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	3MHz
Connectivity	SCI, SPI
Peripherals	POR, PWM, WDT
Number of I/O	62
Program Memory Size	24KB (24K x 8)
Program Memory Type	OTP
EEPROM Size	640 x 8
RAM Size	768 x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 5.5V
Data Converters	A/D 8x8b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	80-QFP
Supplier Device Package	80-QFP (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc711k4cfue3

Revision History

To provide the most up-to-date information, the revision of our documents on the World Wide Web will be the most current. Your printed copy may be an earlier revision. To verify you have the latest information available, refer to:

http://www.motorola.com/semiconductors

The following revision history table summarizes changes contained in this document. For your convenience, the page number designators have been linked to the appropriate location.

Revision History

Date	Revision Level	Description	Page Number(s)
October, 2001	N/A	Original release	N/A

List of Tables

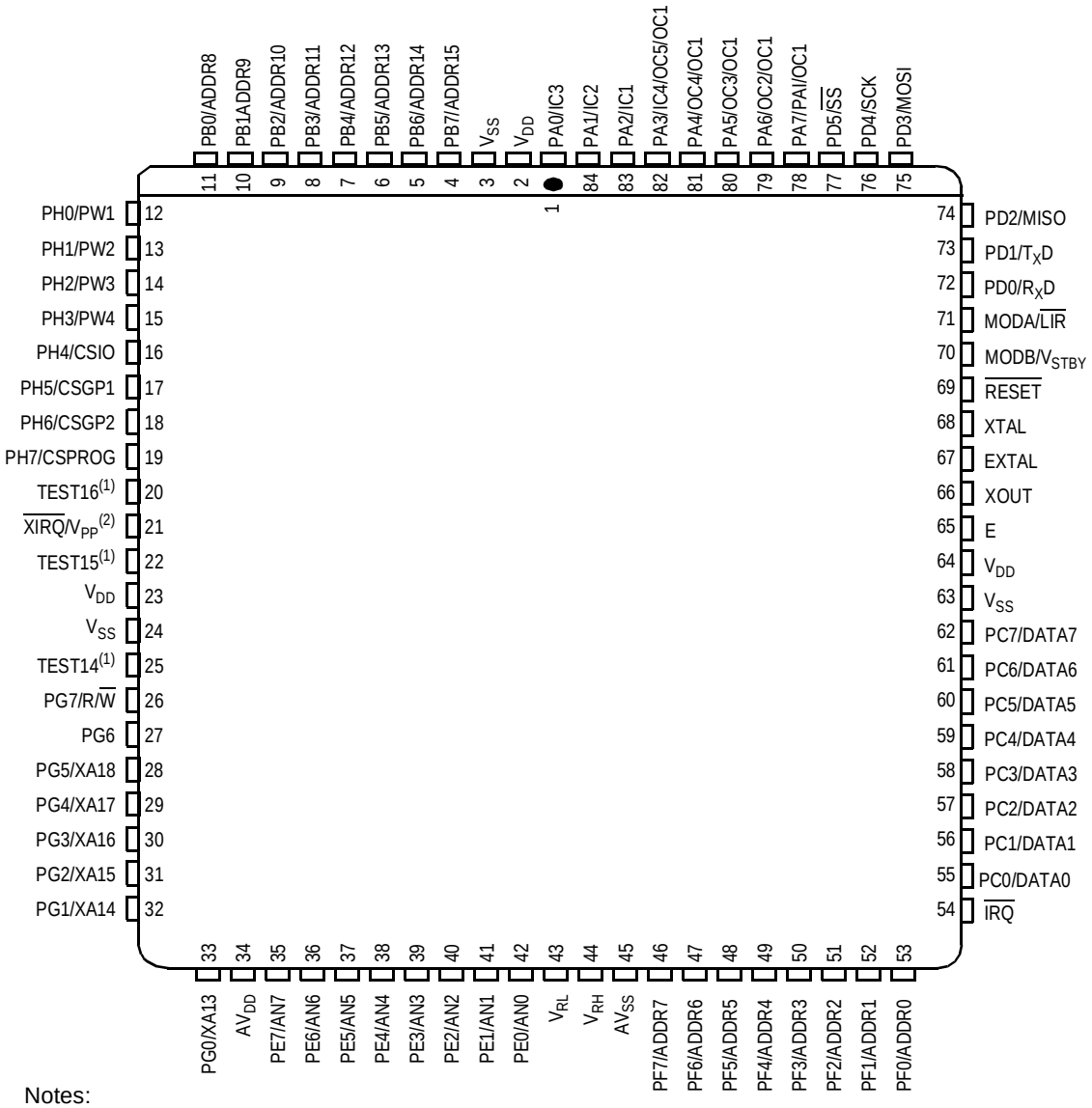
Section 2. Pin Description

2.1 Contents

2.2 Introduction

The M68HC11K Family is available in a variety of packages, as shown in **Table 1-1. M68HC11K Family Devices**. Most pins on this MCU serve two or more functions, as described in this section. Pin assignments for the various package types are shown in **Figure 2-1**, **Figure 2-2**, **Figure 2-3**, and **Figure 2-4**.

Pin Description



Notes:
1. Pins 20, 22, and 25 are used only during factory testing and should not be connected to external circuitry.
2. V_{PP} applies only to EPROM devices.

Figure 2-1. Pin Assignments for M68HC11K 84-Pin PLCC/J-Cerquad

3.3.4 Stack Pointer (SP)

The stack pointer holds the 16-bit address of the next free location in the M68HC11 CPU's automatic program stack. This stack is a data structure that grows downward from high memory to low memory. The stack can be located anywhere in the address space and can be any size up to the amount of memory available in the system. Most application programs initialize the SP at the beginning of an application program with a load stack (LDS) instruction. Thereafter, each time the CPU pushes a new byte onto the stack, it decrements the SP. To pull a byte from the stack, the CPU first increments the SP. **Figure 3-2** is a summary of SP operations.

A jump-to-subroutine (JSR) or branch-to-subroutine (BSR) instruction pushes the address of the instruction immediately after the JSR or BSR onto the stack, least significant byte first. The last instruction of the subroutine is a return-from-subroutine (RTS), which pulls the previously stored return address from the stack and loads it into the program counter. Execution then continues at this recovered return address.

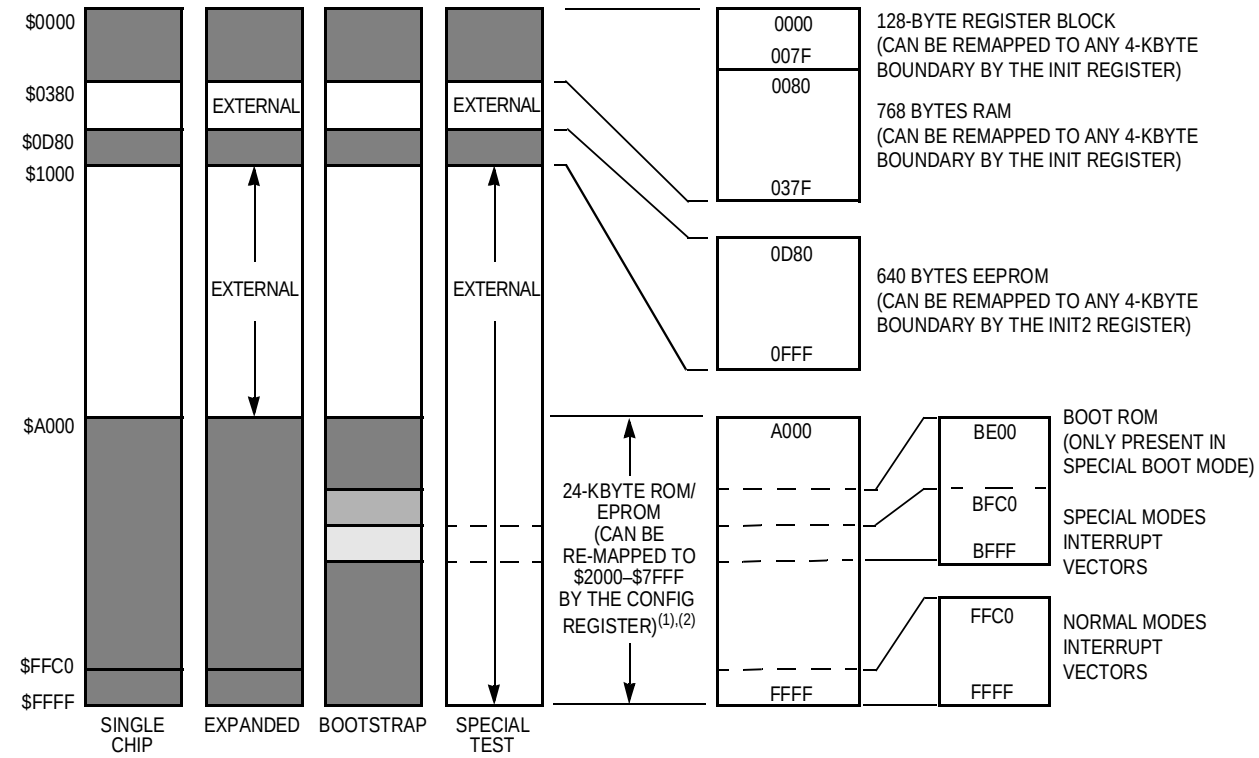
When the processor recognizes an interrupt, it finishes the current instruction, pushes the return address (the current value in the program counter) onto the stack, pushes all of the CPU registers onto the stack, and continues at the address specified by the vector for the interrupt. The interrupt service routine ends with a return-from-interrupt (RTI) instruction, which pulls the saved registers off the stack in reverse order. Program execution resumes at the return address with all register contents restored.

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

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$000A	Port E Data Register (PORTE) See page 143.	Read: Write:	PE7	PE6	PE5	PE4	PE3	PE2	PD1	PD0
		Reset:	Undefined after reset							
$000B	Timer Compare Force Register (CFORC) See page 201.	Read: Write:	FOC1	FOC2	FOC3	FOC4	FOC5	0	0	0
		Reset:	0	0	0	0	0	0	0	0
$000C	Output Compare 1 Mask Register (OC1M) See page 202.	Read: Write:	OC1M7	OC1M6	OC1M5	OC1M4	OC1M3	0	0	0
		Reset:	0	0	0	0	0	0	0	0
$000D	Output Compare 1 Data Register (OC1D) See page 202.	Read: Write:	OC1D7	OC1D6	OC1D5	OC1D4	OC1D3	0	0	0
		Reset:	0	0	0	0	0	0	0	0
$000E	Timer Counter Register High (TCNTH) See page 188.	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
$000F	Timer Counter Register Low (TCNTL) See page 188.	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
$0010	Timer Input Capture 1 Register High (TIC1H) See page 192.	Read: Write:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Reset:	Undefined after reset							
$0011	Timer Input Capture 1 Register Low (TIC1L) See page 192.	Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Reset:	Undefined after reset							
$0012	Timer Input Capture 2 Register High (TIC2H) See page 192.	Read: Write:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Reset:	Undefined after reset							
$0013	Timer Input Capture 2 Register Low (TIC2L) See page 192.	Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Reset:	Undefined after reset							
$0014	Timer Input Capture 3 Register High (TIC3H) See page 192.	Read: Write:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Reset:	Undefined after reset							

= Unimplemented R = Reserved U = Undefined

Figure 4-1. Register and Control Bit Assignments (Sheet 2 of 11)



Note 1.EPROM can be enabled in special test mode by setting the ROMON bit in the CONFIG register after reset.

Figure 4-3. M68HC11K4 Family Memory Map

4.6.4 Bootloader ROM

The bootloader program occupies 512 bytes of bootstrap ROM at addresses $BE00–$BFFF. It is active only in special modes when the RBOOT bit in the HPRIO register is set.

4.7 EPROM/OTPROM (M68HC711K4 and M68HC711KS2)

The M68HC711K4 devices include 24 Kbytes of on-chip EPROM (OTPROM in non-windowed packages). The M68HC711KS2 has 32 Kbytes of EPROM.

The two methods available to program the EPROM are:

- Downloading data through the serial communication interface (SCI) in bootstrap or special test mode
- Programming individual bytes from memory

Before proceeding with programming:

- Ensure that the CONFIG register ROMON bit is set.
- Ensure that the $\overline{\text{IRQ}}$ pin is pulled to a high level.
- Apply 12 volts to the $\overline{\text{XIRQ}}/V_{PP}$ pin.

Program the EPROM only at room temperature. Place an opaque label over the quartz window on windowed parts after programming.

4.7.1 Programming the EPROM with Downloaded Data

The MCU can download EPROM data through the SCI while in the special test or bootstrap modes. This can be done either with custom software, also downloaded through the SCI, or with a built-in utility program in bootstrap ROM. In either case, the 12-volt nominal programming voltage must be present on the $\overline{\text{XIRQ}}/V_{PP}$ pin.

To use the bootstrap ROM utility, download a 3-byte program consisting of a single jump instruction to $BF00, the starting address of the resident EPROM programming utility. The utility program sets the X and Y index

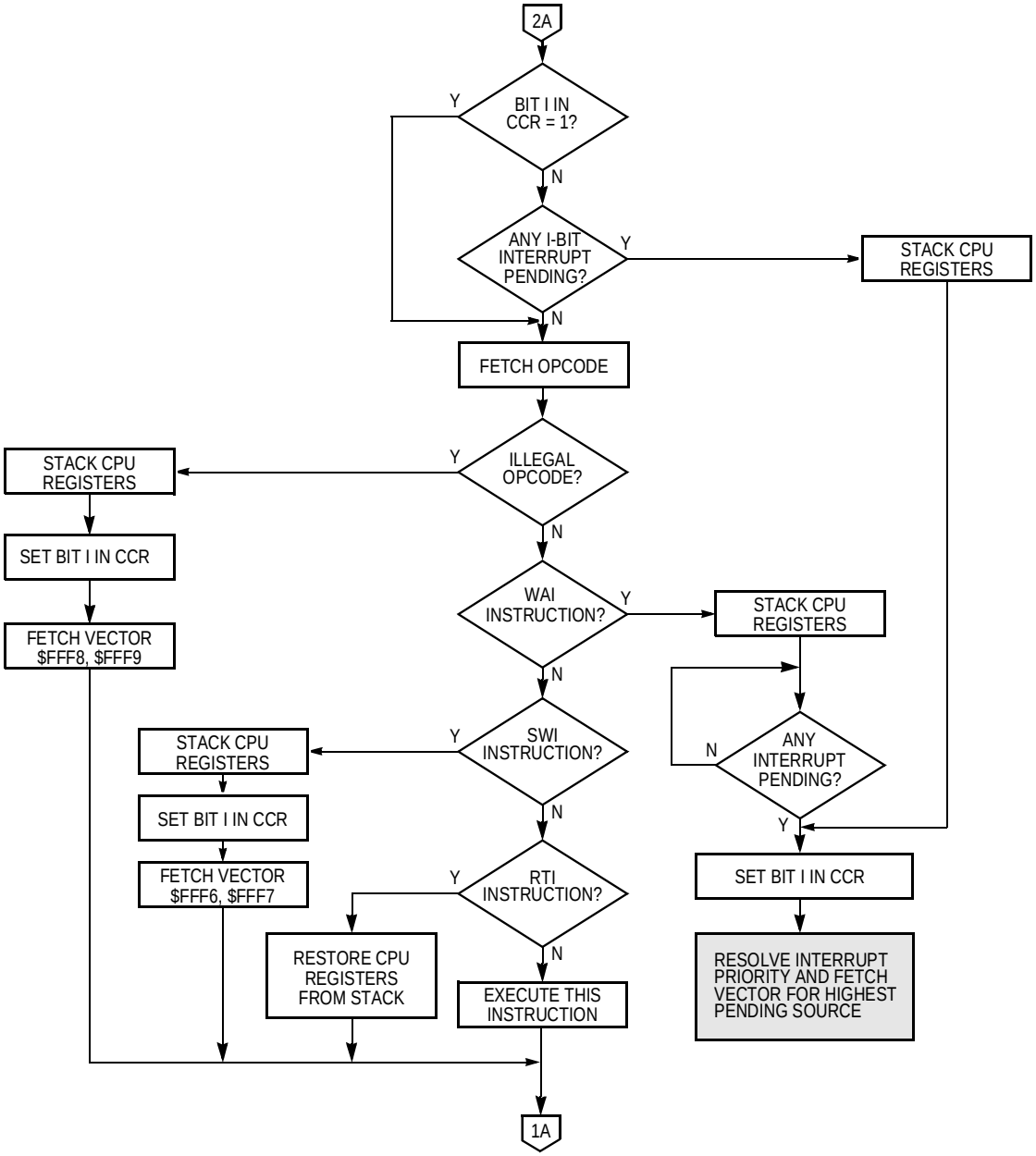


Figure 5-8. Processing Flow Out of Reset (Sheet 2 of 2)

6.10 Port H

The state of port H pin 7 (PH7) at reset is mode dependent. In single-chip or bootstrap modes, it is a high-impedance input; its data direction can be changed through DDRH. In expanded and special test modes PH7 is the program chip select line, $\overline{\text{CSPROG}}$ at reset, but can be reconfigured for GPIO (see **11.4 Chip Selects**).

Port H pins (PH[6:0]) reset to high-impedance inputs in any mode. Data direction can be changed through DDRH. Except for the M68HC11KS devices, bits 6:4 can serve as chip select lines in expanded and special test modes (see **11.4 Chip Selects**). Pins 3:0 can be configured as pulse-width modulator outputs (see **9.9 Pulse-Width Modulator (PWM)**) in any mode.

All eight port H pins have selectable internal pullup resistors (see **6.11 Internal Pullup Resistors**).

Address: $007C

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PH7(1)	PH6(1)	PH5(1)	PH4(1)	PH3	PH2	PH1	PH0
Reset:	0	0	0	0	0	0	0	0
Alternate Pin Function:	CSPROG	CSPG2	CSPG1	CSIO	PW4	PS3	PS2	PS1

1. Not available on KS devices

Figure 6-15. Port H Data Register (PORTH)

Address: $007D

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDH7(1)	DDH6(1)	DDH5(1)	DDH4(1)	DDH3	DDH2	DDH1	DDH0
Reset:	0	0	0	0	0	0	0	0

1. Not available on KS devices

Figure 6-16. Port H Data Direction Register (DDRH)

DDH[7:0] — Data Direction for Port H Bits

0 = Input

1 = Output

- The overrun error (OR) flag is set instead of the RDRF bit when the next byte is ready to be transferred from the receive shift register to the RDR and the RDR is already full. The data in the shift register is lost and the data that was already in RDR is not disturbed.
- The noise flag (NF) is set if there is noise on any of the received bits, including the start and stop bits. The data recovery circuit takes three samples of each bit and indicates noise if any set of three samples is not unanimous. NF is not set until the entire character is received and transferred to the RDR, when RDRF is set.
- The framing error (FE) flag is set when no stop bit is detected in the received data character. FE is set at the same time as RDRF. If the byte received causes both framing and overrun errors, the processor only recognizes the overrun error. The framing error flag inhibits further data transfer into the RDR until the flag is cleared.
- The parity error (PE) flag indicates that the parity bit of a received character does not match the parity calculated by hardware.
- The receiver active flag (RAF) is a read-only bit that is set during data reception and cleared when the line goes idle. This is the only flag cleared by hardware.

Figure 7-3 is a block diagram of the SCI receiver.

extra hardware. The SPI system can send data at up to one half of the E-clock rate when configured as master and the full E-clock rate when configured as a slave.

8.3 SPI Functional Description

The SPI is a 4-wire, full-duplex communication system. Characters are eight bits, transmitted most significant bit (MSB) first. One master device exchanges data with one or more slave devices. Each device selects its mode by writing either a 1 (master) or 0 (slave) to the MSTR bit in the serial peripheral control register (SPCR). As a master device transmits data to a slave device via the MOSI (master out slave in) line, the slave transmits data to the master via the MISO (master in slave out) line. The master produces a common synchronization clock signal and drives it on its SCK (serial clock) pin, which is configured as an output. The slave SCK pin is configured as an input to receive the clock. An external logic low signal is applied to the slave select pin ($\overline{SS}$) of each slave device for which a particular message is intended. Devices not selected ($\overline{SS}$ high) ignore the transmission.

Received characters are double-buffered. Serial input bits are fed into a shift register; when the last bit is received, the completed character is parallel-loaded to a read data buffer. This allows the next message to be received while the current message is being read. As long as the buffer is read before the next received character is ready to be transferred to the buffer, no overrun condition occurs.

Transmitted characters are not double-buffered, they are written directly to the output shift register. This means that new data for transmission cannot be written to the shift register until the previous transmission is complete. An attempt to write during data transmission will not go through; the transmission in progress will proceed undisturbed, and the MCU will set the write collision (WCOL) status bit in the serial peripheral status register (SPSR). After the last bit of a character is shifted out, the SPI transfer complete flag (SPIF) of the SPSR is set. This will also generate an interrupt if the SPIE (SPI interrupt enable) bit in the SPCR is set.

8.4 SPI Signal Descriptions

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

Every SPI output line must have its corresponding port D data direction register (DDRD) bit set. If this bit is clear, the line is disconnected from the SPI logic and becomes a general-purpose input line. SPI input lines are not affected by the data direction register.

8.4.1 Master In Slave Out (MISO)

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

8.4.2 Master Out Slave In (MOSI)

This unidirectional serial data line is an output in a master device and an input in a slave device.

8.4.3 Serial Clock (SCK)

The serial clock (SCK) synchronizes data movement both in and out of all devices. Master and slave devices exchange a byte of information simultaneously during a sequence of eight clock cycles. SCK is generated by the master device so its SCK pin functions as an output. Slave devices receive this signal through their SCK pins, which are configured as inputs.

The SPI clock rate select bits in the master device determine the SCK clock rate. These bits are SPR[1:0] in the serial peripheral control register (SPCR) and SPR2 in the system configuration options 2 register (OPT2). These bits have no effect in a slave device.

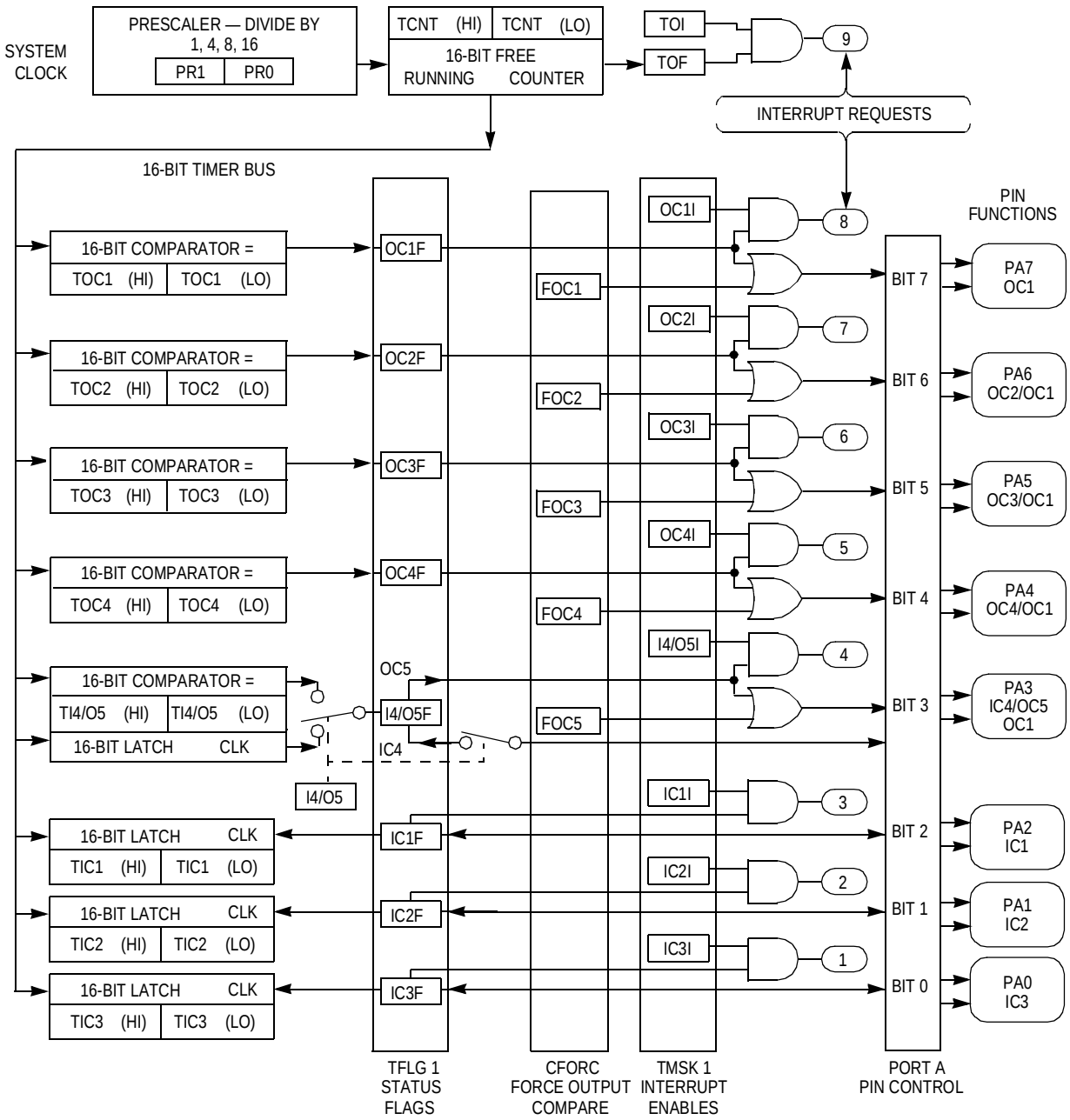


Figure 9-2. Capture/Compare Block Diagram

Timing System

Address: $005A

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	IOSA	IOSB	GP1SA	GP1SB	GP2SA	GP2SB	PCSA	PCSB
Reset:	0	0	0	0	0	0	0	0

Figure 11-14. Chip Select Clock Stretch Register (CSCSTR)

IOS[A:B] — CSIO Stretch Select Bits

GP1S[A:B] — CSGP1 Stretch Select Bits

GP2S[A:B] — CSGP2 Stretch Select Bits

PCS[A:B] — CSPROG Stretch Select Bits

Each of these pairs of bits contain the binary number of cycles of clock stretch, as shown in **Table 11-9**.

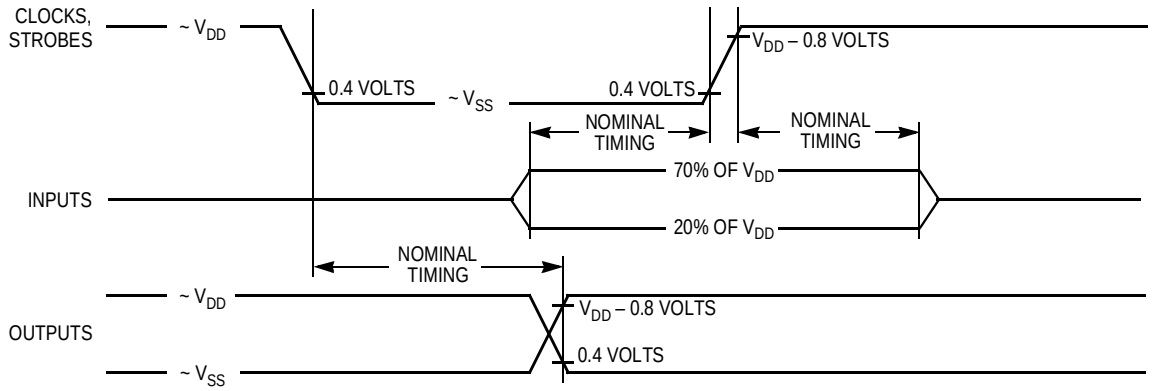
Table 11-9. CSCSTR Bits Versus Clock Cycles

Bit [A:B]	Clock Stretch
0 0	None
0 1	1 cycle
1 0	2 cycles
1 1	3 cycles

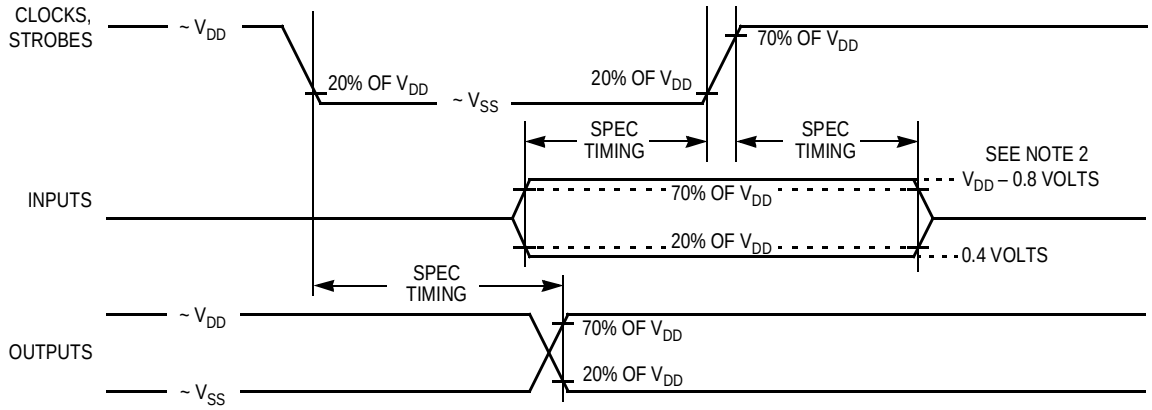
11.5 Memory Expansion Examples

The first example, shown in **Figure 11-15** contains a system with 64 Kbytes of external memory to be accessed through a single 8-Kbyte window. To access eight Kbytes, or 2^{13} address locations, the CPU will need 13 address lines, ADDR[12:0]. The number of memory banks needed is the total memory, 64 Kbytes divided by the window size, eight Kbytes. This yields eight memory banks, or 2^3. Thus, three expansion lines are required, so expansion address lines XA[15:13] replace CPU address lines ADDR[15:13]. **Figure 1-1** shows a memory map and schematic drawing of this system.

Electrical Characteristics



DC TESTING

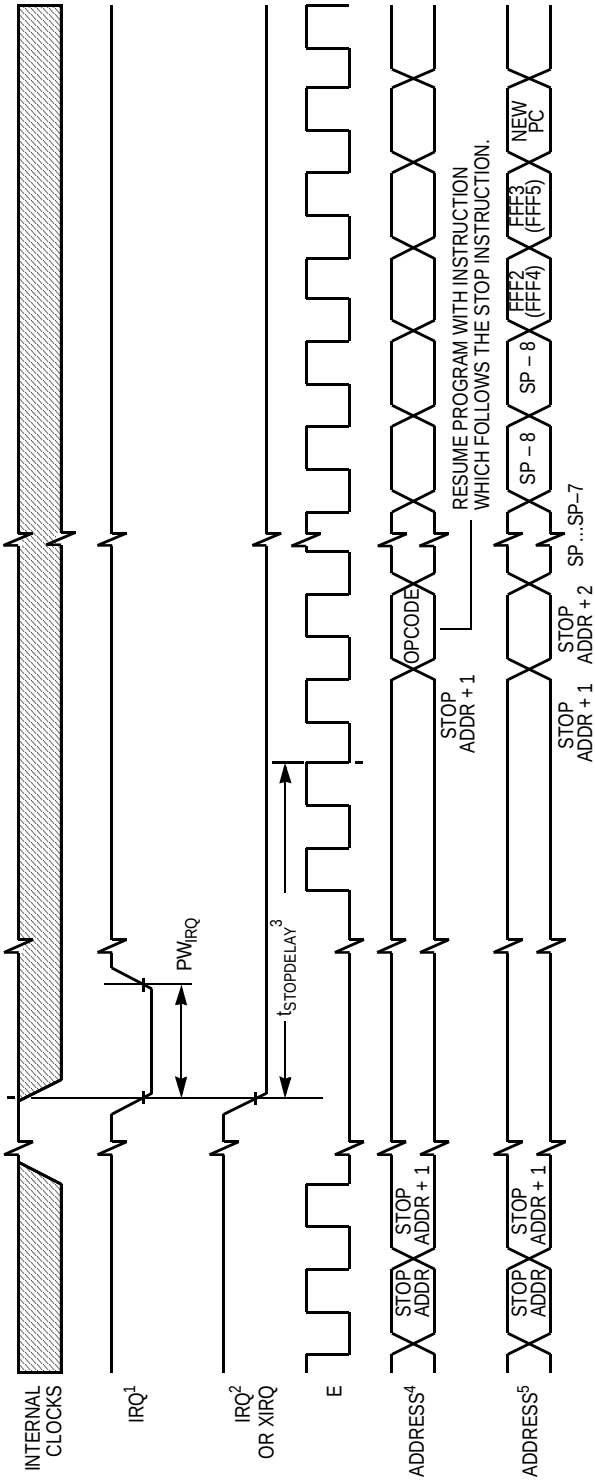


AC TESTING

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.

Figure 12-1. Test Methods



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

Figure 12-4. STOP Recovery Timing Diagram

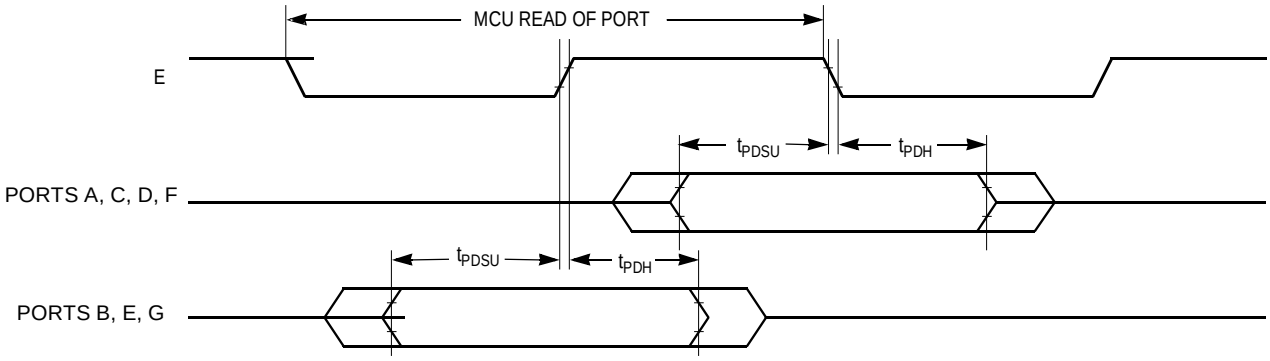


Figure 12-7. Port Read Timing Diagram

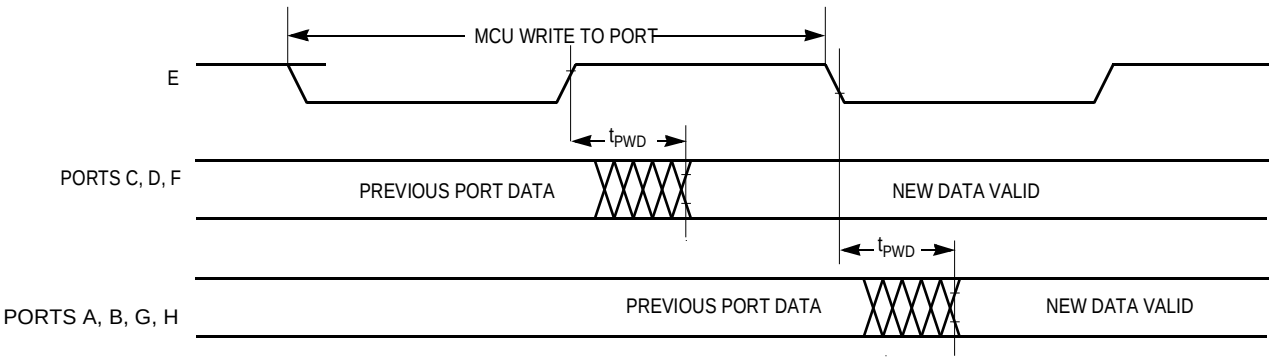


Figure 12-8. Port Write Timing Diagram

12.10 Analog-to-Digital Converter Characteristics

Characteristic[1]	Parameter	Min	Absolute	Max		Unit
				$f_o \leq 2.0$ MHz	$f_o > 2.0$ MHz[2]	
Resolution	Number of bits resolved by A/D converter	—	8	—	—	Bits
Non-linearity	Maximum deviation from the ideal A/D transfer characteristics	—	—	± 1/2	± 1	LSB
Zero error	Difference between the output of an ideal and an actual for zero input voltage	—	—	± 1/2	± 1	LSB
Full scale error	Difference between the output of an ideal and an actual A/D for full-scale input voltage	—	—	± 1/2	± 1	LSB
Total unadjusted error	Maximum sum of non-linearity, zero error, and full-scale error[3]	—	—	± 1/2	± 1 1/2	LSB
Quantization error	Uncertainty because of converter resolution	—	—	± 1/2	± 1/2	LSB
Absolute accuracy	Difference between the actual input voltage and the full-scale weighted equivalent of the binary output code, all error sources included	—	—	± 1	± 2	LSB
Conversion range	Analog input voltage range	V_{RL}	—	V_{RH}	V_{RH}	V
V_{RH}	Maximum analog reference voltage[3]	V_{RL}	—	V_{DD} + 0.1	V_{DD} + 0.1	V
V_{RL}	Minimum analog reference voltage[3]	V_{SS} –0.1	—	V_{RH}	V_{RH}	V
ΔV_R	Minimum difference between V_{RH} and V_{RL}[3]	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} µs
Monotonicity	Conversion result never decreases with an increase in input voltage and has no missing codes		Guaranteed			
Zero Input reading	Conversion result when $V_{In} = V_{RL}$	00	—	—	—	Hex
Full Scale reading	Conversion result when $V_{In} = V_{RH}$	—	—	FF	FF	Hex

Continued