MC68HC05CxRG/AD
REV 1

# HC05

## MC68HC05C4,C8,C9
## MC68HC705C8
## MC68HC805C4
## MC68HCL05C4,C8
## MC68HSC05C4,C8

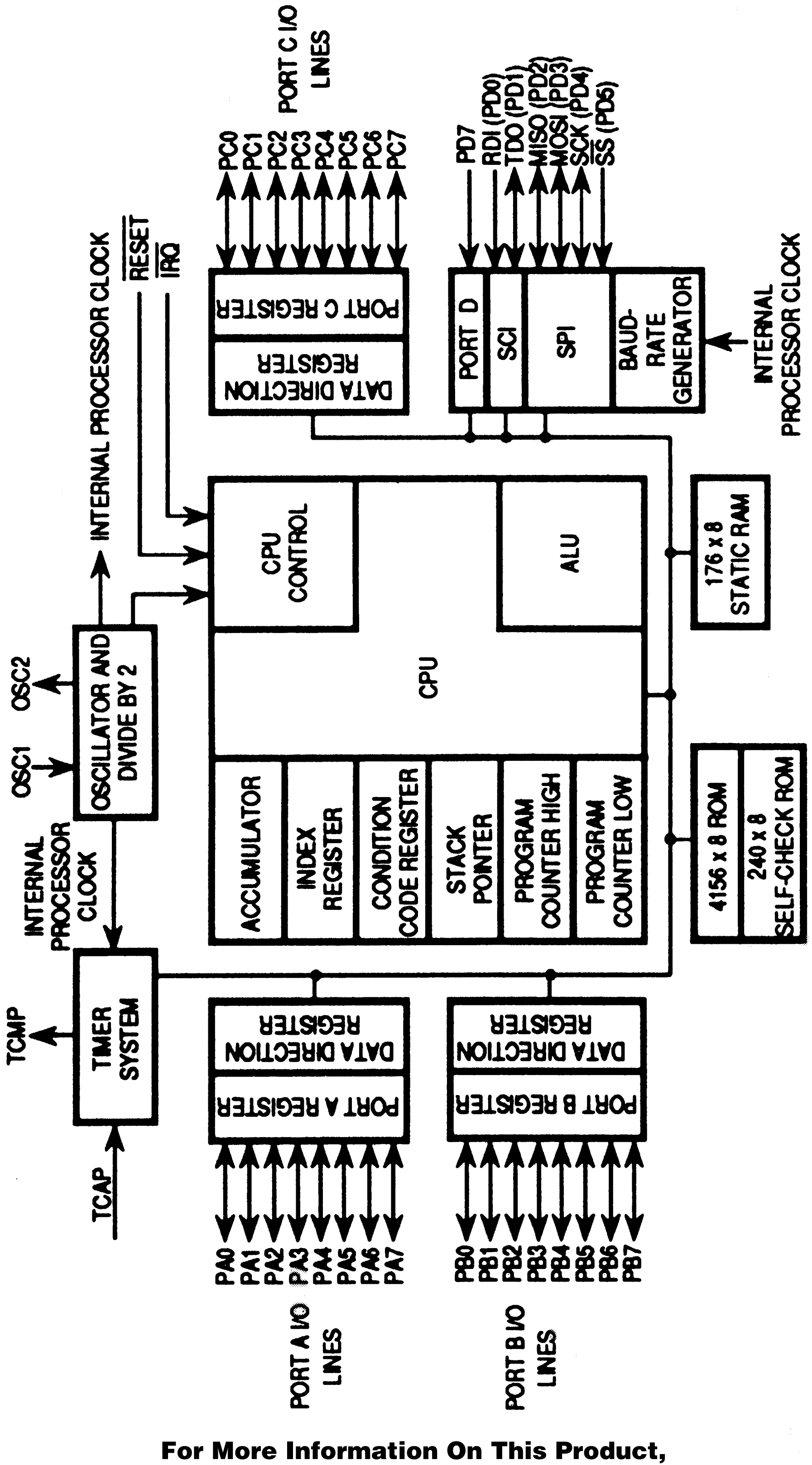
# PROGRAMMING REFERENCE GUIDE

MOTOROLA

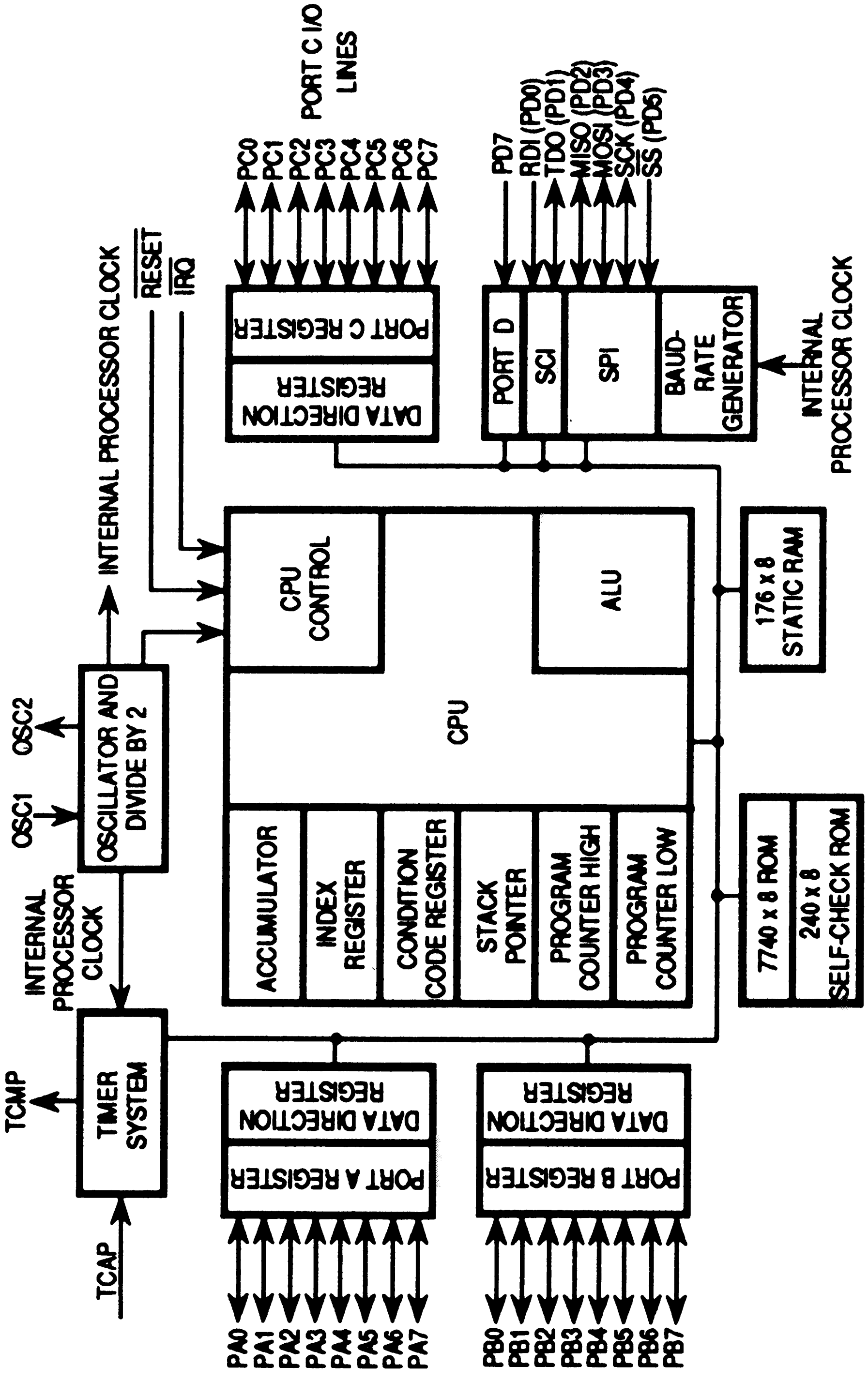
The MC68HC05 Family of HCMOS devices covered in this reference guide are as follows:

MC68HC05C4
MC68HC05C8
MC68HC05C9
MC68HC705C8
MC68HC805C4
MC68HCL05C4
MC68HCL05C8
MC68HSC05C4
MC68HSC05C8

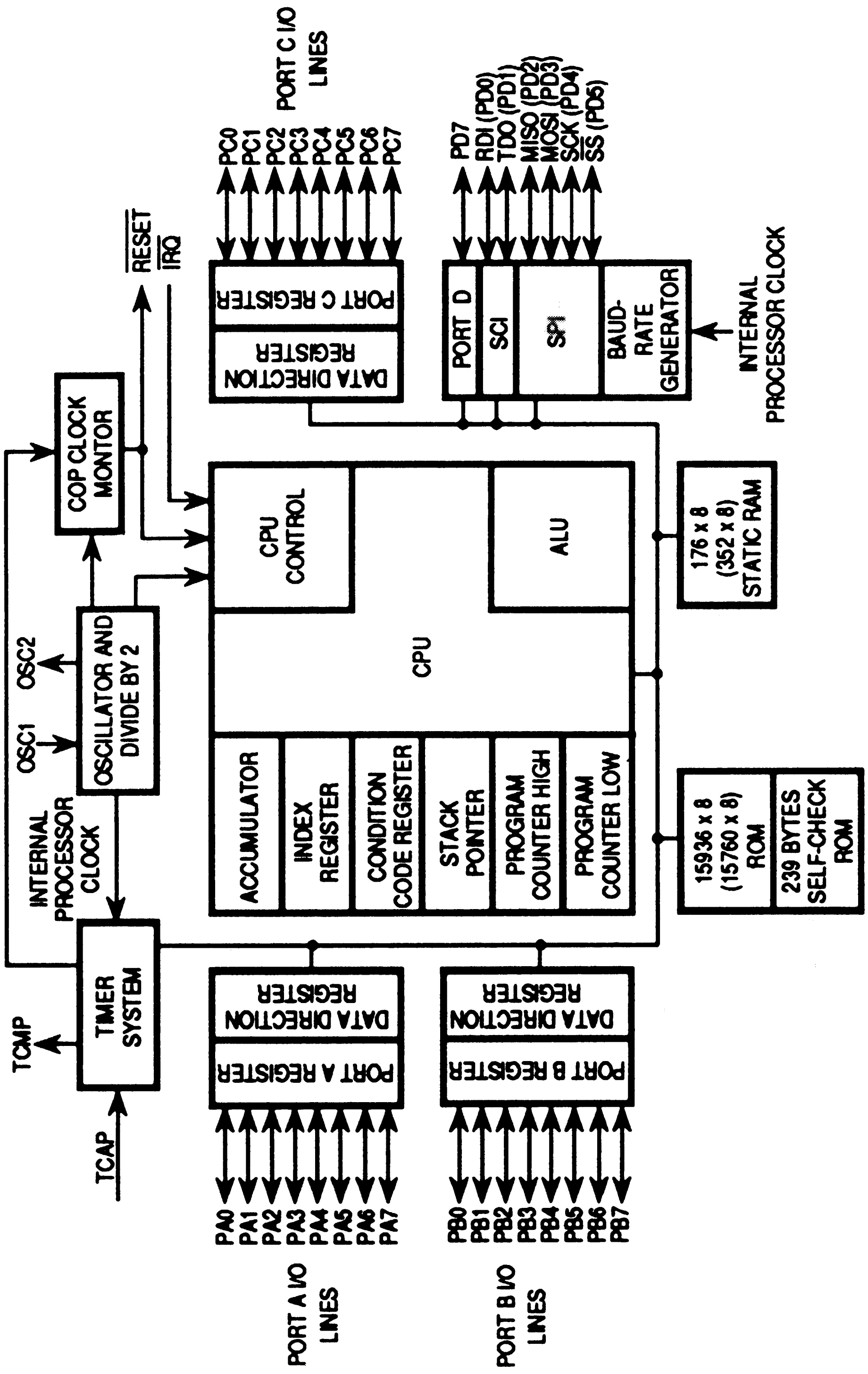
# MC68HC05C4
# MC68HCL05C4
# MC68HSC05C4
# BLOCK DIAGRAM



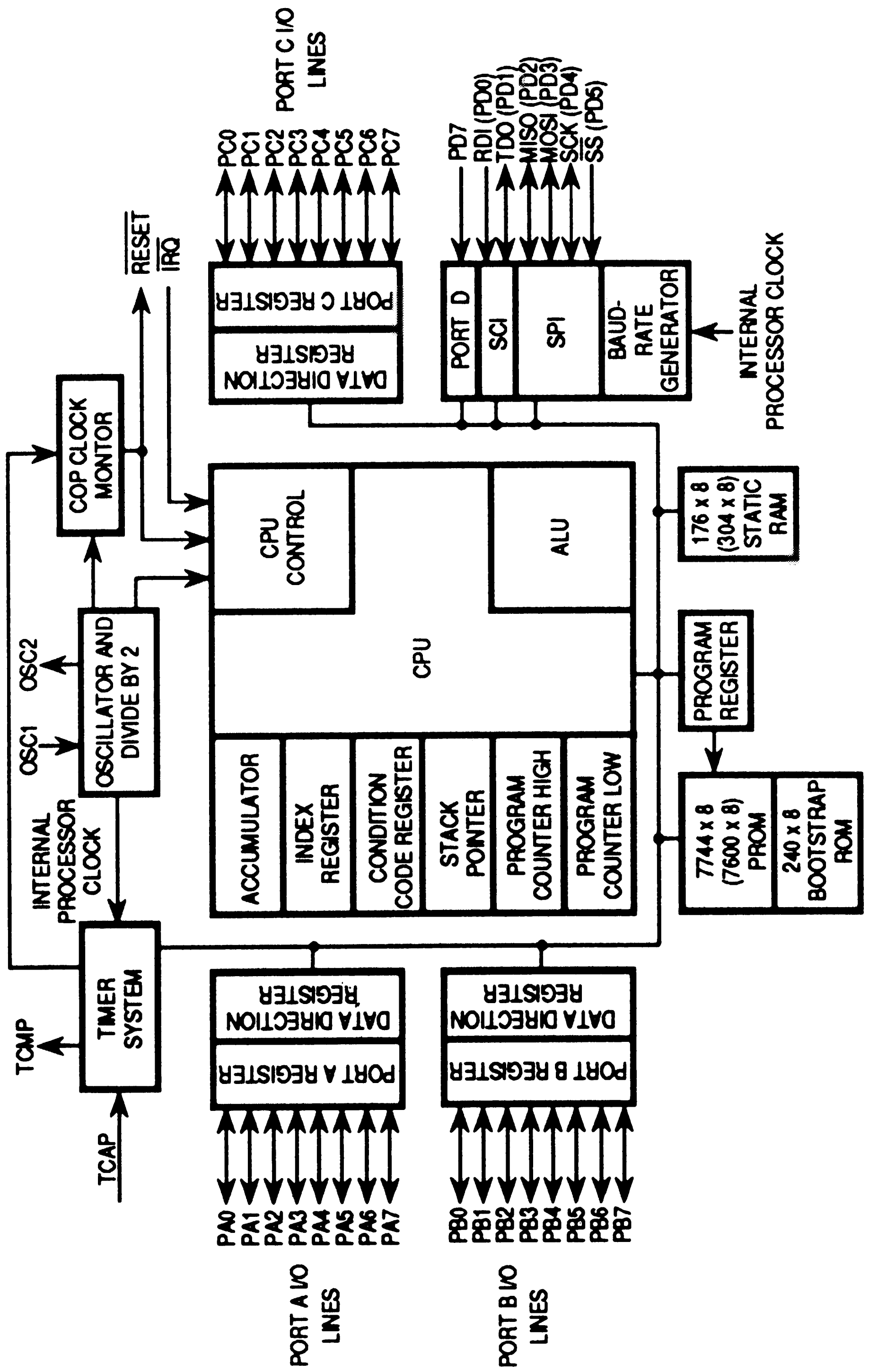
# MC68HC05C8 MC68HCL05C8 MC68HSC05C8 BLOCK DIAGRAM



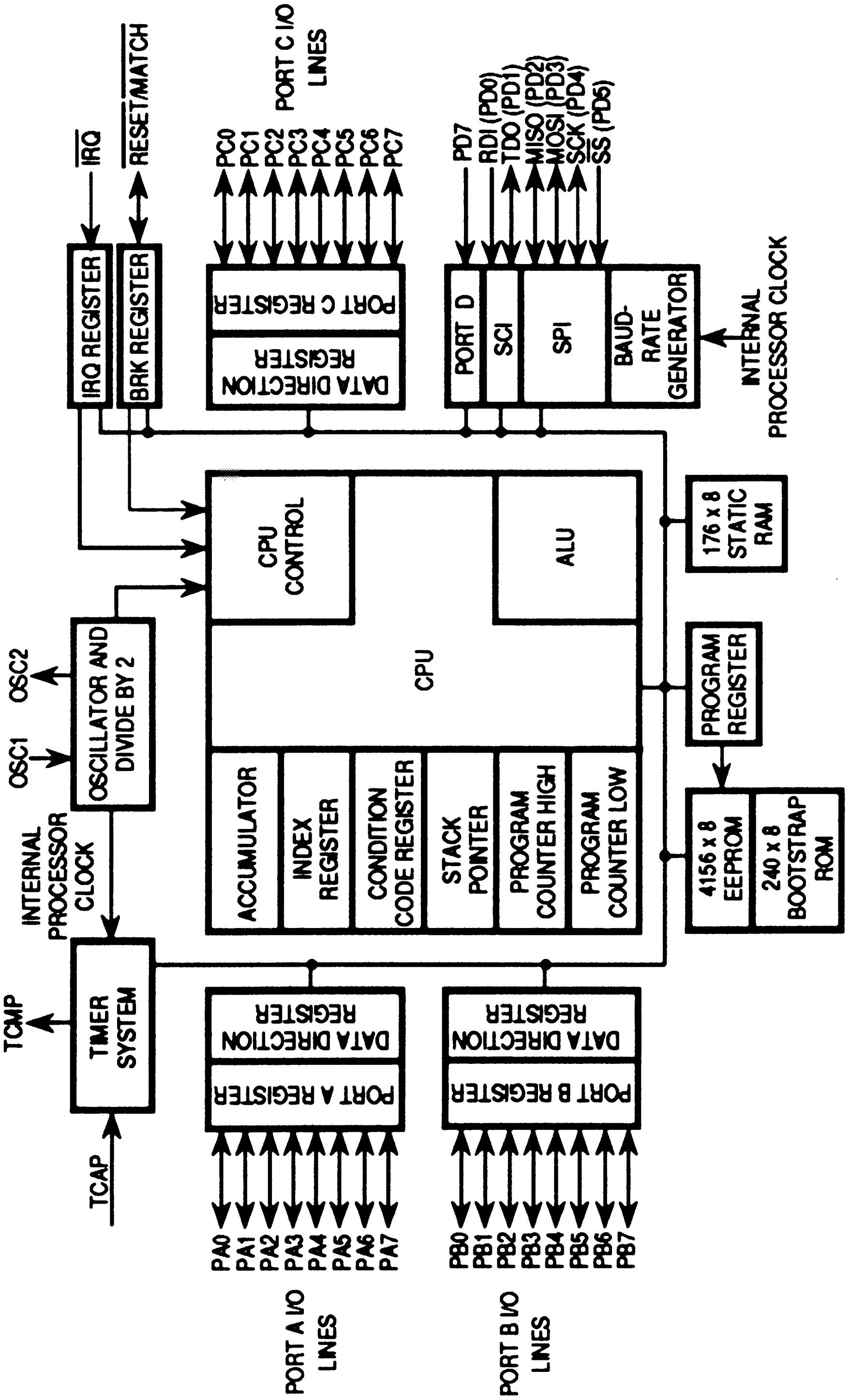
# MC68HC05C9
# BLOCK DIAGRAM



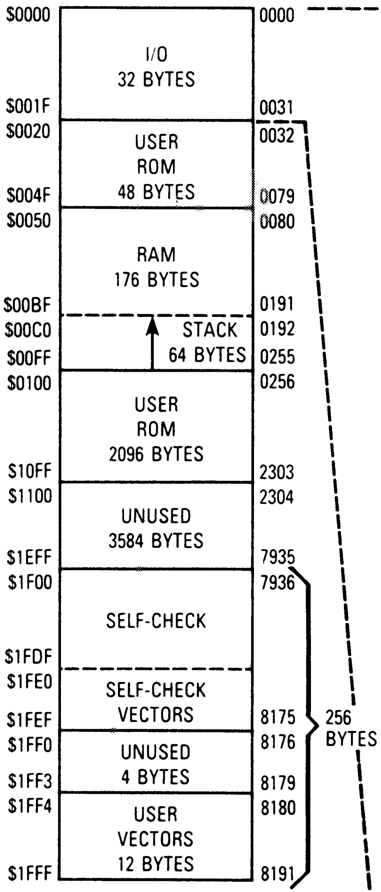
# MC68HC705C8 BLOCK DIAGRAM



# MC68HC805C4 BLOCK DIAGRAM

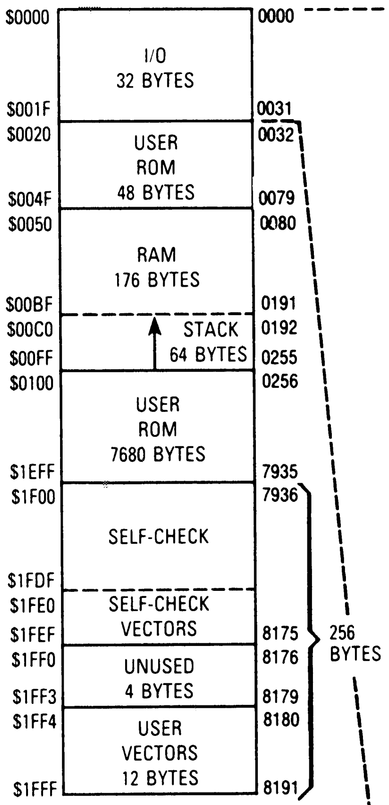


# MC68HC05C4
# MC68HCL05C4
# MC68HSC05C4
# MEMORY MAP



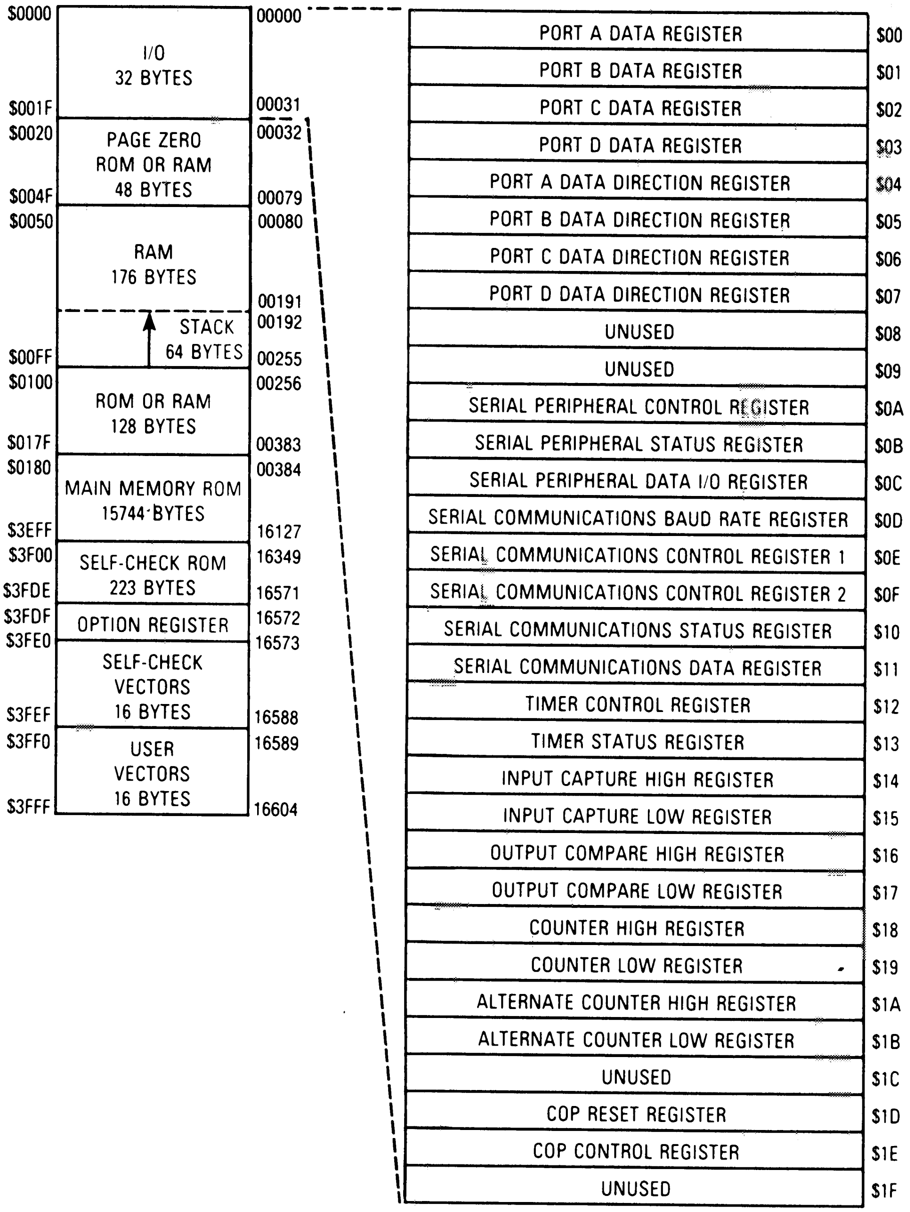
| | |
|---|---|
| PORT A DATA REGISTER | $00 |
| PORT B DATA REGISTER | $01 |
| PORT C DATA REGISTER | $02 |
| PORT D FIXED INPUT REGISTER | $03 |
| PORT A DATA DIRECTION REGISTER | $04 |
| PORT B DATA DIRECTION REGISTER | $05 |
| PORT C DATA DIRECTION REGISTER | $06 |
| UNUSED | $07 |
| UNUSED | $08 |
| UNUSED | $09 |
| SERIAL PERIPHERAL CONTROL REGISTER | $0A |
| SERIAL PERIPHERAL STATUS REGISTER | $0B |
| SERIAL PERIPHERAL DATA I/O REGISTER | $0C |
| SERIAL COMMUNICATIONS BAUD RATE REGISTER | $0D |
| SERIAL COMMUNICATIONS CONTROL REGISTER 1 | $0E |
| SERIAL COMMUNICATIONS CONTROL REGISTER 2 | $0F |
| SERIAL COMMUNICATIONS STATUS REGISTER | $10 |
| SERIAL COMMUNICATIONS DATA REGISTER | $11 |
| TIMER CONTROL REGISTER | $12 |
| TIMER STATUS REGISTER | $13 |
| INPUT CAPTURE HIGH REGISTER | $14 |
| INPUT CAPTURE LOW REGISTER | $15 |
| OUTPUT COMPARE HIGH REGISTER | $16 |
| OUTPUT COMPARE LOW REGISTER | $17 |
| COUNTER HIGH REGISTER | $18 |
| COUNTER LOW REGISTER | $19 |
| ALTERNATE COUNTER HIGH REGISTER | $1A |
| ALTERNATE COUNTER LOW REGISTER | $1B |
| UNUSED | $1C |
| UNUSED | $1D |
| UNUSED | $1E |
| UNUSED | $1F |

# MC68HC05C8
# MC68HCL05C8
# MC68HSC05C8
# MEMORY MAP



| | |
|---|---|
| PORT A DATA REGISTER | $00 |
| PORT B DATA REGISTER | $01 |
| PORT C DATA REGISTER | $02 |
| PORT D FIXED INPUT REGISTER | $03 |
| PORT A DATA DIRECTION REGISTER | $04 |
| PORT B DATA DIRECTION REGISTER | $05 |
| PORT C DATA DIRECTION REGISTER | $06 |
| UNUSED | $07 |
| UNUSED | $08 |
| UNUSED | $09 |
| SERIAL PERIPHERAL CONTROL REGISTER | $0A |
| SERIAL PERIPHERAL STATUS REGISTER | $0B |
| SERIAL PERIPHERAL DATA I/O REGISTER | $0C |
| SERIAL COMMUNICATIONS BAUD RATE REGISTER | $0D |
| SERIAL COMMUNICATIONS CONTROL REGISTER 1 | $0E |
| SERIAL COMMUNICATIONS CONTROL REGISTER 2 | $0F |
| SERIAL COMMUNICATIONS STATUS REGISTER | $10 |
| SERIAL COMMUNICATIONS DATA REGISTER | $11 |
| TIMER CONTROL REGISTER | $12 |
| TIMER STATUS REGISTER | $13 |
| INPUT CAPTURE HIGH REGISTER | $14 |
| INPUT CAPTURE LOW REGISTER | $15 |
| OUTPUT COMPARE HIGH REGISTER | $16 |
| OUTPUT COMPARE LOW REGISTER | $17 |
| COUNTER HIGH REGISTER | $18 |
| COUNTER LOW REGISTER | $19 |
| ALTERNATE COUNTER HIGH REGISTER | $1A |
| ALTERNATE COUNTER LOW REGISTER | $1B |
| UNUSED | $1C |
| UNUSED | $1D |
| UNUSED | $1E |
| UNUSED | $1F |

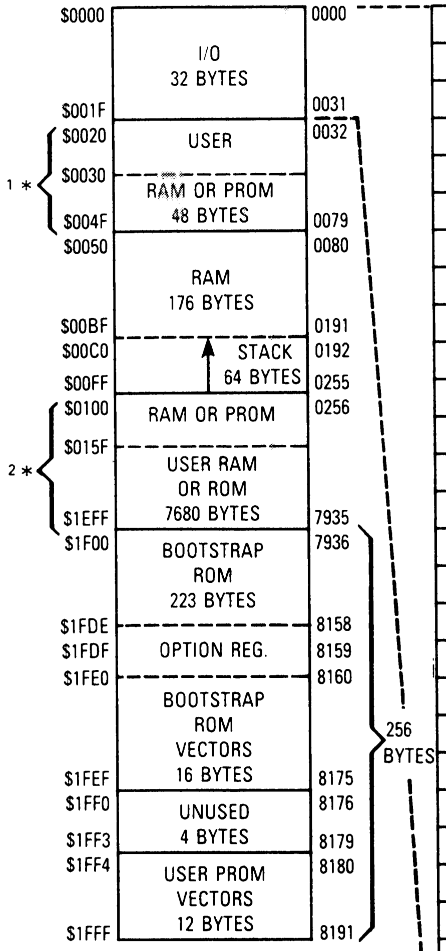
# MC68HC05C9
# MEMORY MAP



| | | | |
|---|---|---|---|
| $0000 | I/O<br>32 BYTES | 00000 | |
| $001F | | 00031 | |
| $0020 | PAGE ZERO<br>ROM OR RAM<br>48 BYTES | 00032 | |
| $004F | | 00079 | |
| $0050 | RAM<br>176 BYTES | 00080 | |
| | | 00191 | |
| | STACK<br>64 BYTES | 00192 | |
| $00FF | | 00255 | |
| $0100 | ROM OR RAM<br>128 BYTES | 00256 | |
| $017F | | 00383 | |
| $0180 | MAIN MEMORY ROM<br>15744 BYTES | 00384 | |
| $3EFF | | 16127 | |
| $3F00 | SELF-CHECK ROM<br>223 BYTES | 16349 | |
| $3FDE | | 16571 | |
| $3FDF | OPTION REGISTER | 16572 | |
| $3FE0 | SELF-CHECK<br>VECTORS<br>16 BYTES | 16573 | |
| $3FEF | | 16588 | |
| $3FF0 | USER<br>VECTORS<br>16 BYTES | 16589 | |
| $3FFF | | 16604 | |

| | |
|---|---|
| PORT A DATA REGISTER | $00 |
| PORT B DATA REGISTER | $01 |
| PORT C DATA REGISTER | $02 |
| PORT D DATA REGISTER | $03 |
| PORT A DATA DIRECTION REGISTER | $04 |
| PORT B DATA DIRECTION REGISTER | $05 |
| PORT C DATA DIRECTION REGISTER | $06 |
| PORT D DATA DIRECTION REGISTER | $07 |
| UNUSED | $08 |
| UNUSED | $09 |
| SERIAL PERIPHERAL CONTROL REGISTER | $0A |
| SERIAL PERIPHERAL STATUS REGISTER | $0B |
| SERIAL PERIPHERAL DATA I/O REGISTER | $0C |
| SERIAL COMMUNICATIONS BAUD RATE REGISTER | $0D |
| SERIAL COMMUNICATIONS CONTROL REGISTER 1 | $0E |
| SERIAL COMMUNICATIONS CONTROL REGISTER 2 | $0F |
| SERIAL COMMUNICATIONS STATUS REGISTER | $10 |
| SERIAL COMMUNICATIONS DATA REGISTER | $11 |
| TIMER CONTROL REGISTER | $12 |
| TIMER STATUS REGISTER | $13 |
| INPUT CAPTURE HIGH REGISTER | $14 |
| INPUT CAPTURE LOW REGISTER | $15 |
| OUTPUT COMPARE HIGH REGISTER | $16 |
| OUTPUT COMPARE LOW REGISTER | $17 |
| COUNTER HIGH REGISTER | $18 |
| COUNTER LOW REGISTER | $19 |
| ALTERNATE COUNTER HIGH REGISTER | $1A |
| ALTERNATE COUNTER LOW REGISTER | $1B |
| UNUSED | $1C |
| COP RESET REGISTER | $1D |
| COP CONTROL REGISTER | $1E |
| UNUSED | $1F |

# MC68HC705C8
# MEMORY MAP



| | |
|---|---|
| PORT A DATA REGISTER | $00 |
| PORT B DATA REGISTER | $01 |
| PORT C DATA REGISTER | $02 |
| PORT D FIXED INPUT REGISTER | $03 |
| PORT A DATA DIRECTION REGISTER | $04 |
| PORT B DATA DIRECTION REGISTER | $05 |
| PORT C DATA DIRECTION REGISTER | $06 |
| UNUSED | $07 |
| UNUSED | $08 |
| UNUSED | $09 |
| SERIAL PERIPHERAL CONTROL REGISTER | $0A |
| SERIAL PERIPHERAL STATUS REGISTER | $0B |
| SERIAL PERIPHERAL DATA I/O REGISTER | $0C |
| SERIAL COMMUNICATIONS BAUD RATE REGISTER | $0D |
| SERIAL COMMUNICATIONS CONTROL REGISTER 1 | $0E |
| SERIAL COMMUNICATIONS CONTROL REGISTER 2 | $0F |
| SERIAL COMMUNICATIONS STATUS REGISTER | $10 |
| SERIAL COMMUNICATIONS DATA REGISTER | $11 |
| TIMER CONTROL REGISTER | $12 |
| TIMER STATUS REGISTER | $13 |
| INPUT CAPTURE HIGH REGISTER | $14 |
| INPUT CAPTURE LOW REGISTER | $15 |
| OUTPUT COMPARE HIGH REGISTER | $16 |
| OUTPUT COMPARE LOW REGISTER | $17 |
| COUNTER HIGH REGISTER | $18 |
| COUNTER LOW REGISTER | $19 |
| ALTERNATE COUNTER HIGH REGISTER | $1A |
| ALTERNATE COUNTER LOW REGISTER | $1B |
| PROGRAM REGISTER | $1C |
| COP RESET REGISTER | $1D |
| COP CONTROL REGISTER | $1E |
| UNUSED | $1F |

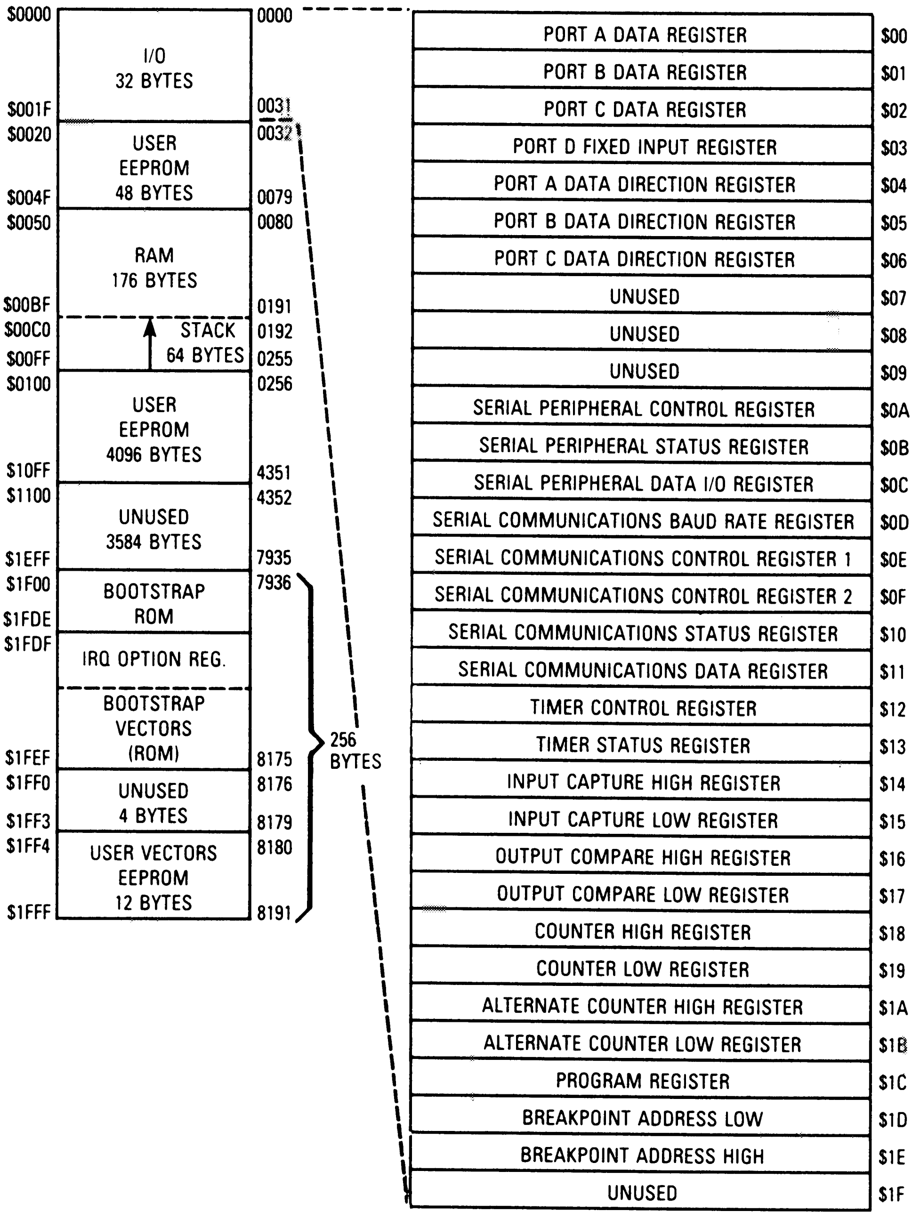
(Option Register $1FDF RAM1 = 0 and RAM0 = 0) (POR or Master Reset)

*The nature of this memory block (RAM or PROM) is controlled by bits RAM0 and RAM1 of the Option Register ($1FDF).

1. RAM0 0 — 48 Bytes User PROM
   RAM0 1 — 32 Bytes RAM with 16 Bytes Unused

2. RAM1 0 — 7680 Bytes User PROM
   RAM1 1 — 7584 Bytes User PROM and 96 Bytes of RAM

# MC68HC805C4 MEMORY MAP

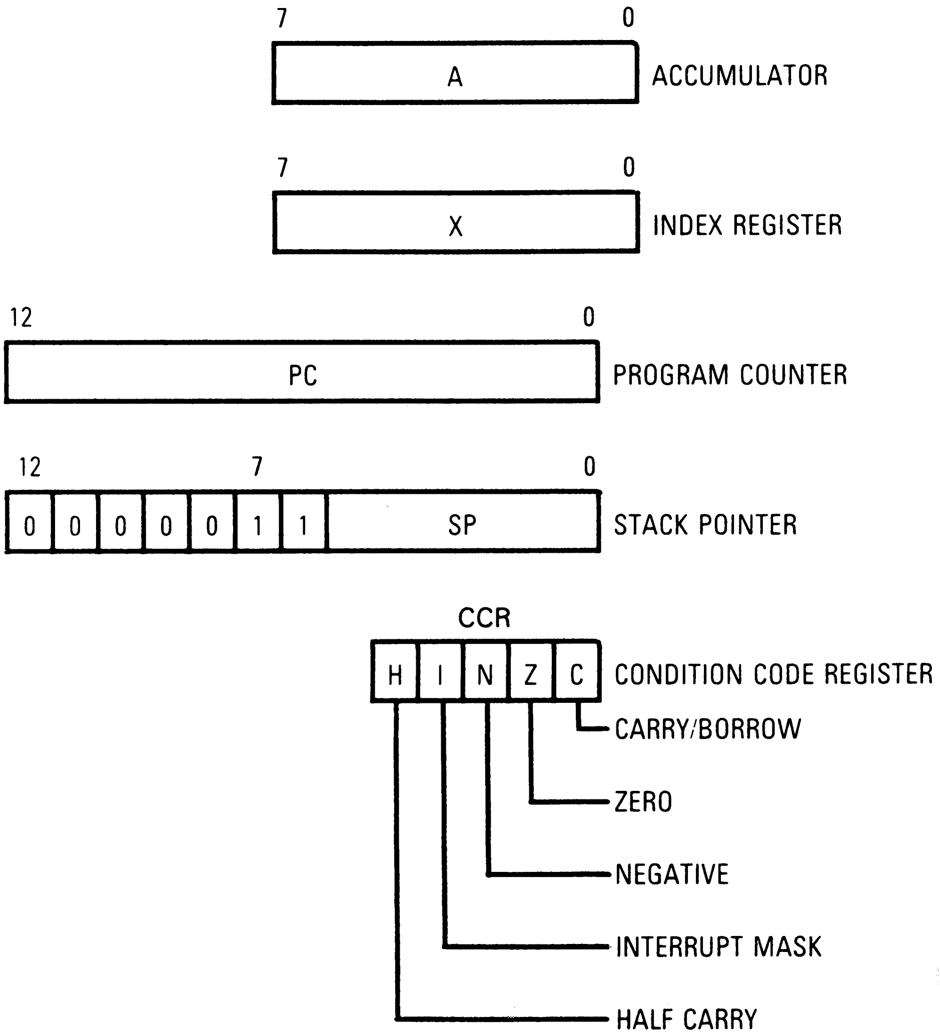


| | | |
|---|---|---|
| $0000 | I/O<br>32 BYTES | 0000 |
| $001F | | 0031 |
| $0020 | USER<br>EEPROM<br>48 BYTES | 0032 |
| $004F | | 0079 |
| $0050 | RAM<br>176 BYTES | 0080 |
| $00BF | | 0191 |
| $00C0 | STACK<br>64 BYTES | 0192 |
| $00FF | | 0255 |
| $0100 | USER<br>EEPROM<br>4096 BYTES | 0256 |
| $10FF | | 4351 |
| $1100 | UNUSED<br>3584 BYTES | 4352 |
| $1EFF | | 7935 |
| $1F00 | BOOTSTRAP<br>ROM | 7936 |
| $1FDE | | |
| $1FDF | IRQ OPTION REG. | |
| | BOOTSTRAP<br>VECTORS<br>(ROM) | |
| $1FEF | | 8175 |
| $1FF0 | UNUSED<br>4 BYTES | 8176 |
| $1FF3 | | 8179 |
| $1FF4 | USER VECTORS<br>EEPROM<br>12 BYTES | 8180 |
| $1FFF | | 8191 |

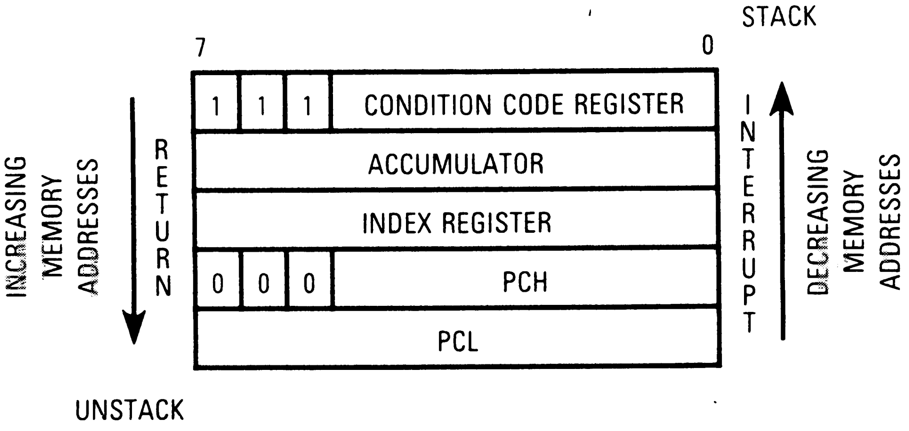
| | |
|---|---|
| PORT A DATA REGISTER | $00 |
| PORT B DATA REGISTER | $01 |
| PORT C DATA REGISTER | $02 |
| PORT D FIXED INPUT REGISTER | $03 |
| PORT A DATA DIRECTION REGISTER | $04 |
| PORT B DATA DIRECTION REGISTER | $05 |
| PORT C DATA DIRECTION REGISTER | $06 |
| UNUSED | $07 |
| UNUSED | $08 |
| UNUSED | $09 |
| SERIAL PERIPHERAL CONTROL REGISTER | $0A |
| SERIAL PERIPHERAL STATUS REGISTER | $0B |
| SERIAL PERIPHERAL DATA I/O REGISTER | $0C |
| SERIAL COMMUNICATIONS BAUD RATE REGISTER | $0D |
| SERIAL COMMUNICATIONS CONTROL REGISTER 1 | $0E |
| SERIAL COMMUNICATIONS CONTROL REGISTER 2 | $0F |
| SERIAL COMMUNICATIONS STATUS REGISTER | $10 |
| SERIAL COMMUNICATIONS DATA REGISTER | $11 |
| TIMER CONTROL REGISTER | $12 |
| TIMER STATUS REGISTER | $13 |
| INPUT CAPTURE HIGH REGISTER | $14 |
| INPUT CAPTURE LOW REGISTER | $15 |
| OUTPUT COMPARE HIGH REGISTER | $16 |
| OUTPUT COMPARE LOW REGISTER | $17 |
| COUNTER HIGH REGISTER | $18 |
| COUNTER LOW REGISTER | $19 |
| ALTERNATE COUNTER HIGH REGISTER | $1A |
| ALTERNATE COUNTER LOW REGISTER | $1B |
| PROGRAM REGISTER | $1C |
| BREAKPOINT ADDRESS LOW | $1D |
| BREAKPOINT ADDRESS HIGH | $1E |
| UNUSED | $1F |

# PROGRAMMING MODEL
# INTERRUPT STACKING ORDER

## PROGRAMMING MODEL



## INTERRUPT STACKING ORDER



NOTE: Since the Stack Pointer decrements during pushes, the PCL is stacked first, followed by PCH, etc. Pulling from the stack is in the reverse order.

# REGISTER AND CONTROL BIT SUMMARY

| | BIT 7 | 6 | 5 | 4 | 3 | 2 | 1 | BIT 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $00 | PA7 | PA6 | PA5 | PA4 | PA3 | PA2 | PA1 | PA0 | PORTA |
| $01 | PB7 | PB6 | PB5 | PB4 | PB3 | PB2 | PB1 | PB0 | PORTB |
| $02 | PC7 | PC6 | PC5 | PC4 | PC3 | PC2 | PC1 | PC0 | PORT C |
| $03 | PD7/* | | PD5/* | PD4/* | PD3/* | PD2/* | PD1/* | PD0/* | PORTD |
| $04 | DDA7 | DDA6 | DDA5 | DDA4 | DDA3 | DDA2 | DDA1 | DDA0 | DDRA |
| $05 | DDB7 | DDB6 | DDB5 | DDB4 | DDB3 | DDB2 | DDB1 | DDB0 | DDRB |
| $06 | DDC7 | DDC6 | DDC5 | DDC4 | DDC3 | DDC2 | DDC1 | DDC0 | DDRC |
| $07 | DDD7 | | DDD5 | DDD4 | DDD3 | DDD2 | DDD1 | DDD0 | DDRD† |
| $08 | | | | | | | | | UNUSED |
| $09 | | | | | | | | | UNUSED |
| $0A | SPIE | SPE | DWOM† | MSTR | CPOL | CPHA | SPR1 | SPR0 | SPCR |
| $0B | SPIF | WCOL | | MODF | | | | | SPSR |
| $0C | SPD7 | SPD6 | SPD5 | SPD4 | SPD3 | SPD2 | SPD1 | SPD0 | SPDR |
| $0D | | | SCP1 | SCP0 | | SCR2 | SCR1 | SCR0 | BAUD |
| $0E | R8 | T8 | | M | WAKE | | | | SCCR1 |
| $0F | TIE | TCIE | RIE | ILIE | TE | RE | RWU | SBK | SCCR2 |
| $10 | TDRE | TC | RDRF | IDLE | OR | NF | FE | | SCSR |
| $11 | SCD7 | SCD6 | SCD5 | SCD4 | SCD3 | SCD2 | SCD1 | SCD0 | SCDAT |
| $12 | ICIE | OCIE | TOIE | 0 | 0 | 0 | IEDG | OLVL | TCR |
| $13 | ICF | OCF | TOF | 0 | 0 | 0 | 0 | 0 | TSR |
| $14 | ICH7 | ICH6 | ICH5 | ICH4 | ICH3 | ICH2 | ICH1 | ICH0 | ICHR |
| $15 | ICL7 | ICL6 | ICL5 | ICL4 | ICL3 | ICL2 | ICL1 | ICL0 | ICLR |
| $16 | OCH7 | OCH6 | OCH5 | OCH4 | OCH3 | OCH2 | OCH1 | OCH0 | OCHR |
| $17 | OCL7 | OCL6 | OCL5 | OCL4 | OCL3 | OCL2 | OCL1 | OCL0 | OCLR |
| $18 | CH7 | CH6 | CH5 | CH4 | CH3 | CH2 | CH1 | CH0 | CHR |
| $19 | CL7 | CL6 | CL5 | CL4 | CL3 | CL2 | CL1 | CL0 | CLR |
| $1A | ACH7 | ACH6 | ACH5 | ACH4 | ACH3 | ACH2 | ACH1 | ACH0 | ACHR |
| $1B | ACL7 | ACL6 | ACL5 | ACL4 | ACL3 | ACL2 | ACL1 | ACL0 | ACLR |
| $1C | | | | | | | | | UNUSED |
| $1D | | | | | | | | | UNUSED |
| $1E | | | | | | | | | UNUSED |
| $1F | | | | | | | | | UNUSED |

*Denotes fixed input port, see following page.

†MC68HC05C9 only

# REGISTER AND CONTROL BIT SUMMARY

|  | BIT 7 | 6 | 5 | 4 | 3 | 2 | 1 | BIT 0 |  |
|---|---|---|---|---|---|---|---|---|---|
| $03 | PD7 |  | PD5 | PD4 | PD3 | PD2 | PD1 | PD0 | PORTD |
|  |  |  | $\overline{SS}$ | SCK | MOSI | MISO | TDO | RDI |  |

(PORT D FIXED INPUT REGISTER)

|  | BIT 7 | 6 | 5 | 4 | 3 | 2 | 1 | BIT 0 |  |
|---|---|---|---|---|---|---|---|---|---|
| $07 | DDD7 |  | DDD5 | DDD4 | DDD3 | DDD2 | DDD1 | DDD0 |  |

|  |  |  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|---|
| $1D | CR7 | CR6 | CR5 | CR4 | CR3 | CR2 | CR1 | CR0 | COPRR |
| $1E | R7 | R6 | R5 | COPF | CME | COPE | CM1 | CM0 | COPCR |
| $3FDF | RAM0 | RAM1 |  |  |  |  | IRQ |  | OPTREG |

(MC68HC05C9 ONLY)

|  | BIT 7 | 6 | 5 | 4 | 3 | 2 | 1 | BIT 0 |  |
|---|---|---|---|---|---|---|---|---|---|
| $1C | 0 | 0 | 0 | 0 | 0 | LAT | 0 | PGM | PR |
| $1D | CR7 | CR6 | CR5 | CR4 | CR3 | CR2 | CR1 | CR0 | COPRR |
| $1E | 0 | 0 | 0 | COPF | CME | COPE | CM1 | CM0 | COPCR |
| $1F |  |  |  |  |  |  |  |  | UNUSED |

|  |  |  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|---|
| $1FDF | RAM0 | RAM1 | 0 | 0 | SEC |  | IRQ | 0 | IRQOR |

(MC68HC705C8 ONLY)

|  | BIT 7 | 6 | 5 | 4 | 3 | 2 | 1 | BIT 0 |  |
|---|---|---|---|---|---|---|---|---|---|
| $1C | 0 | CPEN | 0 | 0 | ERASE | LATA | LATB | EEPGM | PR |
| $1D | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 | ARL |
| $1E | 0 | 0 | MATCH | A12 | A11 | A10 | A9 | A8 | ARH |
| $1F |  |  |  |  |  |  |  |  | UNUSED |

|  |  |  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|---|
| $1FDF | 0 | 0 | 0 | 0 | 0 | 0 | IRQ | 0 | IRQOR |

(MC68HC805C4 ONLY)

# ACHR

## Alternate Counter High Register (ACHR) $1A

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ACH7 | ACH6 | ACH5 | ACH4 | ACH3 | ACH2 | ACH1 | ACH0 |
| RESET | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |

# ACLR

## Alternate Counter Low Register (ACLR) $1B

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ACL7 | ACL6 | ACL5 | ACL4 | ACL3 | ACL2 | ACL1 | ACL0 |
| RESET | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |

# ARH

## (MC68HC805C4 ONLY)

## Hardware Breakpoint Register High (ARH) $1E

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | MATCH | A12 | A11 | A10 | A9 | A8 |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

MATCH — An instruction with the same address as that in the breakpoint register was fetched.
1 = Breakpoint enabled
0 = Breakpoint disabled
A12–A8 — Breakpoint address bits A12–A8

# ARL

## (MC68HC805C4 ONLY)

## Hardware Breakpoint Register Low (ARL) $1D

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

## Baud Rate Register (BAUD) $0D

The baud rate register is used to select the SCI transmitter and receiver baud rate. SCP0 and SCP1 prescaler bits are used in conjunction with the SCR2–SCR0 baud rates to provide multiple baud rate combinations for a given crystal frequency. Bits 3, 6, and 7 always read zero.

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
|  | — | — | SCP1 | SCP0 | — | SCR2 | SCR1 | SCR0 |
| RESET | — | — | 0 | 0 | — | U | U | U |

SCP0 — SCI Prescaler Bit 0
SCP1 — SCI Prescaler Bit 1

Two prescaler bits are used to increase the range of standard baud rates controlled by the SCR2–SCR0 bits. Prescaler internal processor clock division versus bits levels are listed in Table 1.

SCR0 — SCI Baud Rate Bit 0
SCR1 — SCI Baud Rate Bit 1
SCR2 — SCI Baud Rate Bit 2

Three baud rate bits are used to select the baud rates of the SCI transmitter and SCI receiver. Baud rates versus bit levels are listed in Table 2.

Tables 1 and 2 tabulate the divide chain used to obtain the baud rate clock (transmit clock). The actual divider chain is controlled by the combined SCP1–SCP0 and SCR2–SCR0 bits in the baud rate register. All divided frequencies shown in Table 1 represent the final baud rate resulting from the internal processor clock division shown in the divided by column only (prescaler division only). Table 2 lists the prescaler output divided by the action of the SCI select bits (SCR2–SCR0). For example, assume that a 9600 Hz baud rate is required with a 2.4576 MHz external crystal. In this case, the prescaler bits (SCP1–SCP0) could be configured as a divided-by-two. Using the same crystal, the 9600 baud rate can be obained with a prescaler divided-by-one and the SCR2–SCR0 bits configured for a divide-by-eight.

## Table 1. Prescaler Highest Baud Rate Frequency Output

| SCP Bit | | Clock* Divided By | Crystal Frequency MHz | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | | 8.0 | 4.194304 | 4.0 | 2.4576 | 2.0 | 1.8432 |
| 0 | 0 | 1 | 250.00 kHz | 131.072 kHz | 125.000 kHz | 76.80 kHz | 62.60 kHz | 57.60 kHz |
| 0 | 1 | 3 | 83.332 kHz | 43.691 kHz | 41.666 kHz | 25.60 kHz | 20.833 kHz | 19.20 kHz |
| 1 | 0 | 4 | 62.500 kHz | 32.768 kHz | 31.250 kHz | 19.20 kHz | 15.625 kHz | 14.40 kHz |
| 1 | 1 | 13 | 19.200 kHz | 10.082 kHz | 9600 Hz | 5.907 kHz | 4800 Hz | 4430 Hz |

*Refers to the internal processor clock.

NOTE: The divided frequencies shown in Table 1 represent baud rates which are the highest transmit baud rate (Tx) that can be obtained by a specific crystal frequency and only using the prescaler division. Lower baud rates may be obtained by providing a further division using the SCI rate select bits as shown below for some representative prescaler outputs.

## Table 2. Transmit Baud Rate Output for a Given Prescaler Output

| SCR Bits | | | Divided By | Representative Highest Prescaler Baud Rate Output | | | | |
|---|---|---|---|---|---|---|---|---|
| 2 | 1 | 0 | | 131.072 kHz | 32.768 kHz | 76.80 kHz | 19.20 kHz | 9600 Hz |
| 0 | 0 | 0 | 1 | 131.072 kHz | 32.768 kHz | 76.80 kHz | 19.20 kHz | 9600 Hz |
| 0 | 0 | 1 | 2 | 65.536 kHz | 16.384 kHz | 38.40 kHz | 9600 Hz | 4800 Hz |
| 0 | 1 | 0 | 4 | 32.768 kHz | 8.192 kHz | 19.20 kHz | 4800 Hz | 2400 Hz |
| 0 | 1 | 1 | 8 | 16.384 kHz | 4.096 kHz | 9600 Hz | 2400 Hz | 1200 Hz |
| 1 | 0 | 0 | 16 | 8.192 kHz | 2.048 kHz | 4800 Hz | 1200 Hz | 600 Hz |
| 1 | 0 | 1 | 32 | 4.096 kHz | 1.024 kHz | 2400 Hz | 600 Hz | 300 Hz |
| 1 | 1 | 0 | 64 | 2.048 kHz | 512 Hz | 1200 Hz | 300 Hz | 150 Hz |
| 1 | 1 | 1 | 128 | 1.024 kHz | 256 Hz | 600 Hz | 150 Hz | 75 Hz |

NOTE: Table 2 illustrates how the SCI select bits can be used to provide lower transmitter baud rate by further dividing the prescaler output frequency. The five examples are only representative samples. In all cases, the baud rates shown are transmit baud rates (transmit clock) and the receive clock is 16 times higher in frequency than the actual baud rate.

## Counter High Register (CHR) $18

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CH7 | CH6 | CH5 | CH4 | CH3 | CH2 | CH1 | CH0 |
| RESET | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

CLR

## Counter Low Register (CLR) $19

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CL7 | CL6 | CL5 | CL4 | CL3 | CL2 | CL1 | CL0 |
| RESET | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |

## (MC68HC705C8 ONLY)

## COP Control Register (COPCR) $1E

The COPCR shown below is used to control the COP watchdog timer and clock monitor functions.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | COPF | CME | COPE | CM1 | CM0 |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

COPF—Computer Operating Properly
1=COP or clock monitor reset has occurred
0=No COP or clock monitor reset has occurred

CME — Clock Monitor Enable
1=Clock monitor enabled
0=Clock monitor disabled

COPE — Computer Operating Properly Enable
1=COP timeout enabled
0=COP timeout disabled

CM1 — Computer Operating Properly Mode 1
Used in conjunction with CMO0 to establish the COP timeout period. CM1 can be read and set anytime, but is cleared only by reset. See Table 3.

CM0 — Computer Operating Properly Mode 0
Used in conjunction with CM1 to establish the COP timeout period. CM0 can be read and set anytime, but is cleared only by reset. See Table 3.

Bits 7–5 — Not used*
Always read zero

*In the MC68HC05C9, these bits (R7–R5) are reserved factory test bits.

### Table 3. COP Timeout Period

| CM1 | CM0 | $E/2^{15}$ Divided By | XTAL = 4.0 MHz, E = 2.0 MHz | XTAL = 3.5795 MHz, E = 1.7897 MHz | XTAL = 2.0 MHz, E = 1.0 MHz | XTAL = 1.0 MHz, E = 0.5 MHz |
|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 16.38 ms | 18.31 ms | 32.77 ms | 65.54 ms |
| 0 | 1 | 4 | 65.54 ms | 73.24 ms | 131.07 ms | 262.14 ms |
| 1 | 0 | 16 | 262.95 ms | 292.95 ms | 524.29 ms | 1.048 s |
| 1 | 1 | 64 | 1.048 s | 1.172 s | 2.097 s | 4.194 s |

**(MC68HC705C8 AND MC68HC05C9 ONLY)**

## COP Reset Register (COPRR) $1D

The COPRR shown below is used to control the COP watchdog timer and clock monitor functions.

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  |  |  |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

The sequence required to reset the COP timer is as follows:

Write $55 to the COP reset register.

Write $AA to the COP reset register.

Both write operations must occur in the order listed, but any number of instructions may be executed between the two write operations. The elapsed time between software resets must not be greater than the COP timeout period. Reading the COP reset register does not return valid data and does not affect the watchdog timer.

### (MC68HC705C8 AND MC68HC05C9 ONLY)

## COP Reset Register (COPRR) $1D

The COPRR shown below is used to control the COP watchdog timer and clock monitor functions.

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  |  |  |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

The sequence required to reset the COP time is as follows:

Write $55 to the COP reset register.

Write $AA to the COP reset register.

Both write operations must occur in the order listed, but any number of instructions may be executed between the two write operations. The elapsed time between software resets must not be greater than the COP timeout period. Reading the COP reset register does not return valid data and does not affect the watchdog timer.

### (MC68HC705C8 AND MC68HC05C9 ONLY)

## DDRA

### Data Direction Register A (DDRA) $04

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

DDA7–DDA0 — 0 = Inputs
1 = Outputs

## DDRB

### Data Direction Register B (DDRB) $05

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

DDB7–DDB0 — 0 = Inputs
1 = Outputs

## DDRC

### Data Direction Register C (DDRC) $06

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

DDC7–DDC0 — 0 = Inputs
1 = Outputs

# DDRD

### Data Direction Register D (DDRD) $07

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DDD7 | — | DDD5 | DDD4 | DDD3 | DDD2 | DDD1 | DDD0 |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

DDD5–DDD0 — 0 = Inputs
1 = Outputs

Bits 7,6 — Not used.

# ICHR

### Input Capture High Register (ICHR) $14

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ICH7 | ICH6 | ICH5 | ICH4 | ICH3 | ICH2 | ICH1 | ICH0 |
| RESET | U | U | U | U | U | U | U | U |

# ICLR

### Input Capture Low Register (ICLR) $15

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ICL7 | ICL6 | ICL5 | ICL4 | ICL3 | ICL2 | ICL1 | ICL0 |
| RESET | U | U | U | U | U | U | U | U |

## (MC68HC705C8 ONLY)

## Option Register (IRQOR) $1FDF

The option register is used to select the $\overline{\text{IRQ}}$ sensitivity, enable the PROM security, and select the memory configuration.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RAM0 | RAM1 | 0 | 0 | SEC | — | IRQ | 0 |
| RESET | 0 | 0 | 0 | 0 | U | — | 1 | 0 |

RAM0—Random Access Memory Control Bit 0

1 = Maps 32 bytes of RAM into page zero starting at address $0030. Addresses from $0020 to $0030 are reserved. This replaces 48 bytes of PROM that were used at these locations. This bit can be read or written at any time, allowing memory configuration to be changed during program execution.

0 = Provides 48 bytes of PROM at location $0030.

RAM1—Random Access Memory Control Bit 1

1 = Maps 96 bytes of RAM into page zero starting at address $0100. This replaces 96 bytes of PROM that were used at these locations. This bit can be read or written at any time, allowing memory configuraton to be changed during program execution.

0 = Provides 96 bytes of PROM at location $0100.

SEC—Security

1 = Bootloader disabled, MCU operates only in single-chip mode.

0 = Security off, bootloader enabled, expanded mode enabled.

IRQ—Interrupt Request Bit Sensitivity

1 = $\overline{\text{IRQ}}$ pin is both negative edge- and level-sensitive.

0 = $\overline{\text{IRQ}}$ pin is negative edge-sensitive only.

$\overline{\text{IRQ}}$ is set only by reset, but can be cleared by software. This can only be written once.

Bit 0, 4, 5

Always read zero.

Bit 2

Can be either one or zero.

## (MC68HC05C9 ONLY)

## Option Register (IRQOR) $3FDF

The option register is used to select the $\overline{\text{IRQ}}$ sensitivity, enable the ROM security, and select the memory configuration.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RAM0 | RAM1 | 0 | 0 | 0 | 0 | IRQ | 0 |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

RAM0 — Random Access Memory Control Bit 0

1 = Maps 32 bytes of RAM into page zero starting at address $0020. This replaces 48 bytes of ROM that were used at these locations. This bit can be read or written at any time, allowing memory configuration to be changed during program execution.

0 = Provides 48 bytes of ROM at location $0020.

RAM1 — Random Access Memory Control Bit 1

1 = Maps 128 bytes of RAM into page zero starting at address $0100. This replaces 128 bytes of ROM that were used at these locations. This bit can be read or written at any time, allowing memory configuraton to be changed during program execution.

0 = Provides 128 bytes of ROM at location $0100.

IRQ — Interrupt Request Bit Sensitivity

1 = $\overline{\text{IRQ}}$ pin is both negative edge- and level-sensitive.

0 = $\overline{\text{IRQ}}$ pin is negative edge-sensitive only.

$\overline{\text{IRQ}}$ is set only by reset, but can be cleared by software. This can only be written once.

Bit 0, 2, 3, 4, 5

Always read zero.

## IRQOR

### (MC68HC805C4 ONLY)
### IRQ Option Register (IRQOR) $1FDF

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
|  | 0 | 0 | 0 | 0 | 0 | 0 | IRQ | 0 |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

Bits 7–2, 0 — Not used
Always read zero.

IRQ — Interrupt Request Bit Sensitivity
1 = $\overline{\text{IRQ}}$ pin is both negative edge- and level-sensitive.
0 = $\overline{\text{IRQ}}$ pin is negative edge-sensitive only.
IRQ is set only by reset, but can be cleared by software.

## OCHR

### Output Compare High Register (OCHR) $16

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
|  | OCH7 | OCH6 | OCH5 | OCH4 | OCH3 | OCH2 | OCH1 | OCH0 |
| RESET | U | U | U | U | U | U | U | U |

## OCLR

### Output Compare Low Register (OCLR) $17

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
|  | OCL7 | OCL6 | OCL5 | OCL4 | OCL3 | OCL2 | OCL1 | OCL0 |
| RESET | U | U | U | U | U | U | U | U |

## PORTA

### Port A Data Register (PORTA) $00

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PA7 | PA6 | PA5 | PA4 | PA3 | PA2 | PA1 | PA0 |

RESET

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| U | U | U | U | U | U | U | U |

## PORTB

### Port B Data Register (PORTB) $01

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PB7 | PB6 | PB5 | PB4 | PB3 | PB2 | PB1 | PB0 |

RESET

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| U | U | U | U | U | U | U | U |

## PORTC

### Port C Data Register (PORTC) $02

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PC7 | PC6 | PC5 | PC4 | PC3 | PC2 | PC1 | PC0 |

RESET

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| U | U | U | U | U | U | U | U |

## PORTD

### Port D Data Register (PORTD) $03

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PD7 | | PD5 | PD4 | PD3 | PD2 | PD1 | PD0 |

RESET

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| PD7 | 0 | PD5/<br>$\overline{SS}$ | PD4/<br>SCK | PD3/<br>MOSI | PD2/<br>MISO | PD1/<br>TDO | PD0/<br>RDI |

(Port D fixed input register)

RESET

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| U | U | U | U | U | U | U | U |

## (MC68HC705C8 ONLY)

## Program Register (PR) $1C

The program register ($1C) is used to perform PROM programming.

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
|  | 0 | 0 | 0 | 0 | 0 | LAT | 0 | PGM |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

LAT — Latch Enable

1 = Enables PROM data and address bus latches for programming or erasing on the next byte write cycle.

0 = Latch disabled. PROM data and address buses are unlatched for normal CPU operations.

This bit is both readable and writable.

PGM — Program

1 = Applies $V_{PP}$ power to the PROM for programming.

0 = $V_{PP}$ power off.

If LAT is cleared, PGM cannot be set.

Bits 1, 7–3 — Not Used

Always read zero.

## (MC68HC805C4 ONLY)

## Program Register (PR) $1C

The program register ($1C) is used for single-byte EEPROM programming.

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
|  | 0 | CPEN | 0 | 0 | ERASE | LATA | LATB | EEPGM |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

CPEN — Charge Pump Enable

1 = Charge pump enabled

0 = Charge pump disabled

ERASE — Erase EEPROM Enable

1 = Erase enabled

0 = Erase disabled

LATA — Latch A Enable

1 = Enables array A data and address bus latches for programming or erasing on the next byte write cycle.

0 = Latch disabled

LATB — Latch B Enable

1 = Enables array B data and address bus latches for programming or erasing on the next byte write cycle.

0 = Latch disabled

Note: If LATA and LATB are cleared, EEPGM cannot be set.

EEPGM — Electrically Erase/Program
1 = Applies $V_{PP}$ power to the EEPROM array for programming or erasing operation.
0 = $V_{PP}$ power off

Bits 4, 5, 7 — Not used
Always read zero.

# SCCR1

## Serial Communications Control Reg. 1 (SCCR1) $0E

The SCCR1 register control bits determine word length and select the wake-up method.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | R8 | T8 | — | M | WAKE | — | — | — |
| RESET | U | U | — | U | U | — | — | — |

R8 — Receive Data Bit 8
R8 bit provides storage location for the ninth bit in the receive data byte (if M = 1).

T8 — Transmit Data Bit 8
T8 bit provides storage location for the ninth bit in the transmit data byte (if M = 1).

M — SCI Character Word Length
1 = One start bit, nine data bits, one stop bit
0 = One start bit, eight data bits, one stop bit

WAKE — Wake-Up Select
Wake bit selects the receiver wake-up method.
1 = Address bit (most significant bit)
0 = Idle line condition

Bits 2–0 and 5 — Not used
Can read either one or zero.

The address bit is dependent on both the wake-bit and the m-bit level. Additionally, the receiver does not use the wake-up feature unless the RWU control bit is SCCR2 is set.

| Wake | M | Receiver Wake-Up |
|---|---|---|
| 0 | X | Detection of an idle line allows the next data byte received to cause the receive data register to fill and produce an RDRF flag. |
| 1 | 0 | Detection of a received one in the eighth data bit allows an RDRF flag and associated error flags. |
| 1 | 1 | Detection of a received one in the ninth data bit allows an RDRF flag and associated error flags. |

## Serial Communications Control Reg. 2 (SCCR2) $0F

The SCCR2 provides control of individual SCI functions such as interrupts, transmit/receive enabling, receiver wakeup, and break code.

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

TIE — Transmit Interrupt Enable
1 = SCI interrupt enabled
0 = TDRE interrupt disabled

TCIE — Transmit Complete Interrupt Enable
1 = SCI interrupt enabled
0 = TC interrupt disabled

RIE — Receive Interrupt Enable
1 = SCI interrupt enabled
0 = RDRF and OR interrupts disabled

ILIE — Idle Line Interrupt Enable
1 = SCI interrupt enabled
0 = Idle interrupt disabled

TE — Transmit Enable
1 = Transmit shift register output is applied to the TDO line. Depending upon the SCCR1 M bit, a preamble of 10 (M = 0) or 11 (M = 1) consecutive ones is transmitted.
0 = Transmitter disabled after last byte is loaded in the SCDAT and TDRE is set. After last byte is transmitted, TDO line becomes a high-impedance line.

RE — Receive Enable
1 = Receiver shift register input is applied to the RDI line.
0 = Receiver disabled and RDRF, IDLE, OR, NF, and FE status bits are inhibited.

RWU — Receiver Wake-Up

1 = Places receiver in sleep mode and enables wake-up function.

0 = Wake-up function disabled after receiving data word with MSB set (if WAKE = 1). Wake-up function also disabled after receiving 10 (M = 0) or 11 (M = 1) consecutive ones (if WAKE = 0).

SBK — Send Break

1 = Transmitter continually sends blocks of zeros (sets of 10 or 11) until cleared. Upon completion of break code, transmitter sends one high bit for recognition of valid start bit.

0 = Transmitter sends 10 (M = 0) or 11 (M = 1) zeros then reverts to an idle state or continues sending data. If transmitter is empty and idle, setting and clearing the SBK bit may queue up to two character times of break because the first break transfer immediately to the shift register, and the second is queued into the parallel transmit buffer.

# SCDAT

## Serial Communications Data Register (SCDAT) $11

The SCDAT is a read/write register used to receive and transmit SCI data.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SCD7 | SCD6 | SCD5 | SCD4 | SCD3 | SCD2 | SCD1 | SCD0 |
| RESET | U | U | U | U | U | U | U | U |

# SCSR

## Serial Communications Status Register (SCSR) $10

The SCSR provides inputs to the SCI interrupt logic circuits. Noise flag and framing error bits are also contained in the SCSR.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TDRE | TC6 | RDRF | IDLE | OR | NF | FE | — |
| RESET | 1 | 1 | 0 | 0 | 0 | 0 | 0 | — |

TDRE — Transmit Data Register (TDR) Empty

0 = TDR contents transferred to the transmit data shift register.

0 = TDR still contains data. TDRE is cleared by reading the SCSR followed by a write to the TDR.

TC — Transmit Complete

1 = Indicates end of data frame, preamble, or break condition has occurred.

0 = TC bit cleared by reading the SCSR, followed by a write to the TDR.

RDRF—Receive Data Register (RDR) Full

1 = Receive data shift register contents transferred to the RDR.

0 = Receive data shift register transfer did not occur. RDRF is cleared by reading the SCSR, followed by a read of the RDR.

IDLE — Idle Line Detect

1 = Indicates receiver has detected an idle line.

0 = IDLE is cleared by reading the SCSR, followed by a read of the RDR. Once IDLE is cleared, IDLE cannot be set until RDI line becomes active and idle again.

OR — Overrun Error

1 = Indicates receive data shift register data is sent to a full RDR (RDRF = 1). Data causing the overrun is lost, and RDR data is not disturbed.

0 = OR is cleared by reading the SCSR, followed by a read of the RDR.

NF — Noise Flag

1 = Indicates noise is present on the receive bits, including the start and stop bits. NF is not set until RDRF = 1.

0 = NF is cleared by reading the SCSR, followed by a read of the RDR.

FE — Framing Error

1 = Indicates stop bit not detected in received data character. FE is set the same time RDRF is set. If received byte causes both framing and overrun errors, processor will only recognize the overrun error. Further data transfer into the RDR is inhibited until FE is cleared.

0 = FE is cleared by reading the SCSR, followed by a read of the RDR.

Bit 0 — Not used

Can read either one or zero.

## Serial Peripheral Control Register (SPCR) $0A

The SPCR provides control of individual SPI functions such as interrupt and system enabling disabling, master slave mode select, and clock polarity phase rate select.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SPIE | SPE | DWOM* | MSTR | CPOL | CPHA | SPR1 | SPR0 |
| RESET | 0 | 0 | — | 0 | U | U | U | U |

SPIE — Serial Peripheral Interrupt Enable
1 = SPI interrupt enabled
0 = SPI interrupt disabled

SPE — Serial Peripheral System Enable
1 = SPI system on
0 = SPI system off

MSTR — Master Mode Select
1 = Master mode
0 = Slave mode

CPOL — Clock Polarity

Clock polarity bit controls the clock value and is used in conjunction with the clock phase (CPHA) bit.
1 = SCK line idles high
0 = SCK line idles in low state

CPHA — Clock Phase

Clock phase bit along with CPOL controls the clock-data relationship between the master and slave devices. CPOL selects one or two clocking protocols.
1 = $\overline{SS}$ is an output enable control.
0 = Shift clock is the OR or SCK with $\overline{SS}$.
When $\overline{SS}$ is low, first edge of SCK invokes first data sample.

SPR1–SPR0 — SPI Clock Rate Bits

Two clock rate bits are used to select one of four clock rates to be used as SCK in the master mode. In the slave mode, the two clock rate bits have no effect. Clock rate selection is shown in the following table.

| SPR1 | SPR0 | Internal Processor Clock Divided By |
|---|---|---|
| 0 | 0 | 2 |
| 0 | 1 | 4 |
| 1 | 0 | 16 |
| 1 | 1 | 32 |

Bit 5 — Not used*

Can read either one or zero.

(*MC68HC05C9 only, bit 5 (DWOM) is the wire-OR mode bit.)

1 = Disables active pullup devices on Port D, causing outputs to be open drain.
0 = Open-drain disabled.

## Serial Peripheral Data I/O Register (SPDR) $0C

The SPDR is read/write register used to receive and transmit SPI data.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SPD7 | SPD6 | SPD5 | SPD4 | SPD3 | SPD2 | SPD1 | SPD0 |
| RESET | U | U | U | U | U | U | U | U |

A write to the SPDR places data directly into the shift register for transmission. Only a write to this register will initiate transmission/reception of another byte and will only occur in the master device. On completion of byte transmission, the SPIF status bit is set in both master and slave devices.

A read to the SPDR causes the buffer to be read. The first SPIF status bit must be cleared by the time a second data transfer from the shift register to the read buffer begins, or an overrun condition will exist. In overrun cases, the byte causing the overrun is lost.

## SPSR

### Serial Peripheral Status Register (SPSR) $0B

The SPSR contains three status bits.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SPIF | WCOL | — | MODF | — | — | — | — |
| RESET | 0 | 0 | — | 0 | — | — | — | — |

SPIF — Serial Peripheral Data Transfer Flag
- 1 = Indicates data transfer completed between processor and external device. (If SPIF = 1 and SPIE = 1, SPI interrupt is enabled.)
- 0 = Clearing is accomplished by reading SPSR, followed by SPDR access.

WCOL — Write Collision
- 1 = Indicates an attempt is made to write to SPDR while data transfer is in processor.
- 0 = Clearing is accomplished by reading SPSR, followed by SPDR access.

MODF — Mode Fault Flag
- 1 = Indicates multi-master system control conflict.
- 0 = Clearing is accomplished by reading SPSR, followed by a write to the SPCR.

Bits 3–0 and 5 — Not used

Can read either zero or one.

## Timer Control Register (TCR) $12

The TCR is a read/write register containing five control bits. Three bits control interrupts associated with the timer status register flags ICF, OCF, and TOF.

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
|  | ICIE | OCIE | TOIE | 0 | 0 | 0 | IEDG | OLVL |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | U | 0 |

ICIE — Input Capture Interrupt Enable
1 = Interrupt enabled
0 = Interrupt disabled

OCIE — Output Compare Interrupt Enable
1 = Interrupt enabled
0 = Interrupt disabled

TOIE — Timer Overflow Interrupt Enable
1 = Interrupt enabled
0 = Interrupt disabled

IEDG — Input Edge

Value of input edge determines which level transition on TCAP pin will trigger free-running counter transfer to the input capture register.
1 = Positive edge
0 = Negative edge

OLVL — Output Level

Value of output level is clocked into output level register by the next successful output compare and will appear on the TCMP pin.
1 = High output
0 = Low output

Bits 2, 3, and 4 — Not used

Always read zero.

## Timer Status Register (TSR) $13

The TSR is a read-only register containing three status flag bits.

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
|  | ICF | OCF | TOF | 0 | 0 | 0 | 0 | 0 |
| RESET | U | U | U | 0 | 0 | 0 | 0 | 0 |

ICF — Input Capture Flag

1 = Flag set when selected polarity edge is sensed by input capture edge detector.

0 = Flag cleared when TSR and input capture low register ($15) are accessed.

OCF — Output Compare Flag

1 = Flag set when output compare register contents match the free-running counter contents.

0 = Flag cleared when TSR and output compare low register ($17) are accessed.

TOF — Timer Overflow Flag

1 = Flag set when free-running counter transition from $FFFF to $0000 occurs.

0 = Flag cleared when TSR and counter low register ($19) are accessed.

Bits 4–0 — Not used

Always read zero.

# ADDRESSING MODES

### IMMEDIATE (IMM)

The effective address (EA) of an immediate mode instruction is the location following the opcode. This mode is used to fetch a value which is known at the time the program is written, and which is not changed during program execution.

### DIRECT (DIR)

The EA of a direct mode instruction is the contents of the byte following the opcode. This mode is used to fetch a value from any one of the first 256 memory locations with a two-byte instruction.

### EXTENDED (EXT)

The EA of an extended mode instruction is the contents of the next two bytes following the opcode. This mode is used to fetch a value from any location in the MC146805G2 memory location, I/O, RAM, and ROM, with a three-byte instruction.

### INDEXED (IX, IX1, IX2)

The EA of an indexed mode instruction is determined by the contents of the X-register being added to an offset. The offset can be either zero, 8-bit, or 16-bit. For zero offset (IX), the X-register is the EA. For 8-bit offset (IX1), the result of the X-register contents added to the byte following the opcode is the EA. For 16-bit offset (IX2), the result of the X-register contents added to the concatenated contents of the two bytes following the opcode is the EA.

### RELATIVE (REL)

The EA of a relative mode instruction depends upon whether or not the branch is taken. If a branch is taken, EA is formed by adding the byte following the opcode to the value of the program counter, and the program counter is loaded with the EA. If no branch is required, EA is equal to the contents of the program counter.

### BIT SET/CLEAR (BSC)

The EA of a Bit Set/Clear mode instruction is contained in the byte following the opcode. The actual bit which is to be set or cleared is contained in the lower four bits (nibble) of the opcode.

### BIT TEST AND BRANCH (BTB)

This addressing mode combines direct, relative and bit addressing. The EA of this instruction is the contents of the byte following the opcode (direct mode), if no branch is taken. If a branch is taken, the EA becomes the result of the second byte following the opcode being added to the value of the program counter (similar to relative mode). The actual bit which is to be tested is contained in the lower four bits (nibble) of the opcode.

### INHERENT (INH)

This addressing mode has no EA since all information necessary to carry out the instruction is contained in the opcode.

# M68HC05 INSTRUCTION SET

The following table is an alphabetical listing of the instructions available to the M68HC05 MCU user. In listing all the factors necessary to program, the table uses the following symbols:

## Condition Code Symblols

| | | | |
|---|---|---|---|
| H | — Half Carry (Bit 4) | ↕ | — Test and Set if True, (Cleared otherwise) |
| I | — Interrupt Mask (Bit 3) | | |
| N | — Negate (Sign Bit 2) | — | — Not Affected |
| Z | — Zero (Bit 1) | ? | — Load CC Reg. from Stack |
| C | — Carry/Borrow (Bit 0) | 0 | — Cleared |
| | | 1 | — Set |

## Boolean Operators

| | | | |
|---|---|---|---|
| ( ) | — Contents of (i.e.) (M) = means the contents of memory location M | + | — (inclusive) OR |
| | | ⊕ | — EXCLUSIVE OR |
| | | — | — NOT |
| | | - | — negation (twos complement) |
| ← | — is loaded with, 'gets' | | |
| ● | — AND | × | — multiplication |

## MPU Registers

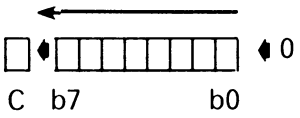
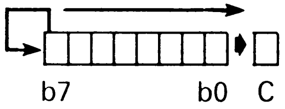
| | | | |
|---|---|---|---|
| A | — Accumulator | PC | — Program Counter |
| ACCA | — Accumulator | PCH | — PC High Byte |
| CC | — Condition Code Reg. | PCL | — PC Low Byte |
| X | — Index Register | SP | — Stack Pointer |
| M | — Any memory location (one byte) | REL | — Relative Address |

| Addressing Modes | (Abbreviation) | Operands | |
|---|---|---|---|
| Inherent | INH | none | |
| Immediate | IMM | ii | |
| Direct (for bit | DIR | dd | |
| test instructions) | | dd | rr |
| Extended | EXT | hh | ll |
| Indexed 0 Offset | IX | none | |
| Indexed 1-Byte | IX1 | ff | |
| Indexed 2-Byte | IX2 | ee | ff |
| Relative | REL | rr | |

[redacted]

# INSTRUCTIONS, ADDRESSING MODES, AND EXECUTION TIMES

| Source Form(s) | Operation | Boolean Expression | Addressing Mode for Operand | Machine Coding (hexadecimal) | | Bytes | Cycles | Condition Code | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Opcode | Operand | | | H | I | N | Z | C |
| ADC (opr) | Add with Carry | ACCA ← ACCA + M + C | IMM | A9 | ii | 2 | 2 | ↕ | — | ↕ | ↕ | ↕ |
| | | | DIR | B9 | dd | 2 | 3 | | | | | |
| | | | EXT | C9 | hh ll | 3 | 4 | | | | | |
| | | | IX2 | D9 | ee ff | 3 | 5 | | | | | |
| | | | IX1 | E9 | ff | 2 | 4 | | | | | |
| | | | IX | F9 | | 1 | 3 | | | | | |
| ADD (opr) | Add | ACCA ← ACCA + M | IMM | AB | ii | 2 | 2 | ↕ | — | ↕ | ↕ | ↕ |
| | | | DIR | BB | dd | 2 | 3 | | | | | |
| | | | EXT | CB | hh ll | 3 | 4 | | | | | |
| | | | IX2 | DB | ee ff | 3 | 5 | | | | | |
| | | | IX1 | EB | ff | 2 | 4 | | | | | |
| | | | IX | FB | | 1 | 3 | | | | | |
| AND (opr) | Logical AND | ACCA ← ACCA ● M | IMM | A4 | ii | 2 | 2 | — | — | ↕ | ↕ | — |
| | | | DIR | B4 | dd | 2 | 3 | | | | | |
| | | | EXT | C4 | hh ll | 3 | 4 | | | | | |
| | | | IX2 | D4 | ee ff | 3 | 5 | | | | | |
| | | | IX1 | E4 | ff | 2 | 4 | | | | | |
| | | | IX | F4 | | 1 | 3 | | | | | |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ASL (opr)<br>ASLA<br>ASLX<br>ASL (opr)<br>ASL (opr) | Arithmetic Shift Left |  | DIR<br>INH(A)<br>INH(X)<br>IX1<br>IX | 38<br>48<br>58<br>68<br>78 | dd<br><br><br>ff<br> | 2<br>1<br>1<br>2<br>1 | 5<br>3<br>3<br>6<br>5 | — | — | ↕ | ↕ | ↕ |
| ASR (opr)<br>ASRA<br>ASRX<br>ASR (opr)<br>ASR (opr) | Arithmetic Shift Right |  | DIR<br>INH(A)<br>INH(X)<br>IX1<br>IX | 37<br>47<br>57<br>67<br>77 | dd<br><br><br>ff<br> | 2<br>1<br>1<br>2<br>1 | 5<br>3<br>3<br>6<br>5 | — | — | ↕ | ↕ | ↕ |
| BCC (rel) | Branch if Carry Clear | ? C = 0 | REL | 24 | rr | 2 | 3 | — | — | — | — | — |
| BCLR n, (opr) | Clear Bit n in Memory | Mn ← 0 | DIR(b0)<br>DIR(b1)<br>DIR(b2)<br>DIR(b3)<br>DIR(b4)<br>DIR(b5)<br>DIR(b6)<br>DIR(b7) | 11<br>13<br>15<br>17<br>19<br>1B<br>1D<br>1F | dd<br>dd<br>dd<br>dd<br>dd<br>dd<br>dd<br>dd | 2<br>2<br>2<br>2<br>2<br>2<br>2<br>2 | 5<br>5<br>5<br>5<br>5<br>5<br>5<br>5 | — | — | — | — | — |
| BCS (rel) | Branch if Carry Set | ? C = 1 | REL | 25 | rr | 2 | 3 | — | — | — | — | — |
| BEQ (rel) | Branch if Equal | ? Z = 1 | REL | 27 | rr | 2 | 3 | — | — | — | — | — |

| Source Form(s) | Operation | Boolean Expression | Addressing Mode for Operand | Machine Coding (hexadecimal) | | Bytes | Cycles | Condition Code | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Opcode | Operand | | | H | I | N | Z | C |
| BHCC (rel) | Branch if Half Carry Clear | ? H = 0 | REL | 28 | rr | 2 | 3 | — | — | — | — | — |
| BHCS (rel) | Branch if Half Carry Set | ? H = 1 | REL | 29 | rr | 2 | 3 | — | — | — | — | — |
| BHI (rel) | Branch if Higher | ? (C + Z) = 0 | REL | 22 | rr | 2 | 3 | — | — | — | — | — |
| BHS (rel) | Branch if Higher or Same | ? C = 0 | REL | 24 | rr | 2 | 3 | — | — | — | — | — |
| BIH (rel) | Branch if $\overline{\text{IRQ}}$ Pin is High | ? $\overline{\text{IRQ}}$ Pin = 1 | REL | 2F | rr | 2 | 3 | — | — | — | — | — |
| BIL (rel) | Branch if $\overline{\text{IRQ}}$ Pin is Low | ? $\overline{\text{IRQ}}$ Pin = 0 | REL | 2E | rr | 2 | 3 | — | — | — | — | — |
| BIT (rel) | Bit Test Memory with A | ACCA ● M | IMM<br>DIR<br>EXT<br>IX2<br>IX1<br>IX | A5<br>B5<br>C5<br>D5<br>E5<br>F5 | ii<br>dd<br>hh ll<br>ee ff<br>ff | 2<br>2<br>3<br>3<br>2<br>1 | 2<br>3<br>4<br>5<br>4<br>3 | — | — | ↕ | ↕ | — |
| BLO (rel) | Branch if Lower | ? C = 1 | REL | 25 | rr | 2 | 3 | — | — | — | — | — |
| BLS (rel) | Branch if Lower or Same | ? (C + X) = 1 | REL | 23 | rr | 2 | 3 | — | — | — | — | — |
| BMC (rel) | Branch if I Bit is Clear | ? I = 0 | REL | 2C | rr | 2 | 3 | — | — | — | — | — |

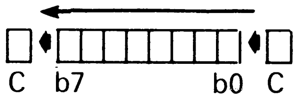
| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BMI (rel) | Branch if Minus | ? N = 1 | REL | 2B | rr | 2 | 3 | — | — | — | — | — |
| BMS (rel) | Branch if I Bit is Set | ? I = 1 | REL | 2D | rr | 2 | 3 | — | — | — | — | — |
| BNE (rel) | Branch if Not Equal | ? Z = 0 | REL | 26 | rr | 2 | 3 | — | — | — | — | — |
| BPL (rel) | Branch if Plus | ? N = 0 | REL | 2A | rr | 2 | 3 | — | — | — | — | — |
| BRA (rel) | Branch Always | ? 1 = 1 | REL | 20 | rr | 2 | 3 | — | — | — | — | — |
| BRCLR n, (opr) (rel) | Branch if Bit n of M = 0 | ? Bit n of M = 0 | DIR(b0) | 01 | dd rr | 3 | 5 | — | — | — | — | ↕ |
| | | | DIR(b1) | 03 | dd rr | 3 | 5 | | | | | |
| | | | DIR(b2) | 05 | dd rr | 3 | 5 | | | | | |
| | | | DIR(b3) | 07 | dd rr | 3 | 5 | | | | | |
| | | | DIR(b4) | 09 | dd rr | 3 | 5 | | | | | |
| | | | DIR(b5) | 0B | dd rr | 3 | 5 | | | | | |
| | | | DIR(b6) | 0D | dd rr | 3 | 5 | | | | | |
| | | | DIR(b7) | 0F | dd rr | 3 | 5 | | | | | |
| BRN (rel) | Branch Never | ? 1 = 0 | REL | 21 | rr | 2 | 3 | — | — | — | — | — |
| BRSET n, (opr) (rel) | Branch if Bit n of M = 1 | ? Bit n of M = 1 | DIR(b0) | 00 | dd rr | 3 | 5 | — | — | — | — | ↕ |
| | | | DIR(b1) | 02 | dd rr | 3 | 5 | | | | | |
| | | | DIR(b2) | 04 | dd rr | 3 | 5 | | | | | |
| | | | DIR(b3) | 06 | dd rr | 3 | 5 | | | | | |
| | | | DIR(b4) | 08 | dd rr | 3 | 5 | | | | | |
| | | | DIR(b5) | 0A | dd rr | 3 | 5 | | | | | |
| | | | DIR(b6) | 0C | dd rr | 3 | 5 | | | | | |
| | | | DIR(b7) | 0E | dd rr | 3 | 5 | | | | | |

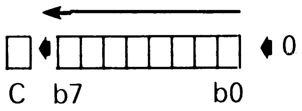
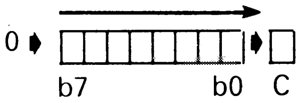
| Source Form(s) | Operation | Boolean Expression | Addressing Mode for Operand | Machine Coding (hexadecimal) | | Bytes | Cycles | Condition Code | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Opcode | Operand | | | H | I | N | Z | C |
| BSET n, (opr) | Set Bit n in Memory | Mn ← 1 | DIR(b0) | 10 | dd | 2 | 5 | — | — | — | — | — |
| | | | DIR(b1) | 12 | dd | 2 | 5 | | | | | |
| | | | DIR(b2) | 14 | dd | 2 | 5 | | | | | |
| | | | DIR(b3) | 16 | dd | 2 | 5 | | | | | |
| | | | DIR(b4) | 18 | dd | 2 | 5 | | | | | |
| | | | DIR(b5) | 1A | dd | 2 | 5 | | | | | |
| | | | DIR(b6) | 1C | dd | 2 | 5 | | | | | |
| | | | DIR(b7) | 1E | dd | 2 | 5 | | | | | |
| BSR (rel) | Branch to Subroutine | PC ← PC+0002 | REL | AD | rr | 2 | 6 | — | — | — | — | — |
| | | (SP) ← PCL; SP ← SP−0001 | | | | | | | | | | |
| | | (SP) ← PCH; SP ← SP−0001 | | | | | | | | | | |
| | | PC ← PC+Rel | | | | | | | | | | |
| CLC | Clear C Bit | C bit ← 0 | INH | 98 | | 1 | 2 | — | — | — | — | 0 |
| CLI | Clear I Bit | I bit ← 0 | INH | 9A | | 1 | 2 | — | 0 | — | — | — |
| CLR (opr) | Clear | M ← 00 | DIR | 3F | dd | 2 | 5 | — | — | 0 | 1 | — |
| CLRA | | A ← 00 | INH(A) | 4F | | 1 | 3 | | | | | |
| CLRX | | X ← 00 | INH(X) | 5F | | 1 | 3 | | | | | |
| CLR (opr) | | M ← 00 | IX1 | 6F | ff | 2 | 6 | | | | | |
| CLR (opr) | | M ← 00 | IX | 7F | | 1 | 5 | | | | | |

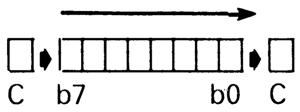
| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CMP (opr) | Compare A with Memory | ACCA − M | IMM | A1 | ii | | 2 | 2 | — | — | ↕ | ↕ | ↕ |
| | | | DIR | B1 | dd | | 2 | 3 | | | | | |
| | | | EXT | C1 | hh | ll | 3 | 4 | | | | | |
| | | | IX2 | D1 | ee | ff | 3 | 5 | | | | | |
| | | | IX1 | E1 | ff | | 2 | 4 | | | | | |
| | | | IX | F1 | | | 1 | 3 | | | | | |
| COM (opr) | 1's Complement | M ← $\overline{M}$ = $FF − M | DIR | 33 | dd | | 2 | 5 | — | — | ↕ | ↕ | 1 |
| COMA | | A ← $\overline{A}$ = $FF − A | INH(A) | 43 | | | 1 | 3 | | | | | |
| COMX | | X ← $\overline{X}$ = $FF − X | INH(X) | 53 | | | 1 | 3 | | | | | |
| COM (opr) | | M ← $\overline{M}$ = $FF − M | IX1 | 63 | ff | | 2 | 6 | | | | | |
| COM (opr) | | M ← $\overline{M}$ = $FF − M | IX | 73 | | | 1 | 5 | | | | | |
| CPX (opr) | Compare X with Memory | X − M | IMM | A3 | ii | | 2 | 2 | — | — | ↕ | ↕ | ↕ |
| | | | DIR | B3 | dd | | 2 | 3 | | | | | |
| | | | EXT | C3 | hh | ll | 3 | 4 | | | | | |
| | | | IX2 | D3 | ee | ff | 3 | 5 | | | | | |
| | | | IX1 | E3 | ff | | 2 | 4 | | | | | |
| | | | IX | F3 | | | 1 | 3 | | | | | |
| DEC (opr) | Decrement | M ← M − 01 | DIR | 3A | dd | | 2 | 5 | — | — | ↕ | ↕ | — |
| DECA | | A ← A − 01 | INH(A) | 4A | | | 1 | 3 | | | | | |
| DECX | DEX (same as DECX) | X ← X − 01 | INH(X) | 5A | | | 1 | 3 | | | | | |
| DEC (opr) | | M ← M − 01 | IX1 | 6A | ff | | 2 | 6 | | | | | |
| DEC (opr) | | M ← M − 01 | IX | 7A | | | 1 | 5 | | | | | |

[redacted]

| Source Form(s) | Operation | Boolean Expression | Addressing Mode for Operand | Machine Coding (hexadecimal) | | Bytes | Cycles | Condition Code | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Opcode | Operand | | | H | I | N | Z | C |
| EOR (opr) | Exclusive OR A with Memory | ACCA ← ACCA ⊕ M | IMM | A8 | ii | 2 | 2 | — | — | ↕ | ↕ | — |
| | | | DIR | B8 | dd | 2 | 3 | | | | | |
| | | | EXT | C8 | hh ll | 3 | 4 | | | | | |
| | | | IX2 | D8 | ee ff | 3 | 5 | | | | | |
| | | | IX1 | E8 | ff | 2 | 4 | | | | | |
| | | | IX | F8 | | 1 | 3 | | | | | |
| INC (opr) | Increment | M ← M+01 | DIR | 3C | dd | 2 | 5 | — | — | ↕ | ↕ | — |
| INCA | | A ← A+01 | INH(A) | 4C | | 1 | 3 | | | | | |
| INCX | INX (same as INCX) | X ← X+01 | INH(X) | 5C | | 1 | 3 | | | | | |
| INC (opr | | M ← M+01 | IX1 | 6C | ff | 2 | 6 | | | | | |
| INC (opr) | | M ← M+01 | IX | 7C | | 1 | 5 | | | | | |
| JMP (opr) | Jump | PC ← effective address | DIR | BC | dd | 2 | 2 | — | — | — | — | — |
| | | | EXT | CC | hh ll | 3 | 3 | | | | | |
| | | | IX2 | DC | ee ff | 3 | 4 | | | | | |
| | | | IX1 | EC | ff | 2 | 3 | | | | | |
| | | | IX | FC | | 1 | 2 | | | | | |
| JSR (opr) | Jump to Subroutine | PC ← PC+n (n=1, 2, or 3) | DIR | BD | dd | 2 | 5 | — | — | — | — | — |
| | | (SP) ← PCL; SP ← SP−0001 | EXT | CD | hh ll | 3 | 6 | | | | | |
| | | (SP) ← PCH; SP ← SP−0001 | IX2 | DD | ee ff | 3 | 7 | | | | | |
| | | PC ← effective address | IX1 | ED | ff | 2 | 6 | | | | | |
| | | | IX | FD | | 1 | 5 | | | | | |

| Source Form(s) | Operation | Boolean Expression | Addressing Mode for Operand | Machine Coding (hexadecimal) | | Bytes | Cycles | Condition Code | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Opcode | Operand | | | H | I | N | Z | C |
| NEG (opr)<br>NEGA<br>NEGX<br>NEG (opr)<br>NEG (opr | Negates (2's Complement) | M ← −M (i.e. 00 − M)<br>A ← −A<br>X ← −X<br>M ← −M<br>M ← −M | DIR<br>INH(A)<br>INH(X)<br>IX1<br>IX | 30<br>40<br>50<br>60<br>70 | dd<br><br><br>ff<br> | 2<br>1<br>1<br>2<br>1 | 5<br>3<br>3<br>6<br>5 | — | — | ↕ | ↕ | ↕ |
| NOP | No Operation | | INH | 9D | | 1 | 2 | — | — | — | — | |
| ORA (opr) | Inclusive OR | ACCA ← ACCA + M | IMM<br>DIR<br>EXT<br>IX2<br>IX1<br>IX | AA<br>BA<br>CA<br>DA<br>EA<br>FA | ii<br>dd<br>hh ll<br>ee ff<br>ff<br> | 2<br>2<br>3<br>3<br>2<br>1 | 2<br>3<br>4<br>5<br>4<br>3 | — | — | ↕ | ↕ | |
| ROL (opr)<br>ROLA<br>ROLX<br>ROL (opr)<br>ROL (opr | Rotate Left through Carry |  | DIR<br>INH(A)<br>INH(X)<br>IX1<br>IX | 39<br>49<br>59<br>69<br>79 | dd<br><br><br>ff<br> | 2<br>1<br>1<br>2<br>1 | 5<br>3<br>3<br>6<br>5 | — | — | ↕ | ↕ | ↕ |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LDA (opr) | Load A from Memory | ACCA ← M | IMM<br>DIR<br>EXT<br>IX2<br>IX1<br>IX | A6<br>B6<br>C6<br>D6<br>E6<br>F6 | ii<br>dd<br>hh ll<br>ee ff<br>ff | 2<br>2<br>3<br>3<br>2<br>1 | 2<br>3<br>4<br>5<br>4<br>3 | — | — | ↕ | ↕ | — |
| LDX (opr) | Load X from Memory | X ← M | IMM<br>DIR<br>EXT<br>IX2<br>IX1<br>IX | AE<br>BE<br>CE<br>DE<br>EE<br>FE | ii<br>dd<br>hh ll<br>ee ff<br>ff | 2<br>2<br>3<br>3<br>2<br>1 | 2<br>3<br>4<br>5<br>4<br>3 | — | — | ↕ | ↕ | — |
| LSL (opr)<br>LSLA<br>LSLX<br>LSL (opr)<br>LSL (opr) | Logical Shift Left |  | DIR<br>INH(A)<br>INH(X)<br>IX1<br>IX | 38<br>48<br>58<br>68<br>78 | dd<br><br><br>ff | 2<br>1<br>1<br>2<br>1 | 5<br>3<br>3<br>6<br>5 | — | — | ↕ | ↕ | ↕ |
| LSR (opr)<br>LSRA<br>LSRX<br>LSR (opr)<br>LSR (opr) | Logical Shift Right |  | DIR<br>INH(A)<br>INH(X)<br>IX1<br>IX | 34<br>44<br>54<br>64<br>74 | dd<br><br><br>ff | 2<br>1<br>1<br>2<br>1 | 5<br>3<br>3<br>6<br>5 | — | — | 0 | ↕ | ↕ |
| MUL | Unsigned Multiply | X:A ← X•A | INH | 42 | | 1 | 11 | 0 | — | — | — | 0 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ROR (opr)<br>RORA<br>RORX<br>ROR (opr)<br>ROR (opr) | Rotate Right through Carry |  | DIR<br>INH(A)<br>INH(X)<br>IX1<br>IX | 36<br>46<br>56<br>66<br>76 | dd<br><br><br>ff | 2<br>1<br>1<br>2<br>1 | 5<br>3<br>3<br>6<br>5 | — | — | ↕ | ↕ | ↕ |
| RSP | Reset Stack Pointer | SP ← $00FF | INH | 9C |  | 1 | 2 | — | — | — | — | — |
| RTI | Return from Interrupt | SP ← SP + 0001; CC ← (SP<br>SP ←SP + 0001; ACCA← (SP)<br>SP ← SP + 0001; X ← (SP)<br>SP ← SP + 0001; PCH ← (SP)<br>SP ← SP + 0001; PCL ← (SP | INH | 80 |  | 1 | 9 | (Loaded from Stack)<br>↕ | ↕ | ↕ | ↕ | ↕ |
| RTS | Return from Subroutine | SP ← SP + 0001; PCH ← (SP)<br>SP ← SP + 0001; PCL ← (SP) | INH | 81 |  | 1 | 6 | — | — | — | — | — |
| SBC (opr) | Subtract with Carry | ACCA ← ACCA − M C | IMM<br>DIR<br>EXT<br>IX2<br>IX1<br>IX | A2<br>B2<br>C2<br>D2<br>E2<br>F2 | ii<br>dd<br>hh ll<br>ee ff<br>ff | 2<br>2<br>3<br>3<br>2<br>1 | 2<br>3<br>4<br>5<br>4<br>3 | — | — | ↕ | ↕ | ↕ |
| SEC | Set C Bit | C bit ← 1 | INH | 99 |  | 1 | 2 | — | — | — | — | 1 |
| SEI | Set I Bit | I bit ← 1 | INH | 9B |  | 1 | 2 | — | 1 | — | — | — |

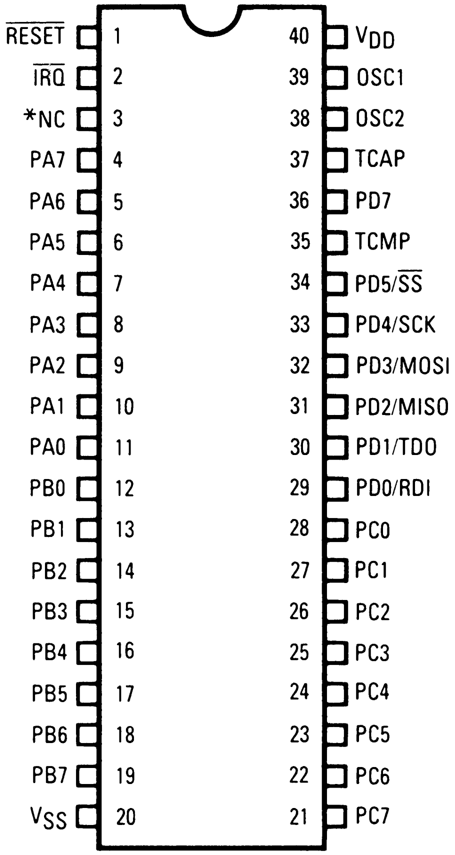
| Source Form(s) | Operation | Boolean Expression | Addressing Mode for Operand | Machine Coding (hexadecimal) | | Bytes | Cycles | Condition Code | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Opcode | Operand | | | H | I | N | Z | C |
| STA (opr) | Store A in Memory | M ← ACCA | DIR | B7 | dd | 2 | 4 | — | — | ↕ | ↕ | — |
| | | | EXT | C7 | hh ll | 3 | 5 | | | | | |
| | | | IX2 | D7 | ee ff | 3 | 6 | | | | | |
| | | | IX1 | E7 | ff | 2 | 5 | | | | | |
| | | | IX | F7 | | 1 | 4 | | | | | |
| STOP | Enable $\overline{\text{IRQ}}$, Stop Oscillator | | INH | 8E | | 1 | 2 | — | 0 | — | — | — |
| STX (opr) | Store X in Memory | M ← X | DIR | BF | dd | 2 | 4 | — | — | ↕ | ↕ | 0 |
| | | | EXT | CF | hh ll | 3 | 5 | | | | | |
| | | | IX2 | DF | ee ff | 3 | 6 | | | | | |
| | | | IX1 | EF | ff | 2 | 5 | | | | | |
| | | | IX | FF | | 1 | 4 | | | | | |
| SUB (opr) | Subtract | ACCA ← ACCA − M | IMM | A0 | ii | 2 | 2 | — | — | ↕ | ↕ | ↕ |
| | | | DIR | B0 | dd | 2 | 3 | | | | | |
| | | | EXT | C0 | hh ll | 3 | 4 | | | | | |
| | | | IX2 | D0 | ee ff | 3 | 5 | | | | | |
| | | | IX1 | E0 | ff | 2 | 4 | | | | | |
| | | | IX | F0 | | 1 | 3 | | | | | |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SWI | Software Interrupt | PC ← PC + 0001<br>(SP) ← PCL; SP ← SP − 0001<br>(SP) ← PCH; SP ← SP − 0001<br>(SP) ← X; SP ← SP − 0001<br>(SP) ← ACCA; SP ← SP − 0001<br>(SP) ← CC; SP ← SP − 0001<br>I bit ← 1<br>PCH ← n − 0003 (vector<br>PCL ← n − 0002 fetch) | INH | 83 | | 1 | 10 | — | 1 | — | — | — |
| TAX | Transfer A to X | X ← ACCA | INH | 97 | | 1 | 2 | — | — | — | — | — |
| TST (opr)<br>TSTA<br>TSTX<br>TST (opr)<br>TST (opr) | Test for Negative or Zero | M − 0 | DIR<br>INH(A)<br>INH(X)<br>IX1<br>IX | 3D<br>4D<br>5D<br>6D<br>7D | dd<br><br><br>ff | 2<br>1<br>1<br>2<br>1 | 4<br>3<br>3<br>5<br>4 | — | — | ↕ | ↕ | 0 |
| TXA | Transfer X to A | ACCA ← X | INH | 9F | | 1 | 2 | — | — | — | — | — |
| WAIT | Enable Interrupts, Halt CPU | | INH | 8F | | 1 | 2 | — | 0 | — | — | — |

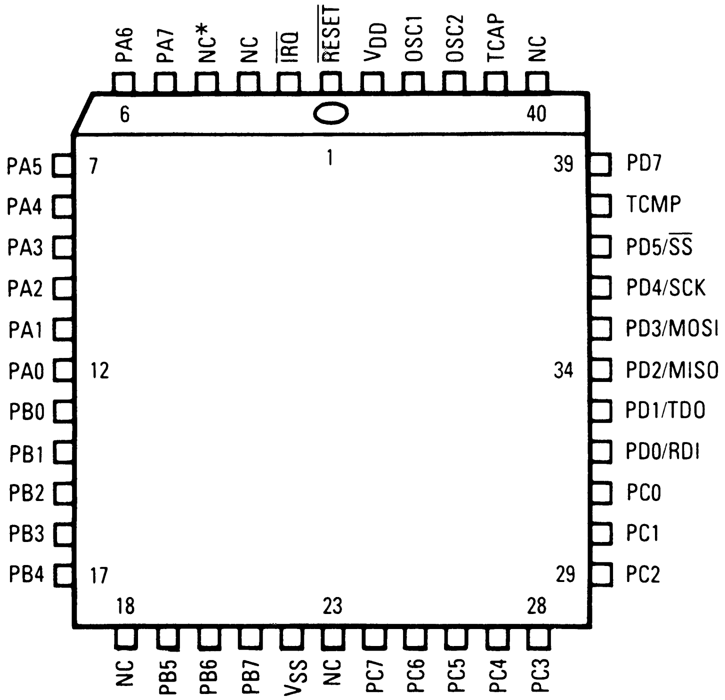
# MC68HC05C4/C8 MC68HCL05C4/C8 AND MC68HSC05C4/C8 MC68HC705C8

# PIN ASSIGNMENTS

### 40-PIN DUAL-IN-LINE PACKAGE



### 44-LEAD PLCC PACKAGE



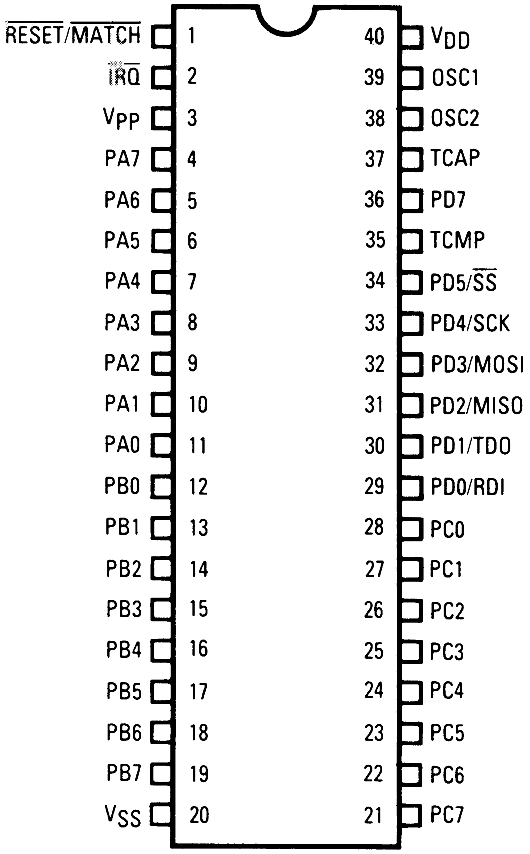
NOTE: Bulk substrate tied to $V_{SS}$.

*This pin is $V_{PP}$ on the MC68HC705C8.

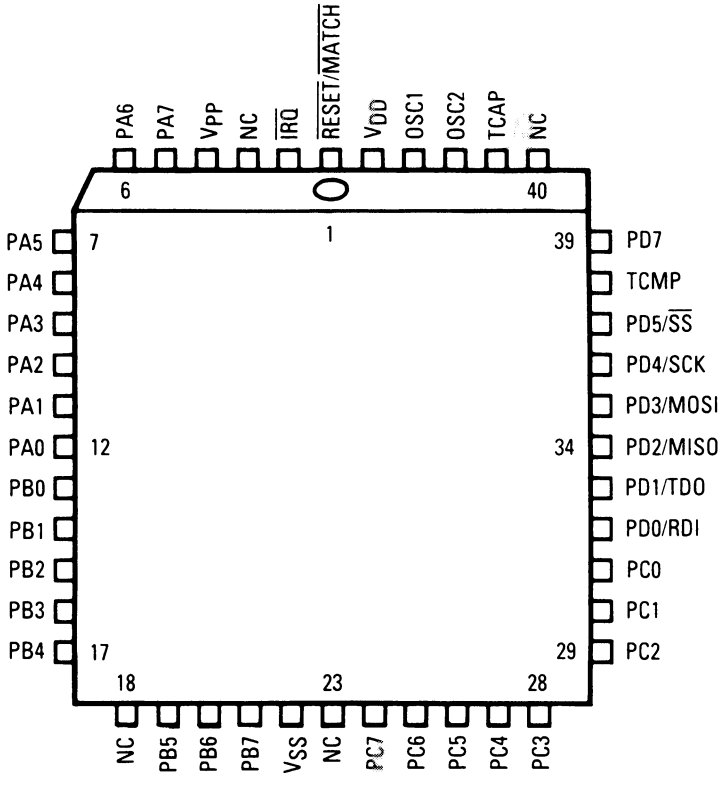
# MC68HC805C4

# PIN ASSIGNMENTS

### 40-PIN DUAL-IN-LINE PACKAGE



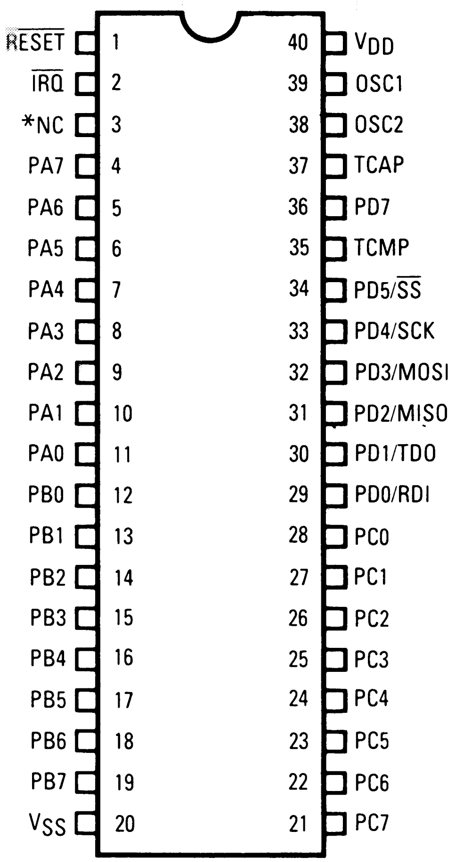
### 44-LEAD PLCC PACKAGE



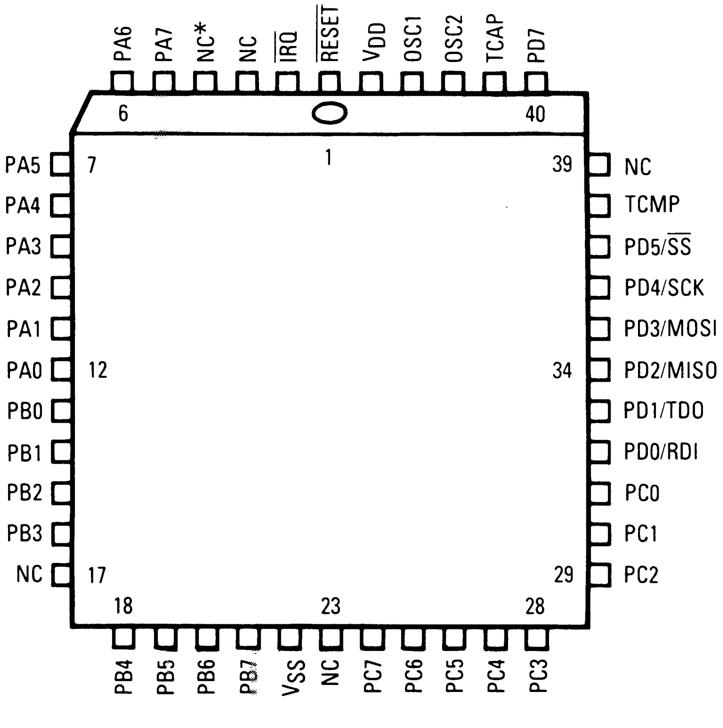
NOTE: Bulk substrate tied to VSS.

# MC68HC05C9 (ONLY) PIN ASSIGNMENTS

40-PIN DUAL-IN-LINE PACKAGE



44-LEAD PLCC PACKAGE



NOTE: Bulk substrate tied to $V_{SS}$.

*This pin is $V_{PP}$ on the MC68HC705C8.

# ASCII CHART

ASCII CHARACTER SET (7-Bit Code)

| LS Dig. \ MS Dig. | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| 0 | NUL | DLE | SP | 0 | @ | P | ` | p |
| 1 | SOH | DC1 | ! | 1 | A | Q | a | q |
| 2 | STX | DC2 | " | 2 | B | R | b | r |
| 3 | ETX | DC3 | # | 3 | C | S | c | s |
| 4 | EOT | DC4 | $ | 4 | D | T | d | t |
| 5 | ENQ | NAK | % | 5 | E | U | e | u |
| 6 | ACK | SYN | & | 6 | F | V | f | v |
| 7 | BEL | ETB | ' | 7 | G | W | g | w |
| 8 | BS | CAN | ( | 8 | H | X | h | x |
| 9 | HT | EM | ) | 9 | I | Y | i | y |
| A | LF | SUB | * | : | J | Z | j | z |
| B | VT | ESC | + | ; | K | [ | k | { |
| C | FF | FS | , | < | L | \ | l | \| |
| D | CR | GS | – | = | M | ] | m | } |
| E | SO | RS | . | > | N | ∧ | n | ~ |
| F | SI | US | / | ? | O | — | o | DEL |

## HEXADECIMAL AND DECIMAL CONVERSION

How to use:

Conversion to Decimal: Find the decimal weights for corresponding hexadecimal characters beginning with the least significant character. The sum of the decimal weights is the decimal value of the hexadecimal number.

Conversion to Hexadecimal: Find the highest decimal value in the table which is lower than or equal to the decimal number to be converted. The corresponding hexadecimal character is the most significant. Subtract the decimal value found from the decimal number to be converted. With the difference repeat the process to find subsequent hexadecimal characters.

| 15 | Byte |  | 8 | 7 | Byte |  | 0 |
|---|---|---|---|---|---|---|---|
| 15 | Char 12 | 11 | Char 8 | 7 | Char 4 | 3 | Char 0 |
| Hex | Dec | Hex | Dec | Hex | Dec | Hex | Dec |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 4,096 | 1 | 256 | 1 | 16 | 1 | 1 |
| 2 | 8,192 | 2 | 512 | 2 | 32 | 2 | 2 |
| 3 | 12,228 | 3 | 768 | 3 | 48 | 3 | 3 |
| 4 | 16,384 | 4 | 1,024 | 4 | 64 | 4 | 4 |
| 5 | 20,480 | 5 | 1,280 | 5 | 80 | 5 | 5 |
| 6 | 24,576 | 6 | 1,536 | 6 | 96 | 6 | 6 |
| 7 | 28,672 | 7 | 1,792 | 7 | 112 | 7 | 7 |
| 8 | 32,768 | 8 | 2,048 | 8 | 128 | 8 | 8 |
| 9 | 36,864 | 9 | 2,304 | 9 | 144 | 9 | 9 |
| A | 40,960 | A | 2,560 | A | 160 | A | 10 |
| B | 45,056 | B | 2,816 | B | 176 | B | 11 |
| C | 49,152 | C | 3,072 | C | 192 | C | 12 |
| D | 53,248 | D | 3,328 | D | 208 | D | 13 |
| E | 57,344 | E | 3,584 | E | 224 | E | 14 |
| F | 61,440 | F | 3,840 | F | 240 | F | 15 |

NOTES

NOTES

NOTES

NOTES

BLOCK DIAGRAMS

MEMORY MAPS

REGISTER/CONTROL
BIT ASSIGNMENTS

INSTRUCTIONS
ADDRESSING MODES
EXECUTION TIMES

MECHANICAL DATA

HEX/DEC CONVERSION
ASCII CHART

The MC68HC05 Family of HCMOS devices covered in this reference guide are as follows:

MC68HC05C4
MC68HC05C8
MC68HC05C9
MC68HC705C8
MC68HC805C4
MC68HCL05C4
MC68HCL05C8
MC68HSC05C4
MC68HSC05C8

Motorola reserves the right to make changes without further notice to any product herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in Motorola data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola products could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part. Motorola and Ⓜ are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

Mfax is a trademark of Motorola, Inc.

**How to reach us:**

**USA/EUROPE/Locations Not Listed:** Motorola Literature Distribution;
P.O. Box 5405, Denver, Colorado 80217. 303-675-2140 or 1-800-441-2447

**Mfax**™. RMFAX0@email.sps.mot.com – TOUCHTONE 602-244-6609
– US & Canada ONLY 1-800-774-1848

**INTERNET:** http://motorola.com/sps

**JAPAN:** Nippon Motorola Ltd.; SPD, Strategic Planning Office, 4-32-1, Nishi-Gotanda, Shinagawa-ku, Tokyo 141, Japan. 81–3–5487–8488

**ASIA/PACIFIC:** Motorola Semiconductors H.K. Ltd.; 8B Tai Ping Industrial Park,
51 Ting Kok Road, Tai Po, N.T., Hong Kong. 852–26629298

MOTOROLA

1ATX23834-3 PRINTED IN USA 5/97 IMPERIAL LITHO 29621 3,000 LITCSIC UCCL