface, and source-level debug. These features significantly reduce the time necessary to develop and debug an embedded MCU system. The unit's compact size requires a minimum of desk space.

The MMDS11 is one component of Freescale's modular approach to MCU-based product development. This modular approach allows easy configuration of the MMDS11 to fit a wide range of requirements. It also reduces development system cost by allowing the user to purchase only the modular components necessary to support the particular MCU derivative.

MMDS11 features include:

- Real-time, non-intrusive, in-circuit emulation at the MCU's operating frequency
- Real-time bus state analyzer
  - 8 K x 64 real-time trace buffer
  - Display of real-time trace data as raw data, disassembled instructions, raw data and disassembled instructions, or assembly-language source code
  - Four hardware triggers for commencing trace and to provide breakpoints
  - Nine triggering modes
  - As many as 8190 pre- or post-trigger points for trace data
  - 16 general-purpose logic clips, four of which can be used to trigger the bus state analyzer sequencer
  - 16-bit time tag or an optional 24-bit time tag that reduces the logic clips traced from 16 to eight
- Four data breakpoints (hardware breakpoints)
- Hardware instruction breakpoints over either the 64-Kbyte M68HC11 memory map or over a 1-Mbyte bank switched memory map
- 32 real-time variables, nine of which can be displayed in the variables window. These variables may be read or written while the MCU is running
- 32 bytes of real-time memory can be displayed in the memory window. This memory may be read or written while the MCU is running
- 64 Kbytes of fast emulation memory (SRAM)
- Current-limited target input/output connections
- Six software-selectable oscillator clock sources: five internally generated frequencies and an external frequency via a bus analyzer logic clip
- Command and response logging to MS-DOS® disk files to save session history
- SCRIPT command for automatic execution of a sequence of MMDS11 commands
- Assembly or C-language source-level debugging with global variable viewing
- Host/emulator communications speeds as high as 57,600 baud for quick program loading

® MS-DOS is a registered trademark of Microsoft Corporation.

Basic Bootstrap Mode

This section describes only basic functions of the bootstrap mode. Other functions of the bootstrap mode are described in detail in the remainder of this application note.

When an M68HC11 is reset in bootstrap mode, the reset vector is fetched from a small internal read-only memory (ROM) called the bootstrap ROM or boot ROM. The firmware program in this boot ROM then controls the bootloading process, in this manner:

- First, the on-chip SCI (serial communications interface) is initialized. The first character received ($FF) determines which of two possible baud rates should be used for the remaining characters in the download operation.
- Next, a binary program is received by the SCI system and is stored in RAM.
- Finally, a jump instruction is executed to pass control from the bootloader firmware to the user's loaded program.

Bootstrap mode is useful both at the component level and after the MCU has been embedded into a finished user system.

At the component level, Freescale uses bootstrap mode to control a monitored burn-in program for the on-chip electrically erasable programmable read-only memory (EEPROM). Units to be tested are loaded into special circuit boards that each hold many MCUS. These boards are then placed in burn-in ovens. Driver boards outside the ovens download an EEPROM exercise and diagnostic program to all MCUs in parallel. The MCUs under test independently exercise their internal EEPROM and monitor programming and erase operations. This technique could be utilized by an end user to load program information into the EPROM or EEPROM of an M68HC11 before it is installed into an end product. As in the burn-in setup, many M68HC11s can be gang programmed in parallel. This technique can also be used to program the EPROM of finished products after final assembly.

Freescale also uses bootstrap mode for programming target devices on the M68HC11 evaluation modules (EVM). Because bootstrap mode is a privileged mode like special test, the EEPROM-based configuration register (CONFIG) can be programmed using bootstrap mode on the EVM.

The greatest benefits from bootstrap mode are realized by designing the finished system so that bootstrap mode can be used after final assembly. The finished system need not be a single-chip mode application for the bootstrap mode to be useful because the expansion bus can be enabled after resetting the MCU in bootstrap mode. Allowing this capability requires almost no hardware or design cost and the addition of this capability is invisible in the end product until it is needed.

The ability to control the embedded processor through downloaded programs is achieved without the disassembly and chip-swapping usually associated with such control. This mode provides an easy way to load non-volatile memories such as EEPROM with calibration tables or to program the application firmware into a one-time programmable (OTP) MCU after final assembly.

Another powerful use of bootstrap mode in a finished assembly is for final test. Short programs can be downloaded to check parts of the system, including components and circuitry external to the embedded MCU. If any problems appear during product development, diagnostic programs can be downloaded to find the problems, and corrected routines can be downloaded and checked before incorporating them into the main application program.

Step 6

After the programming operation is complete, PCbug11 will display this message

Total bytes loaded: $xxxx

Total bytes programmed: $yyyy

- You should now remove the programming voltage from P4 connector pin 18, the XIRQ* pin.
- Each ORG directive in your assembly language source will cause a pair of these lines to be generated. For this operation, $yyyy will be incremented by the size of each block of code programmed into the EPROM of the MC68HC711E9.
- PCbug11 will display the above message whether or not the programming operation was successful. As a precaution, you should have PCbug11 verify your code.
- At the PCbug11 command prompt type: VERF C:\MYPROG\ISHERE.S19

Substitute the name of your program into the command above. Use a full path name if your program is not located in the same directory as PCbug11.

If the verify operation fails, a list of addresses which did not program correctly is displayed. Should this occur, you probably need to erase your part more completely. To do so, allow the MC68HC711E9 to sit for at least 45 minutes under an ultraviolet light source. Attempt the programming operation again. If you have purchased devices in plastic packages (one-time programmable parts), you will need to try again with a new, unprogrammed device.