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	HC05
Core Size	8-Bit
Speed	2.1MHz
Connectivity	SIO
Peripherals	POR, WDT
Number of I/O	21
Program Memory Size	4.5KB (4.5K x 8)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	176 x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 5.5V
Data Converters	A/D 4x8b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-SOIC (0.295", 7.50mm Width)
Supplier Device Package	28-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc705p6ecdwe

Chapter 4
Resets

Chapter 5
Interrupts

Chapter 6
Input/Output Ports

Chapter 7
Serial Input/Output Port (SIOP)

1.3 Functional Pin Description

The following paragraphs describe the functionality of each pin on the MC68HC705P6A package. Pins connected to subsystems described in other chapters provide a reference to the chapter instead of a detailed functional description.

1.3.1 V_{DD} and V_{SS}

Power is supplied to the MCU through V_{DD} and V_{SS}. V_{DD} is connected to a regulated +5 volt supply and V_{SS} is connected to ground.

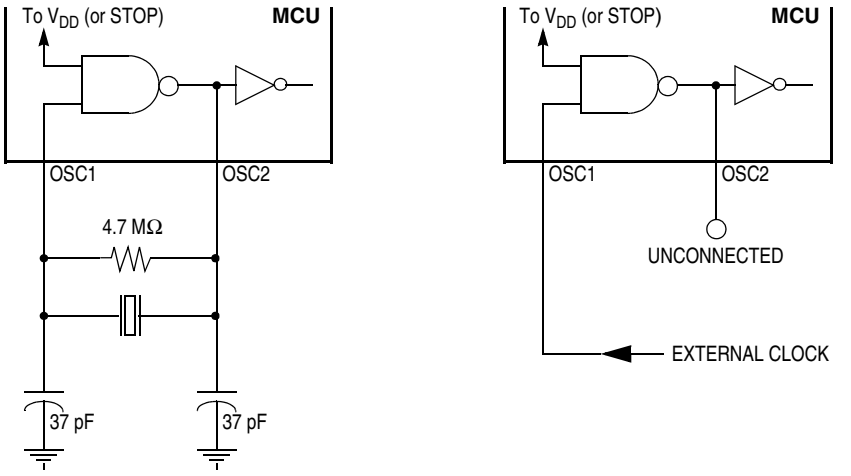
Very fast signal transitions occur on the MCU pins. The short rise and fall times place very high short-duration current demands on the power supply. To prevent noise problems, take special care to provide good power supply bypassing at the MCU. Use bypass capacitors with good high-frequency characteristics and position them as close to the MCU as possible. Bypassing requirements vary, depending on how heavily the MCU pins are loaded.

1.3.2 OSC1 and OSC2

The OSC1 and OSC2 pins are the control connections for the on-chip oscillator. The OSC1 and OSC2 pins can accept the following:

1. A crystal as shown in Figure 1-2(a)
2. A ceramic resonator as shown in Figure 1-2(a)
3. An external clock signal as shown in Figure 1-2(b)

The frequency, f_{osc}, of the oscillator or external clock source is divided by two to produce the internal bus clock operating frequency, f_{op}. The oscillator cannot be turned off by software unless the MOR bit, SWAIT, is clear when a STOP instruction is executed.



(a) Crystal or Ceramic Resonator Connections

(b) External Clock Source Connections

Figure 1-2. Oscillator Connections

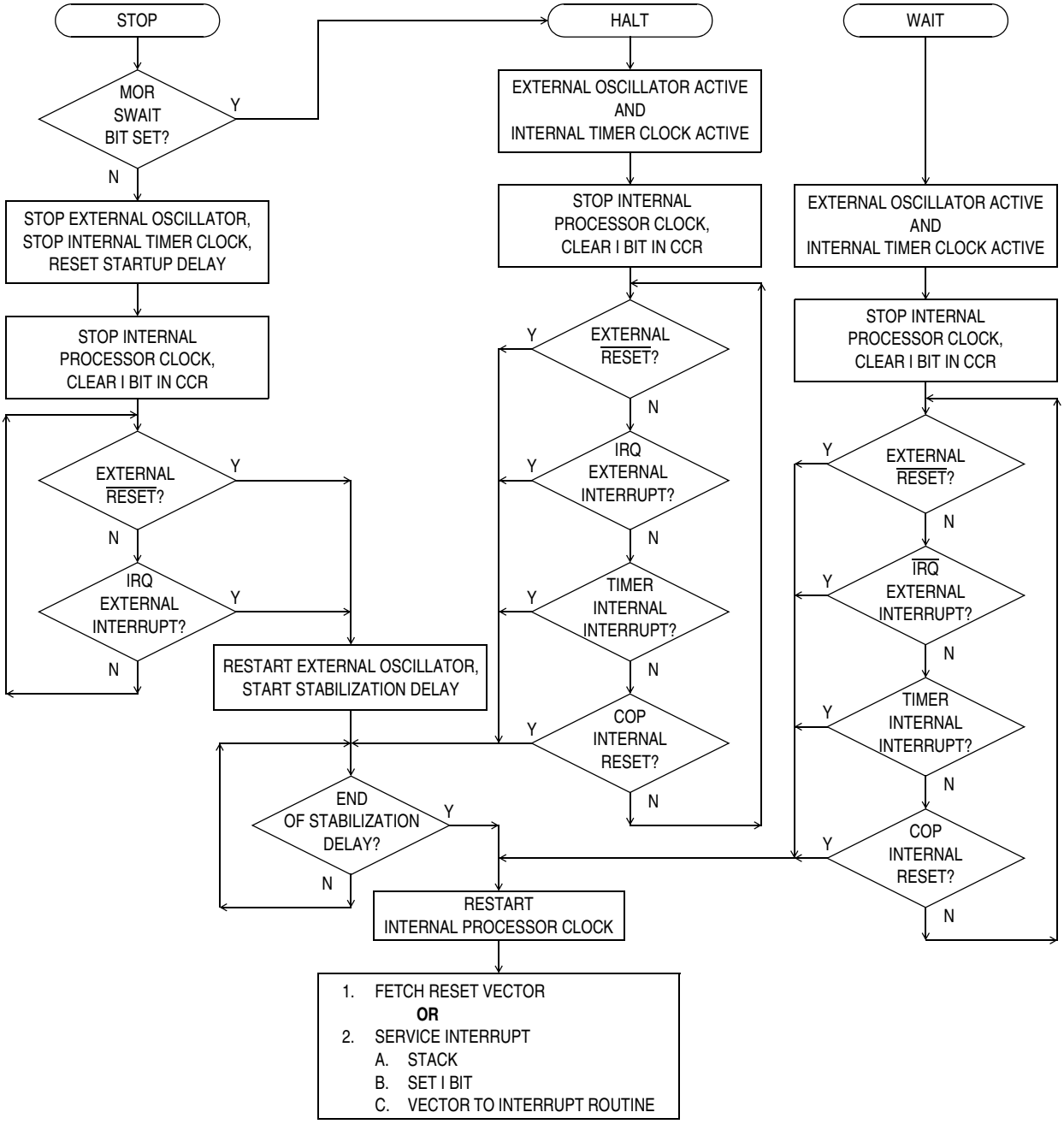


Figure 3-2. STOP/WAIT Flowcharts

3.4.1.2 Halt Mode

NOTE

*Halt mode is **NOT** designed for intentional use. Halt mode is only provided to keep the COP watchdog timer active in the event a STOP instruction is executed inadvertently. This mode of operation is usually achieved by invoking wait mode.*

Execution of the STOP instruction when the SWAIT bit in the MOR is set places the MCU in this low-power mode. Halt mode consumes the same amount of power as wait mode (both halt and wait modes consume more power than stop mode).

In halt mode, the internal clock is halted, suspending all processor and internal bus activity. Internal timer clocks remain active, permitting interrupts to be generated from the 16-bit timer or a reset to be generated from the COP watchdog timer. Execution of the STOP instruction automatically clears the I bit in the condition code register, enabling the $\overline{\text{IRQ}}$ external interrupt. All other registers, memory, and input/output lines remain in their previous states.

If the 16-bit timer interrupt is enabled, it will cause the processor to exit the halt mode and resume normal operation. The halt mode also can be exited when an $\overline{\text{IRQ}}$ external interrupt or external $\overline{\text{RESET}}$ occurs. When exiting the halt mode, the internal clock will resume after a delay of one to 4064 internal clock cycles. This varied delay time is the result of the halt mode exit circuitry testing the oscillator stabilization delay timer (a feature of the stop mode), which has been free-running (a feature of the wait mode).

3.4.2 WAIT Instruction

The WAIT instruction places the MCU in a low-power mode which consumes more power than stop mode. In wait mode, the internal clock is halted, suspending all processor and internal bus activity. Internal timer clocks remain active, permitting interrupts to be generated from the 16-bit timer and reset to be generated from the COP watchdog timer. Execution of the WAIT instruction automatically clears the I bit in the condition code register, enabling the $\overline{\text{IRQ}}$ external interrupt. All other registers, memory, and input/output lines remain in their previous state.

If the 16-bit timer interrupt is enabled, it will cause the processor to exit wait mode and resume normal operation. The 16-bit timer may be used to generate a periodic exit from wait mode. Wait mode may also be exited when an $\overline{\text{IRQ}}$ external interrupt or $\overline{\text{RESET}}$ occurs.

3.5 COP Watchdog Timer Considerations

The COP watchdog timer is active in user mode of operation when the COP bit in the MOR is set. Executing the STOP instruction when the SWAIT bit in the MOR is clear will cause the COP to be disabled. Therefore, it is recommended that the STOP instruction be modified to produce halt mode (set bit SWAIT in the MOR) if the COP watchdog timer is required to function at all times.

Furthermore, it is recommended that the COP watchdog timer be disabled for applications that will use the wait mode for time periods that will exceed the COP timeout period.

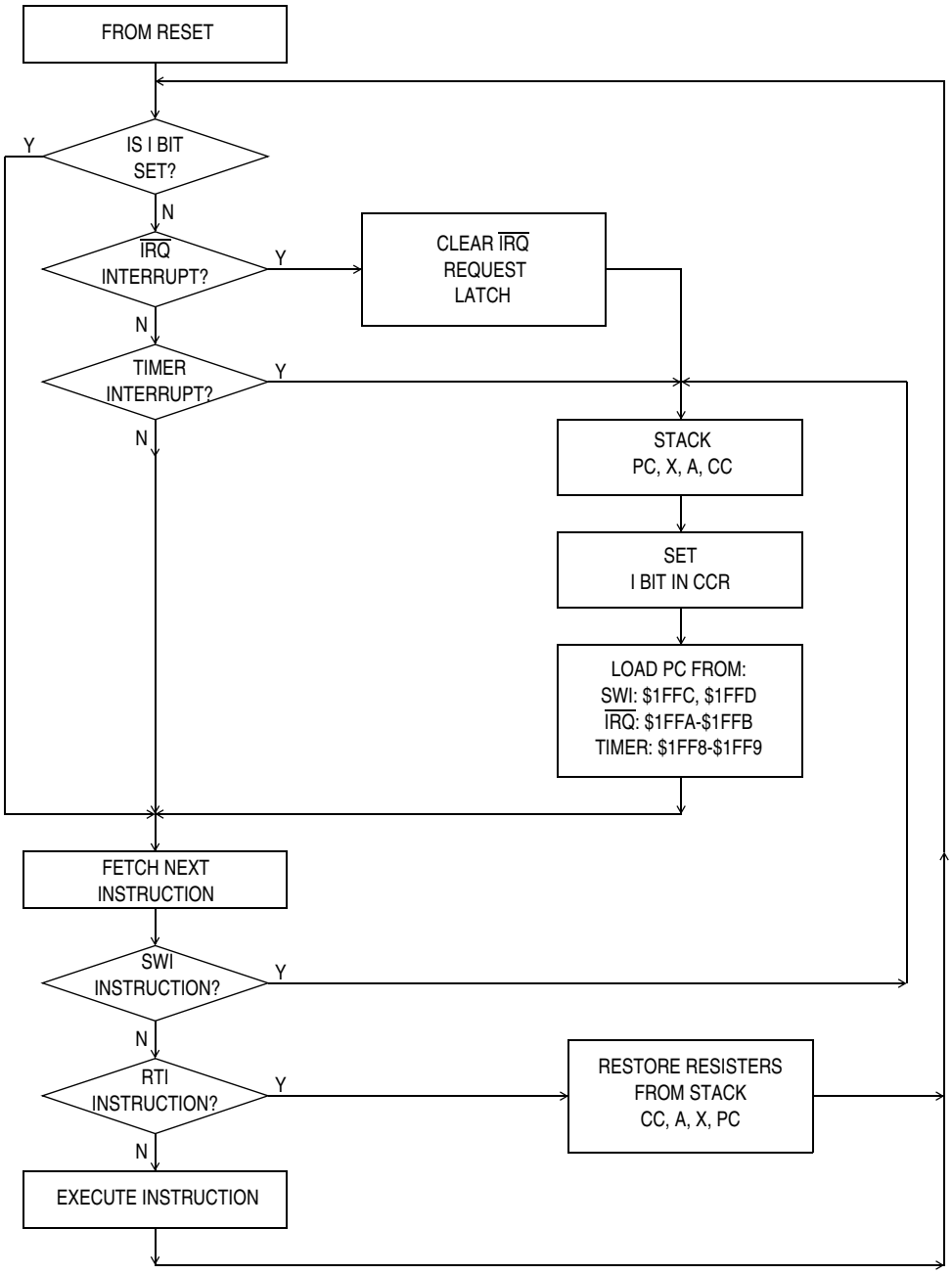


Figure 5-1. Interrupt Processing Flowchart

5.2 Interrupt Types

The interrupts fall into three categories: reset, software, and hardware.

5.2.1 Reset Interrupt Sequence

The reset function is not in the strictest sense an interrupt; however, it is acted upon in a similar manner as shown in Figure 5-1. A low-level input on the $\overline{\text{RESET}}$ pin or internally generated RST signal causes the program to vector to its starting address which is specified by the contents of memory locations $1FFE and $1FFF. The I bit in the condition code register is also set. The MCU is configured to a known state during this type of reset as previously described in Chapter 4 Resets.

5.2.2 Software Interrupt (SWI)

The SWI is an executable instruction. It is also a non-maskable interrupt since it is executed regardless of the state of the I bit in the CCR. As with any instruction, interrupts pending during the previous instruction will be serviced before the SWI opcode is fetched. The interrupt service routine address for the SWI instruction is specified by the contents of memory locations $1FFC and $1FFD.

5.2.3 Hardware Interrupts

All hardware interrupts are maskable by the I bit in the CCR. If the I bit is set, all hardware interrupts (internal and external) are disabled. Clearing the I bit enables the hardware interrupts. Four hardware interrupts are explained in the following subsections.

5.2.3.1 External Interrupt ($\overline{IRQ}$)

The $\overline{\text{IRQ}}/V_{PP}$ pin drives an asynchronous interrupt to the CPU. An edge detector flip-flop is latched on the falling edge of $\overline{\text{IRQ}}/V_{PP}$. If either the output from the internal edge detector flip-flop or the level on the $\overline{\text{IRQ}}/V_{PP}$ pin is low, a request is synchronized to the CPU to generate the IRQ interrupt. If the LEVEL bit in the mask option register is clear (edge-sensitive only), the output of the internal edge detector flip-flop is sampled and the input level on the $\overline{\text{IRQ}}/V_{PP}$ pin is ignored. The interrupt service routine address is specified by the contents of memory locations $1FFA and $1FFB. If the port A interrupts are enabled by the MOR, they generate external interrupts identically to the $\overline{\text{IRQ}}/V_{PP}$ pin.

NOTE

The internal interrupt latch is cleared nine internal clock cycles after the interrupt is recognized (immediately after location $1FFA is read). Therefore, another external interrupt pulse could be latched during the $\overline{IRQ}$ service routine.

Another interrupt will be serviced if the $\overline{IRQ}$ pin is still in a low state when the RTI in the service routine is executed.

5.2.3.2 Input Capture Interrupt

The input capture interrupt is generated by the 16-bit timer as described in Chapter 8 Capture/Compare Timer. The input capture interrupt flag is located in register TSR and its corresponding enable bit can be found in register TCR. The I bit in the CCR must be clear for the input capture interrupt to be enabled. The interrupt service routine address is specified by the contents of memory locations $1FF8 and $1FF9.

6.3 Port B

Port B is a 3-bit bidirectional port which can share pins PB5–PB7 with the SIOP communications subsystem. The port B data register is located at address $0001 and its data direction register (DDR) is located at address $0005. The contents of the port B data register are indeterminate at initial powerup and must be initialized by user software. Reset does not affect the data registers, but clears the DDRs, thereby setting all of the port pins to input mode. Writing a 1 to a DDR bit sets the corresponding port pin to output mode (see Figure 6-2).

Port B may be used for general I/O applications when the SIOP subsystem is disabled. The SPE bit in register SPCR is used to enable/disable the SIOP subsystem. When the SIOP subsystem is enabled, port B registers are still accessible to software. Writing to either of the port B registers while a data transfer is under way could corrupt the data. See Chapter 7 Serial Input/Output Port (SIOP) for a discussion of the SIOP subsystem.

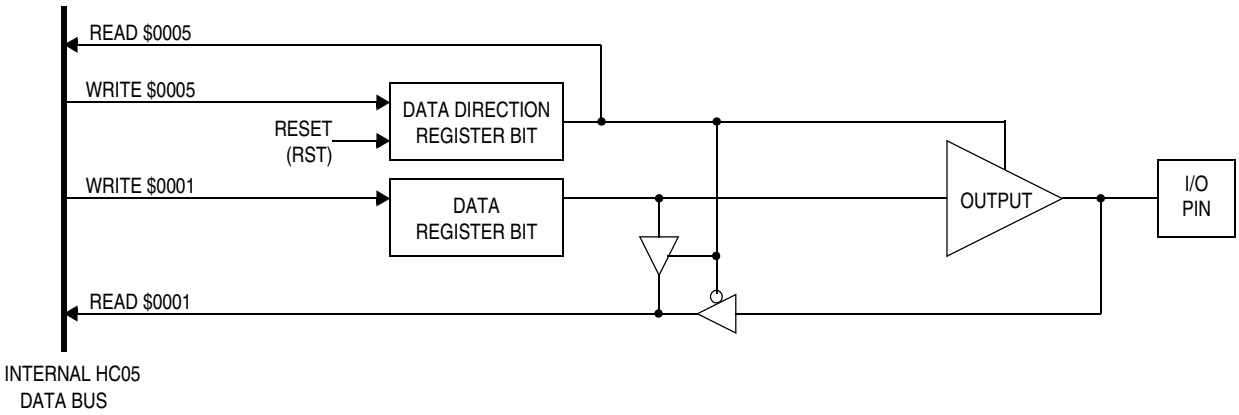


Figure 6-2. Port B I/O Circuitry

6.4 Port C

Port C is an 8-bit bidirectional port which can share pins PC3–PC7 with the A/D subsystem. The port C data register is located at address $0002 and its data direction register (DDR) is located at address $0006. The contents of the port C data register are indeterminate at initial powerup and must be initialized by user software. Reset does not affect the data registers, but clears the DDRs, thereby setting all of the port pins to input mode. Writing a 1 to a DDR bit sets the corresponding port pin to output mode (see Figure 6-3).

Port C may be used for general I/O applications when the A/D subsystem is disabled. The ADON bit in register ADSC is used to enable/disable the A/D subsystem. Care must be exercised when using pins PC0–PC2 while the A/D subsystem is enabled. Accidental changes to bits that affect pins PC3–PC7 in the data or DDR registers will produce unpredictable results in the A/D subsystem. See Chapter 9 Analog Subsystem.

Table 6-1. Port A I/O Functions

DDRA	I/O Pin Mode	Accesses to DDRA @ $0004	Accesses to Data Register @ $0000	
		Read/Write	Read	Write
0	IN, Hi-Z	DDRA0–DDRA7	I/O Pin	See Note
1	OUT	DDRA0–DDRA7	PA0–PA7	PA0–PA7

Note: Does not affect input, but stored to data register

Table 6-2. Port B I/O Functions

DDRB	I/O Pin Mode	Accesses to DDRB @ $0005	Accesses to Data Register @ $0001	
		Read/Write	Read	Write
0	IN, Hi-Z	DDRB5–DDRB7	I/O Pin	See Note
1	OUT	DDRB5–DDRB7	PB5–PB7	PB5–PB7

Note: Does not affect input, but stored to data register

Table 6-3. Port C I/O Functions

DDRC	I/O Pin Mode	Accesses to DDRC @ $0006	Accesses to Data Register @ $0002	
		Read/Write	Read	Write
0	IN, Hi-Z	DDRC0–DDRC7	I/O Pin	See Note
1	OUT	DDRC0–DDRC7	PC0–PC7	PC0–PC7

Note: Does not affect input, but stored to data register

Table 6-4. Port D I/O Functions

DDRD	I/O Pin Mode	Accesses to DDRD @ $0007	Accesses to Data Register @ $0003	
		Read/Write	Read	Write
0	IN, Hi-Z	DDRD5	I/O Pin	See Note 1
1	OUT	DDRD5	PD5	PD5

Notes:
1. Does not affect input, but stored to data register
2. PD7 is input only

NOTE

To avoid generating a glitch on an I/O port pin, data should be written to the I/O port data register before writing a logic 1 to the corresponding data direction register.

At power-on or reset, all DDRs are cleared, which configures all port pins as inputs. The DDRs are capable of being written to or read by the processor. During the programmed output state, a read of the data register will actually read the value of the output data latch and not the level on the I/O port pin.

Chapter 7
Serial Input/Output Port (SIOP)

7.1 Introduction

The simple synchronous serial I/O port (SIOP) subsystem is designed to provide efficient serial communications between peripheral devices or other MCUs. The SIOP is implemented as a 3-wire master/slave system with serial clock (SCK), serial data input (SDI), and serial data output (SDO). A block diagram of the SIOP is shown in Figure 7-1. A mask programmable option determines whether the SIOP is MSB or LSB first.

The SIOP subsystem shares its input/output pins with port B. When the SIOP is enabled (SPE bit set in register SCR), port B DDR and data registers are modified by the SIOP. Although port B DDR and data registers can be altered by application software, these actions could affect the transmitted or received data.

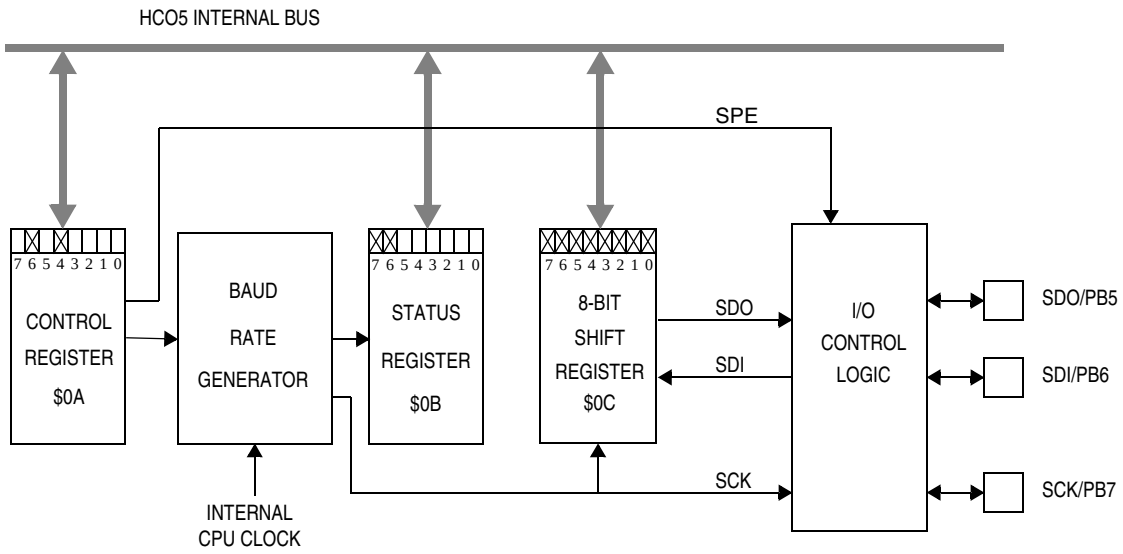


Figure 7-1. SIOP Block Diagram

7.3 SIOP Registers

The SIOP is programmed and controlled by the SIOP control register (SCR) located at address $000A, the SIOP status register (SSR) located at address $000B, and the SIOP data register (SDR) located at address $000C.

7.3.1 SIOP Control Register (SCR)

This register is located at address $000A and contains two bits. Figure 7-3 shows the position of each bit in the register and indicates the value of each bit after reset.

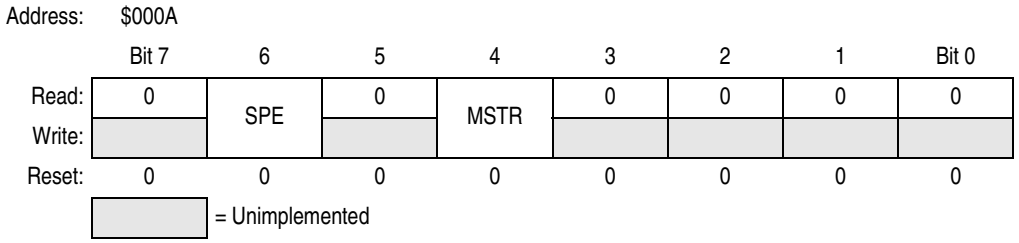


Figure 7-3. SIOP Control Register (SCR)

SPE — Serial Peripheral Enable

When set, the SPE bit enables the SIOP subsystem such that SDO/PB5 is the serial data output, SDI/PB6 is the serial data input, and SCK/PB7 is a serial clock input in the slave mode or a serial clock output in the master mode. Port B DDR and data registers can be manipulated as usual (except for PB5); however, these actions could affect the transmitted or received data.

The SPE bit is readable at any time. However, writing to the SIOP control register while a transmission is in progress will cause the SPIF and DCOL bits in the SIOP status register (see below) to operate incorrectly. Therefore, the SIOP control register should be written once to enable the SIOP and then not written to until the SIOP is to be disabled. Clearing the SPE bit while a transmission is in progress will 1) abort the transmission, 2) reset the serial bit counter, and 3) convert the port B/SIOP port to a general-purpose I/O port. Reset clears the SPE bit.

MSTR — Master Mode Select

When set, the MSTR bit configures the serial I/O port for master mode. A transfer is initiated by writing to the SDR. Also, the SCK pin becomes an output providing a synchronous data clock dependent upon the oscillator frequency. When the device is in slave mode, the SDO and SDI pins do not change function. These pins behave exactly the same in both the master and slave modes.

The MSTR bit is readable and writeable at any time regardless of the state of the SPE bit. Clearing the MSTR bit will abort any transfers that may have been in progress. Reset clears the MSTR bit as well as the SPE bit, disabling the SIOP subsystem.

8.2 Timer Operation

The core of the capture/compare timer is a 16-bit free-running counter. The counter provides the timing reference for the input capture and output compare functions. The input capture and output compare functions provide a means to latch the times at which external events occur, to measure input waveforms, and to generate output waveforms and timing delays. Software can read the value in the 16-bit free-running counter at any time without affecting the counter sequence.

Because of the 16-bit timer architecture, the I/O registers for the input capture and output compare functions are pairs of 8-bit registers.

Because the counter is 16 bits long and preceded by a fixed divide-by-4 prescaler, the counter rolls over every 262,144 internal clock cycles. Timer resolution with a 4-MHz crystal is 2 μs.

8.2.1 Input Capture

The input capture function is a means to record the time at which an external event occurs. When the input capture circuitry detects an active edge on the TCAP pin, it latches the contents of the timer registers into the input capture registers. The polarity of the active edge is programmable.

Latching values into the input capture registers at successive edges of the same polarity measures the period of the input signal on the TCAP pin. Latching values into the input capture registers at successive edges of opposite polarity measures the pulse width of the signal.

8.2.2 Output Compare

The output compare function is a means of generating an output signal when the 16-bit counter reaches a selected value. Software writes the selected value into the output compare registers. On every fourth internal clock cycle the output compare circuitry compares the value of the counter to the value written in the output compare registers. When a match occurs, the timer transfers the programmable output level bit (OLVL) from the timer control register to the TCMP pin.

The programmer can use the output compare register to measure time periods, to generate timing delays, or to generate a pulse of specific duration or a pulse train of specific frequency and duty cycle on the TCMP pin.

8.3 Timer I/O Registers

The following I/O registers control and monitor timer operation:

- Timer control register (TCR)
- Timer status register (TSR)
- Timer registers (TRH and TRL)
- Alternate timer registers (ATRH and ATRL)
- Input capture registers (ICRH and ICRL)
- Output compare registers (OCRH and OCRL)

8.3.2 Timer Status Register

The timer status register (TSR), shown in Figure 8-3, contains flags to signal the following conditions:

- An active signal on the TCAP pin, transferring the contents of the timer registers to the input capture registers
- A match between the 16-bit counter and the output compare registers, transferring the OLVL bit to the TCMP pin
- A timer roll over from $FFFF to $0000

Address: $0013

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	ICF	OCF	TOF	0	0	0	0	0
Write:								
Reset:	U	U	U	0	0	0	0	0

= Unimplemented U = Undetermined

Figure 8-3. Timer Status Register (TSR)

ICF — Input Capture Flag

The ICF bit is set automatically when an edge of the selected polarity occurs on the TCAP pin. Clear the ICF bit by reading the timer status register with ICF set and then reading the low byte ($0015) of the input capture registers. Resets have no effect on ICF.

OCF — Output Compare Flag

The OCF bit is set automatically when the value of the timer registers matches the contents of the output compare registers. Clear the OCF bit by reading the timer status register with OCF set and then reading the low byte ($0017) of the output compare registers. Resets have no effect on OCF.

TOF — Timer Overflow Flag

The TOF bit is set automatically when the 16-bit counter rolls over from $FFFF to $0000. Clear the TOF bit by reading the timer status register with TOF set, and then reading the low byte ($0019) of the timer registers. Resets have no effect on TOF.

Chapter 10
EPROM

10.1 Introduction

The user EPROM consists of 48 bytes of user page zero EPROM from $0020 to $004F, 4608 bytes of user EPROM from $0100 to $12FF, the two MOR reset values located at $1EFF and $1F00, and 16 bytes of user vectors EPROM from $1FF0 to $1FFF. The bootloader ROM and vectors are located from $1F01 to $1FEF.

10.2 EPROM Erasing

NOTE

Only parts packaged in a windowed package may be erased. Others are one-time programmable and may not be erased by UV exposure.

The MC68HC705P6A can be erased by exposure to a high-intensity ultraviolet (UV) light with a wavelength of 2537 angstroms. The recommended dose (UV intensity multiplied by exposure time) is 15 Ws/cm^2. UV lamps without shortwave filters should be used, and the EPROM device should be positioned about one inch from the UV lamp. An erased EPROM byte will read as $00.

10.3 EPROM Programming Sequence

The bootloader software goes through a complete write cycle of the EPROM including the MOR. This is followed by a verify cycle which continually branches in a loop if an error is found. A sample routine to program a byte of EPROM is shown in Table 10-1.

NOTE

To avoid damage to the MCU, V_{DD} must be applied to the MCU before V_{PP}.

10.4 EPROM Registers

Three registers are associated with the EPROM: the EPROM programming register (EPROG) and the two mask option registers (MOR). The EPROG register controls the actual programming of the EPROM bytes and the MOR. The MOR registers control the six mask options found on the ROM version of this MCU (MC68HC05P6), the EPROM security feature, and eight additional port A interrupt options.

10.5 EPROM Programming Register (EPROG)

This register is used to program the EPROM array. Only the ELAT and EPGM bits are available. Table 10-1 shows the location of each bit in the EPROG register and the state of these bits coming out of reset. All the bits in the EPROG register are cleared by reset.

11.3 MOR Programming

The contents of the MOR should be programmed in bootloader mode using the hardware shown in Figure 10-2. MC68HC705P6A EPROM Programming Flowchart. In order to allow programming, all the implemented bits in the MOR are essentially read-write bits in bootloader mode as shown in Figure 11-1.

The programming of the MOR is the same as user EPROM.

1. Set the ELAT bit in the EPROG register.
2. Write the desired data to the desired MOR address.
3. Set the EPGM bit in the EPROG.
4. Wait for the programming time (t_{EPGM}).
5. Clear the ELAT and EPGM bits in the EPROG.
6. Remove the programming voltage from the $\overline{IRQ}/V_{PP}$ pin.

A sample routine to program a byte of EPROM is shown in Table 11-2.

Once the MOR bits have been programmed, the options are not loaded into the MOR registers until the part is reset.

Table 11-2. MOR Programming Routine

```
001C              EPROG   EQU  $1C           PROGRAMMING REG
00FF              DATA2   EQU  $FF           SAMPLE MOR VALUES
0023              DATA1   EQU  #23
1EFF              MOR2    EQU  $1EFF         MOPR ADDRESSES
1F00              MOR1    EQU  $1F00
0000              EPGM    EQU  $00           EPGM BIT IN EPROG REG

00E0                      ORG  $E0

00E0   A6 04              LDA  #$04          SET ELAT BIT
00E2   B7 1C              STA  EPROG         IN EPGM REG AT $1C
00E4   A6 FF              LDA  #DATA2        DATA BYTE
00E6   C7 1E FF           STA  MOR2          WRITE IT TO MOR LOC
00E9   12 1C              BSET EPGM,EPROG    TURN ON PGM VOLTAGE
00EB   AD 03              BSR  DELAY         WAIT 4 ms MINIMUM
00ED   3F 1C              CLR  EPROG         CLR EPGM REGISTER
00EF   81                 RTS
```

12.2.2 Index Register

The index register shown in Figure 12-1 is an 8-bit register that can perform two functions:

- Indexed addressing
- Temporary storage

In indexed addressing with no offset, the index register contains the low byte of the operand address, and the high byte is assumed to be $00. In indexed addressing with an 8-bit offset, the CPU finds the operand address by adding the index register contents to an 8-bit immediate value. In indexed addressing with a 16-bit offset, the CPU finds the operand address by adding the index register contents to a 16-bit immediate value.

The index register can also serve as an auxiliary accumulator for temporary storage. The index register is unaffected by a reset of the device.

12.2.3 Stack Pointer

The stack pointer shown in Figure 12-1 is a 16-bit register internally. In devices with memory maps less than 64 Kbytes, the unimplemented upper address lines are ignored. The stack pointer contains the address of the next free location on the stack. During a reset or the reset stack pointer (RSP) instruction, the stack pointer is set to $00FF. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack.

When accessing memory, the 10 most significant bits are permanently set to 0000000011. The six least significant register bits are appended to these 10 fixed bits to produce an address within the range of $00FF to $00C0. Subroutines and interrupts may use up to 64 ($40) locations. If 64 locations are exceeded, the stack pointer wraps around and writes over the previously stored information. A subroutine call occupies two locations on the stack and an interrupt uses five locations.

12.2.4 Program Counter

The program counter shown in Figure 12-1 is a 16-bit register internally. In devices with memory maps less than 64 Kbytes, the unimplemented upper address lines are ignored. The program counter contains the address of the next instruction or operand to be fetched.

Normally, the address in the program counter increments to the next sequential memory location every time an instruction or operand is fetched. Jump, branch, and interrupt operations load the program counter with an address other than that of the next sequential location.

12.2.5 Condition Code Register

The CCR shown in Figure 12-1 is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. The fifth bit is the interrupt mask. These bits can be individually tested by a program, and specific actions can be taken as a result of their state. The condition code register should be thought of as having three additional upper bits that are always ones. Only the interrupt mask is affected by a reset of the device. The following paragraphs explain the functions of the lower five bits of the condition code register.

H — Half Carry Bit

When the half-carry bit is set, it means that a carry occurred between bits 3 and 4 of the accumulator during the last ADD or ADC (add with carry) operation. The half-carry bit is required for binary-coded decimal (BCD) arithmetic operations.

13.3.4 Bit Manipulation Instructions

The CPU can set or clear any writable bit in the first 256 bytes of memory, which includes I/O registers and on-chip RAM locations. The CPU can also test and branch based on the state of any bit in any of the first 256 memory locations.

Table 13-4. Bit Manipulation Instructions

Instruction	Mnemonic
Bit Clear	BCLR
Branch if Bit Clear	BRCLR
Branch if Bit Set	BRSET
Bit Set	BSET

13.3.5 Control Instructions

These instructions act on CPU registers and control CPU operation during program execution.

Table 13-5. Control Instructions

Instruction	Mnemonic
Clear Carry Bit	CLC
Clear Interrupt Mask	CLI
No Operation	NOP
Reset Stack Pointer	RSP
Return from Interrupt	RTI
Return from Subroutine	RTS
Set Carry Bit	SEC
Set Interrupt Mask	SEI
Stop Oscillator and Enable $\overline{\text{IRQ}}$ Pin	STOP
Software Interrupt	SWI
Transfer Accumulator to Index Register	TAX
Transfer Index Register to Accumulator	TXA
Stop CPU Clock and Enable Interrupts	WAIT

Table 13-6. Instruction Set Summary (Sheet 3 of 6)

Source Form	Operation	Description	Effect on CCR					Address Mode	Opcode	Operand	Cycles
			H	I	N	Z	C				
CLR *opr* CLRA CLRX CLR *opr*,X CLR ,X	Clear Byte	M ← $00 A ← $00 X ← $00 M ← $00 M ← $00	—	—	0	1	—	DIR INH INH IX1 IX	3F 4F 5F 6F 7F	dd ff 	5 3 3 6 5
CMP #*opr* CMP *opr* CMP *opr* CMP *opr*,X CMP *opr*,X CMP ,X	Compare Accumulator with Memory Byte	(A) – (M)	—	—	↕	↕	↕	IMM DIR EXT IX2 IX1 IX	A1 B1 C1 D1 E1 F1	ii dd hh ll ee ff ff 	2 3 4 5 4 3
COM *opr* COMA COMX COM *opr*,X COM ,X	Complement Byte (One's Complement)	M ← ($\overline{M}$) = $FF – (M) A ← ($\overline{A}$) = $FF – (A) X ← ($\overline{X}$) = $FF – (X) M ← ($\overline{M}$) = $FF – (M) M ← ($\overline{M}$) = $FF – (M)	—	—	↕	↕	1	DIR INH INH IX1 IX	33 43 53 63 73	dd ff 	5 3 3 6 5
CPX #*opr* CPX *opr* CPX *opr* CPX *opr*,X CPX *opr*,X CPX ,X	Compare Index Register with Memory Byte	(X) – (M)	—	—	↕	↕	↕	IMM DIR EXT IX2 IX1 IX	A3 B3 C3 D3 E3 F3	ii dd hh ll ee ff ff 	2 3 4 5 4 3
DEC *opr* DECA DECX DEC *opr*,X DEC ,X	Decrement Byte	M ← (M) – 1 A ← (A) – 1 X ← (X) – 1 M ← (M) – 1 M ← (M) – 1	—	—	↕	↕	—	DIR INH INH IX1 IX	3A 4A 5A 6A 7A	dd ff 	5 3 3 6 5
EOR #*opr* EOR *opr* EOR *opr* EOR *opr*,X EOR *opr*,X EOR ,X	EXCLUSIVE OR Accumulator with Memory Byte	A ← (A) ⊕ (M)	—	—	↕	↕	—	IMM DIR EXT IX2 IX1 IX	A8 B8 C8 D8 E8 F8	ii dd hh ll ee ff ff 	2 3 4 5 4 3
INC *opr* INCA INCX INC *opr*,X INC ,X	Increment Byte	M ← (M) + 1 A ← (A) + 1 X ← (X) + 1 M ← (M) + 1 M ← (M) + 1	—	—	↕	↕	—	DIR INH INH IX1 IX	3C 4C 5C 6C 7C	dd ff 	5 3 3 6 5
JMP *opr* JMP *opr* JMP *opr*,X JMP *opr*,X JMP ,X	Unconditional Jump	PC ← Jump Address	—	—	—	—	—	DIR EXT IX2 IX1 IX	BC CC DC EC FC	dd hh ll ee ff ff 	2 3 4 3 2
JSR *opr* JSR *opr* JSR *opr*,X JSR *opr*,X JSR ,X	Jump to Subroutine	PC ← (PC) + n (n = 1, 2, or 3) Push (PCL); SP ← (SP) – 1 Push (PCH); SP ← (SP) – 1 PC ← Effective Address	—	—	—	—	—	DIR EXT IX2 IX1 IX	BD CD DD ED FD	dd hh ll ee ff ff 	5 6 7 6 5

Table 13-6. Instruction Set Summary (Sheet 5 of 6)

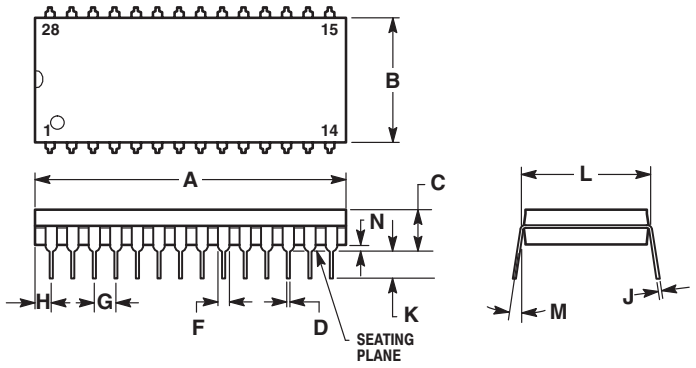
Source Form	Operation	Description	Effect on CCR					Address Mode	Opcode	Operand	Cycles
			H	I	N	Z	C				
RTI	Return from Interrupt	SP ← (SP) + 1; Pull (CCR) SP ← (SP) + 1; Pull (A) SP ← (SP) + 1; Pull (X) SP ← (SP) + 1; Pull (PCH) SP ← (SP) + 1; Pull (PCL)	↕	↕	↕	↕	↕	INH	80		9
RTS	Return from Subroutine	SP ← (SP) + 1; Pull (PCH) SP ← (SP) + 1; Pull (PCL)	—	—	—	—	—	INH	81		6
SBC #*opr* SBC *opr* SBC *opr* SBC *opr*,X SBC *opr*,X SBC ,X	Subtract Memory Byte and Carry Bit from Accumulator	A ← (A) – (M) – (C)	—	—	↕	↕	↕	IMM DIR EXT IX2 IX1 IX	A2 B2 C2 D2 E2 F2	ii dd hh ll ee ff ff	2 3 4 5 4 3
SEC	Set Carry Bit	C ← 1	—	—	—	—	1	INH	99		2
SEI	Set Interrupt Mask	I ← 1	—	1	—	—	—	INH	9B		2
STA *opr* STA *opr* STA *opr*,X STA *opr*,X STA ,X	Store Accumulator in Memory	M ← (A)	—	—	↕	↕	—	DIR EXT IX2 IX1 IX	B7 C7 D7 E7 F7	dd hh ll ee ff ff	4 5 6 5 4
STOP	Stop Oscillator and Enable IRQ Pin		—	0	—	—	—	INH	8E		2
STX *opr* STX *opr* STX *opr*,X STX *opr*,X STX ,X	Store Index Register In Memory	M ← (X)	—	—	↕	↕	—	DIR EXT IX2 IX1 IX	BF CF DF EF FF	dd hh ll ee ff ff	4 5 6 5 4
SUB #*opr* SUB *opr* SUB *opr* SUB *opr*,X SUB *opr*,X SUB ,X	Subtract Memory Byte from Accumulator	A ← (A) – (M)	—	—	↕	↕	↕	IMM DIR EXT IX2 IX1 IX	A0 B0 C0 D0 E0 F0	ii dd hh ll ee ff ff	2 3 4 5 4 3
SWI	Software Interrupt	PC ← (PC) + 1; Push (PCL) SP ← (SP) – 1; Push (PCH) SP ← (SP) – 1; Push (X) SP ← (SP) – 1; Push (A) SP ← (SP) – 1; Push (CCR) SP ← (SP) – 1; I ← 1 PCH ← Interrupt Vector High Byte PCL ← Interrupt Vector Low Byte	—	1	—	—	—	INH	83		1 0
TAX	Transfer Accumulator to Index Register	X ← (A)	—	—	—	—	—	INH	97		2
TST *opr* TSTA TSTX TST *opr*,X TST ,X	Test Memory Byte for Negative or Zero	(M) – $00	—	—	↕	↕	—	DIR INH INH IX1 IX	3D 4D 5D 6D 7D	dd ff	4 3 3 5 4

Chapter 15
Mechanical Specifications

15.1 Introduction

The MC68HC705P6A is available in either a 28-pin plastic dual in-line (PDIP) or a 28-pin small outline integrated circuit (SOIC) package.

15.2 Plastic Dual In-Line Package (Case 710)



NOTES:
1. POSITIONAL TOLERANCE OF LEADS (D), SHALL BE WITHIN 0.25mm (0.010) AT MAXIMUM MATERIAL CONDITION, IN RELATION TO SEATING PLANE AND EACH OTHER.
2. DIMENSION L TO CENTER OF LEADS WHEN FORMED PARALLEL.
3. DIMENSION B DOES NOT INCLUDE MOLD FLASH.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	36.45	37.21	1.435	1.465
B	13.72	14.22	0.540	0.560
C	3.94	5.08	0.155	0.200
D	0.36	0.56	0.014	0.022
F	1.02	1.52	0.040	0.060
G	2.54 BSC		0.100 BSC	
H	1.65	2.16	0.065	0.085
J	0.20	0.38	0.008	0.015
K	2.92	3.43	0.115	0.135
L	15.24 BSC		0.600 BSC	
M	0°	15°	0°	15°
N	0.51	1.02	0.020	0.040

Chapter 16
Ordering Information

16.1 Introduction

This section contains ordering information for the available package types.

16.2 MC Order Numbers

The following table shows the MC order numbers for the available package types.

MC Order Number	Operating Temperature Range
MC68HC705P6ACP[1] (extended)	–40°C to 85°C
MC68HC705P6ACDW[2] (extended)	–40°C to 85°C

1. P = Plastic dual in-line package
2. DW = Small outline integrated circuit (SOIC) package