# 68HC705C5

## SPECIFICATION
## (General Release)

©August 29, 1994

CSIC System Design Group
Austin, Texas

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# SECTION 1     INTRODUCTION

## 1.1     GENERAL

The MC68HC705C5 is a 40-pin device based on the MC68HC05C5.  The memory map includes 5184 bytes of user EPROM, 176 bytes of RAM, and 128 bytes of EEPROM.  The MCU has four 8-bit I/O ports: A, B, C, and D.  Port C has high sink current capability.  The MC68HC705C5 includes a simple Serial I/O Peripheral (SIOP), 16-bit Timer, and an on-chip Computer Operating Properly (COP) watchdog circuit.

## 1.2     FEATURES

- HC05 Core

- 40-pin DIP or 44-pin plastic-leaded chip carrier (PLCC) package

- On-Chip Oscillator with Crystal/Ceramic Resonator

- 5184 Bytes of User EPROM

- 176 Bytes of On-Chip RAM

- 128 Bytes of EEPROM

- EEPROM Low Voltage Program Inhibit (LVPI) enabled by LVPI bit in MOR register

- Hardware EEPROM Program Enable

- 16-Bit Timer

- COP Watchdog Timer controlled by the COP bit in the MOR register

- 32 Bidirectional I/O Lines

- Single-Chip Mode

- Boot Mode

- Power-Saving STOP and WAIT Modes

- Edge-Sensitive or Edge- and Level-Sensitive Interrupt Trigger controlled by the IRQ bit in the MOR register

- Simple Serial Input/Output Port MSB/LSB controlled by the SIOP bit in the MOR register

- 10 mA Sink Capability on One 8-Bit Port

OSC1  OSC2

TCAP → TIMER SYSTEM

INTERNAL PROCESSOR CLOCK

OSCILLATOR AND DIVIDE BY 2

COP SYSTEM

$\overline{RESET}$

$\overline{IRQ}$

PA0
PA1
PA2
PORT A I/O LINES  PA3
PA4
PA5
PA6
PA7

PORT A REG   DATA DIR REG

PB0
PB1
PB2
PORT B I/O LINES  PB3
PB4
SDO/PB5
SDI/PB6
SCK/PB7

PORT B REG   DATA DIR REG

SIOP

ACCUMULATOR

INDEX REGISTER

CONDITION CODE REGISTER

STACK POINTER

PROGRAM COUNTER HIGH

PROGRAM COUNTER LOW

CPU CONTROL

CPU

ALU

DATA DIR REG   PORT C REG

PC0
PC1
PC2
PC3  PORT C I/O LINES
PC4
PC5
PC6
PC7

DATA DIR REG   PORT D REG

PD0
PD1
PD2
PD3  PORT D I/O LINES
PD4
PD5
PD6/TCMP
PD7

5184 X 8 USER PROM

367 X 8 BOOT ROM

176 X 8 RAM

128 X 8 EEPROM

PE/V$_{PP}$

**Figure 1-1:  Block Diagram of the MC68HC705C5**

## 1.3     PROGRAMMABLE OPTIONS

Programmable options on the MC68HC705C5 include: $\overline{IRQ}$ (edge-sensitive only or edge and level-sensitive), SIOP (MSB or LSB first), COP Watchdog Timer (enable/disable) and LVPI (enable/disable).   These options are programmed into the MOR by the bootloader.

## 1.4    SIGNAL DESCRIPTION

### 1.4.1    $V_{DD}$ AND $V_{SS}$

Power is supplied to the microcontroller using these two pins.  $V_{DD}$ is the positive supply, and $V_{SS}$ is ground.

### 1.4.2    PE/$V_{PP}$

PE is the Program Enable for the EEPROM.  This pin has a very weak internal pullup.  If this pin is held at a logic "one" level or left not connected, the EEPROM can be programmed and erased.  If this pin is held at a logic "zero" level, the EEPROM programming and erasing is disabled.  The pullup will not be able to pull the pin high if any circuit is connected to PE. PE is shared with $V_{PP}$. $V_{PP}$ is the programming voltage for the EPROM.

### 1.4.3    $\overline{IRQ}$

This pin has an IRQ bit in the MOR register  that provides two different choices of interrupt triggering sensitivity.  The $\overline{IRQ}$ pin contains an internal Schmitt trigger as part of its input to improve noise immunity.  Refer to **4.5 INTERRUPTS** for more detail.

### 1.4.4    OSC1 AND OSC2

These pins provide control input for an on-chip clock oscillator circuit.  A crystal, a ceramic resonator, or an external signal connects to these pins providing a system clock.  The oscillator frequency is two times the internal bus rate.

### 1.4.5    $\overline{RESET}$

This active low pin is used to reset the MCU to a known start-up state by pulling $\overline{RESET}$ low.  The $\overline{RESET}$ pin contains an internal Schmitt trigger as part of its input to improve noise immunity.

### 1.4.6    TCAP

This pin controls the input capture feature for the on-chip programmable timer.  The TCAP pin contains an internal Schmitt trigger as part of its input to improve noise immunity.

### 1.4.7    PA0-PA7

These eight I/O lines comprise port A.  The state of any pin is software programmable and all port A lines are configured as input during power-on or reset.  For detailed information on I/O programming, see **5.4.1 INPUT/OUTPUT PROGRAMMING**.

### 1.4.8    PB0-PB7

These eight I/O lines comprise port B.  The state of any pin is software programmable and all port B lines are configured as input during power-on or reset.  For a detailed description on I/O programming refer to **5.4.1 INPUT/OUTPUT PROGRAMMING**. Three of the port B pins (PB5-PB7) are shared with the SIOP subsystem.  Refer to **SECTION 8  SIMPLE SERIAL INPUT/OUTPUT PORT** for a detailed description of the SIOP.

### 1.4.9    PC0-PC7

These eight I/O lines comprise port C.  The state of any pin is software programmable and all port C lines are configured as input during power-on or reset.  For a detailed description on I/O programming, refer to **5.4.1 INPUT/OUTPUT PROGRAMMING**.

### 1.4.10    PD0-PD7

These eight I/O lines comprise port D.  The state of any pin is software programmable and all port D lines are configured as input during power-on or reset.   For a detailed description on I/O programming, refer to **5.4.1 INPUT/OUTPUT PROGRAMMING**.  PD6 is shared with TCMP.  Refer to **SECTION 6  TIMER** for more information on the TCMP pin.

# SECTION 2        OPERATING MODES

The MCU has two modes of operation:  Single-chip Mode and Bootloader Mode.

**Table 2-1** shows the conditions required to go into each mode, where $V_{TST} = 2 \times V_{DD}$.

**Table 2-1:  Operating Mode Conditions**

| $\overline{RESET}$ | $\overline{IRQ}$ | TCAP | MODE |
|---|---|---|---|
| ⌐ | $V_{SS}$ to $V_{DD}$ | $V_{SS}$ to $V_{DD}$ | Single-chip |
| _⌐ | $V_{TST}$ | $V_{DD}$ | Boot |

## 2.1     SINGLE-CHIP MODE

In Single-Chip Mode, the address and data buses are not available externally, but there are four 8-bit I/O ports. This mode allows the microcontroller unit (MCU) to function as a self-contained microcontroller, with maximum use of the pins for on-chip peripheral functions. All address and data activity occurs within the MCU.  Single-Chip Mode is entered on the rising edge of $\overline{RESET}$, if the $\overline{IRQ}$ pin is within normal operating range.

www.DataSheet4U.com

| | | | | | |
|---|---|---|---|---|---|
| RESET | → | 1 | 40 | ← | V$_{DD}$ |
| IRQ | → | 2 | 39 | ← | OSC1 |
| PE | → | 3 | 38 | → | OSC2 |
| PA7 | ↔ | 4 | 37 | ← | TCAP |
| PA6 | ↔ | 5 | 36 | → | PD7 |
| PA5 | ↔ | 6 | 35 | → | PD6/TCMP |
| PA4 | ↔ | 7 | 34 | → | PD5 |
| PA3 | ↔ | 8 | 33 | → | PD4 |
| PA2 | ↔ | 9 | 32 | → | PD3 |
| PA1 | ↔ | 10 | 31 | → | PD2 |
| PA0 | ↔ | 11 | 30 | → | PD1 |
| PB0 | ↔ | 12 | 29 | → | PD0 |
| PB1 | ↔ | 13 | 28 | → | PC0 |
| PB2 | ↔ | 14 | 27 | → | PC1 |
| PB3 | ↔ | 15 | 26 | → | PC2 |
| PB4 | ↔ | 16 | 25 | → | PC3 |
| SDO/PB5 | ↔ | 17 | 24 | → | PC4 |
| SDI/PB6 | ↔ | 18 | 23 | → | PC5 |
| SCK/PB7 | ↔ | 19 | 22 | → | PC6 |
| V$_{SS}$ | → | 20 | 21 | → | PC7 |
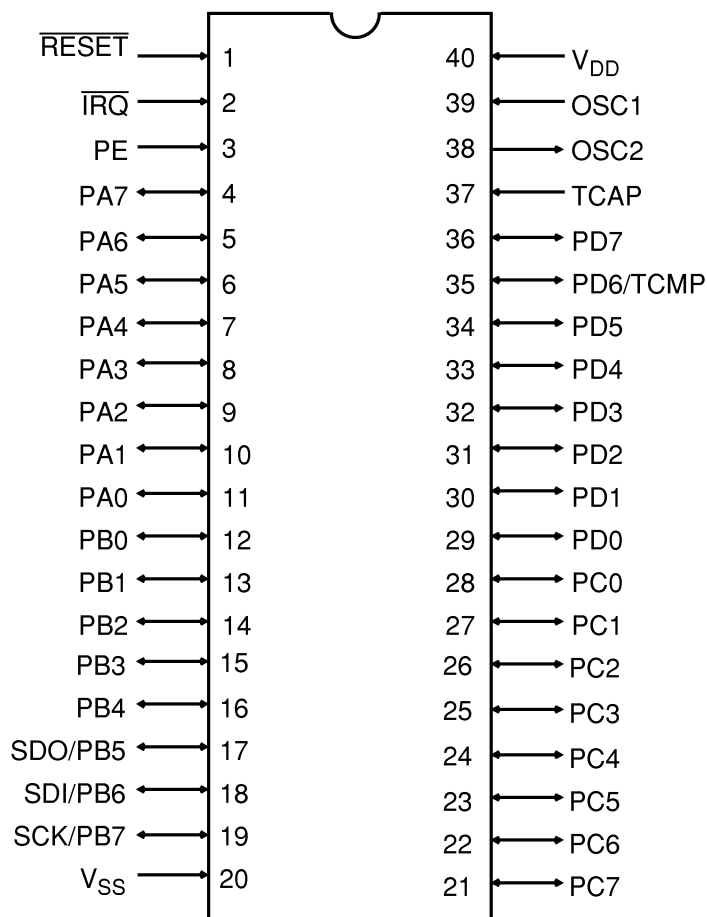
Figure 2-1:  Single-Chip Mode Pinout of the MC68HC705C5

## 2.2    BOOTLOADER MODE

Boot mode is used primarily for programming of the internal EPROM. User code resides either in an external EPROM (for example, 2764) or is driven from a host computer. The internal bootloader program of the 705C5 performs a programming pass then does a verify pass. If the source of code is in an external EPROM, the code residing in the external EPROM is copied into the internal eprom of the 705C5.

**Figure 2-2: Programmer Interface to Host** shows the timing required to interface the MC68HC705C5 being programmed to a host.  If the source of code is being driven, a host computer, a sync in and out is used.

Bootloader Mode is entered upon the rising edge of RESET if the IRQ pin is twice the V$_{DD}$ voltage and the TCAP pin is at logic one.  The Bootloader code resides where the Self-Check ROM is in the ROM device, from $1E81 to $1FEF. User code must have a 1-to-1 correspondence with the internal EPROM addresses.

---

NOTE:    The COP **MUST BE DISABLED** by the hardware in bootloader mode.

---

### 2.2.1 BOOTLOADER FUNCTIONS

Ports C and D are used to select among bootmode routines (for example, PROGRAM, PROGRAM/VERIFY, OR DUMP) and various types of testing routines. See **Table 2-3: Bootloader Mode Routines** and **Table 2-4: Bootloader Mode Test Routines**.

### 2.2.2 HOST INTERFACE

The host interface allows the user to program from a parallel port. Two Port D pins facilitate the handshaking. Port D 6 is generated from the MCU, Port D 7 is generated from the host's parallel port.

Three internal routines have the syncing capability. These are the DUMP, PROGRAM and VERIFY routines.

#### 2.2.2.1 DUMP

Initially, the MCU will hold its CLK pin high, signifying that data is invalid on Port B. CLK will then be driven to logic zero by the MCU when data is valid. The MCU poles the SYNC pin for a logic zero, signaling that the host has read the EPROM data on Port B.

The host should initialize its SYNC pin to a logic one to insure the first byte of data is not lost causing an offset of 1 byte for the data capture. When CLK goes low, the host reads the data on Port B. The host then drives SYNC to logic zero showing a read state. The MCU will drive CLK to logic one while it calculates the next address and reads the corresponding EPROM data byte. After writing the EPROM data to Port B, the MCU will drive CLK to logic zero.

There is no END OF ADDRESS REACHED SIGNAL other than the CLK will continue to be driven to logic one even if the SYNC pin is varied.

#### 2.2.2.2 PROGRAM

This routine is very much the same as the DUMP routine. The exception is that instead of the MCU driving Port B with the internal EPROM data, the host drives Port B with the data to be programmed. The initial requirement for the SYNC pin is the same. If not met, the internal data may be offset from the desired address locations.

#### 2.2.2.3 VERIFY

This routine is very much the same as the PROGRAM routine. The exception is that instead of the MCU programming the data, it compares data driven on Port B to internal EPROM data. If there is a mismatch, the routine hangs, thus never enabling the LED. If every location passes, the LED is enabled and the MCU is put into a low-power WAIT mode. The initial requirement for the SYNC pin is the same. If not met, the internal data may be offset from the desired address locations and a mismatch will occur immediately.

PORTB       DATA OUT

CLK (OUT) **PD6**

SYNC (IN) **PD7**

HOST READ DATA
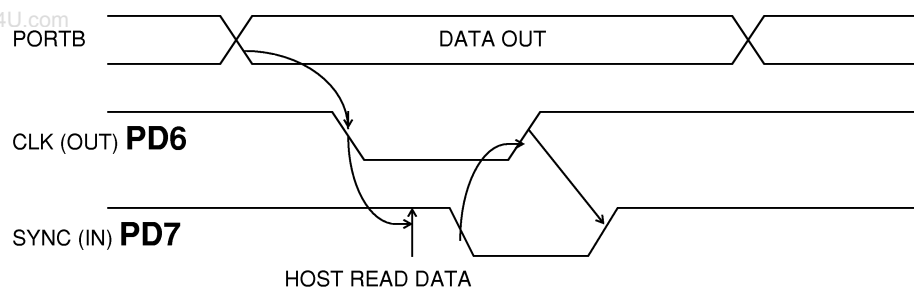
(a) DUMP EPROM INTERFACE TO A HOST

DATA READ

CLK (OUT) **PD6**

SYNC (IN) **PD7**

DATA IN
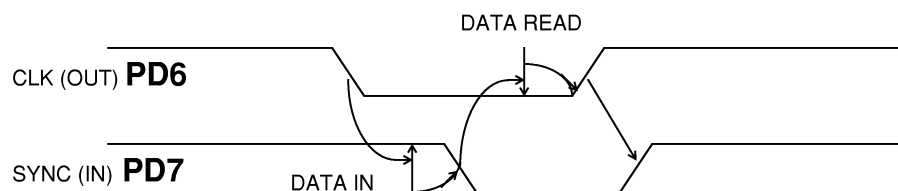
(b) PROGRAM/VERIFY INTERFACE TO A HOST

**Figure 2-2:  Programmer Interface to Host**

www.DataSheet4U.com

```
                              ┌─────────┐
                             (   START   )
                              └────┬────┘
                                   │
              YES          ◇─────────────◇          NO
        ┌──────────────────   PC6=1      ────────────────┐
        │                    ◇─────────────◇             │
        │                                          ┌──────────────────────┐
        │                                          │ See Table 2-3: Bootloader │
        │                                          │    Mode Routines      │
        │                                          └──────────────────────┘
  ┌──────┐
  │ GATE │
  └──────┘
     ▲
  NO │
     │        YES         ↓
◇─────────◇ ←──── ◇─────────◇  NO
│ PD2=1   │       │ PD5=1   │
◇─────────◇       ◇─────────◇
     │  YES          │
     ▼        STRESS  │
 ┌───────┐       ┌──────────────┐
 │ DRAIN │       │  INITIALIZE  │
 └───────┘       └──────────────┘
```
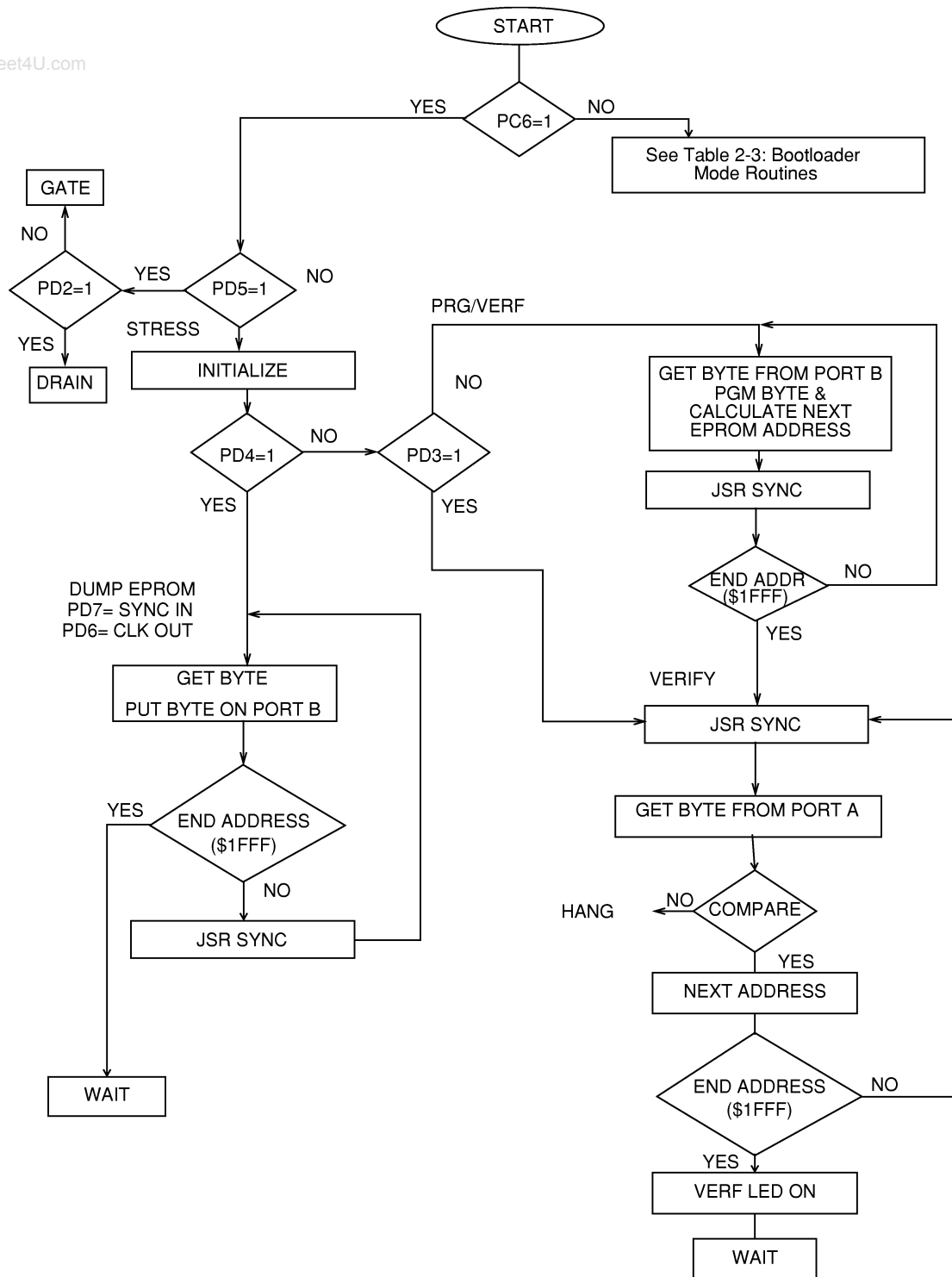
NOTE: PD7 must be Logic 0 if host interface NOT BEING USED.

**Figure 2-3:  MC68HC705C5 Bootloader Flowchart**

www.DataSheet4U.com

## Table 2-2: Vector Addresses for Interrupts and Reset

| Register | Flag Name | Interrupts | CPU Interrupt | Vector Address |
|---|---|---|---|---|
| N/A | N/A | Reset | RESET | $1FFE-$1FFF |
| N/A | N/A | Software | SWI | $1FFC-$1FFD |
| N/A | N/A | External Interrupt | IRQ | $1FFA-$1FFB |
| TSR | ICF | Timer Input Capture | TIMER | $1FF8-$1FF9 |
| TSR | OCF | Timer Output Compare | TIMER | $1FF8-$1FF9 |
| TSR | TOF | Timer Overflow | TIMER | $1FF8-$1FF9 |

## 2.3 OTPROM/EPROM (PROM) PROGRAMMING WITH THE MC68HC05PGMR

Programming of the internal PROM of the MC68HC705C5 MCU is most readily accomplished using the MC68HC05PGMR Programmer Board, available from Motorola. This printed circuit board (PCB) permits the user to program the MCU using an 8K EPROM device already programmed with user code (stand alone programming), or to download developed user code from a host computer using the board's terminal I/O port. For more information concerning the MC68HC705PGMR and its usages, contact your local Motorola representative for a copy of M68HC05PGMR/002.

## 2.4 STAND-ALONE PROGRAMMING

The programming of the PROM MCU in stand-alone mode is a matter of installing the MCU in the PCB, along with an 8K EPROM device programmed with user code, then subjecting it to a series of routines. The routines necessary to program and verify the PROM device are:

- Program and Verify PROM
- Verify PROM Contents (Only)

Another board routine available to the user is:

- Dump PROM Contents

The user first configures the MCU for the bootloader mode of operations by installing a fabricated jumper across pins 1 and 2 of the board mode select header, J1. Next, the board mode switches (S3, S4, S5, and S6) are set to determine the routine to be executed after the next reset, as shown in **Table 2-3: Bootloader Mode Routines** and **Table 2-4: Bootloader Mode Test Routines**.

## Table 2-3: Bootloader Mode Routines

| Routine | PC6 | PD5 | PD4 | PD3 | PD2 |
|---|---|---|---|---|---|
| TEST[1] | 0 | X | X | X | X |
| STRESS[2] | 1 | 1 | X | X | **** |
| DUMP | 1 | 0 | 1 | X | 1 |
| PROGRAM VERIFY | 1 | 0 | 0 | 0 | 1 |
| VERIFY ONLY | 1 | 0 | 0 | 1 | 1 |

1. See **Table 2-4: Bootloader Mode Test Routines**.

2. PORT D pin 2 (PD2) at logic level one selects the drain stress test. PD2 at logic level zero selects gate stress test.

## Table 2-4: Bootloader Mode Test Routines

| Routine | PC6 | PC3 | PC1 | PC0 |
|---|---|---|---|---|
| JUMP TO RAM+1 | 0 | 0 | X | 0 |
| JUMP TO EPROM | 0 | 1 | X | 0 |
| WRITE ERASE W/ GATE STRESS | 0 | X | 1 | 1 |
| WRITE ERASE W/O GATE STRESS | 0 | X | 0 | 1 |

### 2.4.1    BEFORE PROGRAMMING

Before attempting to program the PROM of the MC68HC705C5 using an MC68HC05PGMR PCB in stand-alone mode, the user should ensure that:

- A jumper is installed on pins 1 and 2 of mode select header J1.

- An 8k 2764-type EPROM has been programmed with the necessary user code.

- The erasure window (if any) of the device to be programmed is covered.

- $V_{DD}$ of +5 Vdc is available on the board.

- $V_{PP}$ of +15.5 Vdc $\leq\pm$ 0.25 Vdc, measured at connector P1 (pin 5) or $V_{PP}$ of +14.75 Vdc $\pm$0.25 Vdc, measured at socket U2 or U3 (pin 3) whichever socket is to be used, is available on the board.

### CAUTION:

If the $V_{PP}$ level at the MCU exceeds +16 Vdc, the MC68HC705C5 MCU will suffer permanent damage.

## 2.4.2    PREPROGRAMMING STEPS

---

NOTE:    No PROM MCU should be inserted in or removed from its board socket
(U2 or U3) while $V_{PP}$ (P1, slot 5) or $V_{DD}$ (switch 1) is active on the board.

---

Once the preceding conditions are met, the user should take the following steps before beginning programming:

1. Remove the $V_{PP}$ power source.

2. Set switch 1 in the OFF position (removes $V_{DD}$).

3. Place the programmed 8K 2764-type EPROM in socket U1.

4. Insert the erased PROM MCU device to be programmed in the proper socket:

    - MC68HC705C5 in socket U2 (40-pin DIP), or;

    - MC68HC705C5FN in socket U3 (44-pin PLCC), with the device notch at the upper right corner of the socket.

5. Set switch S2 in the RESET position.

## 2.4.3    PROGRAM AND VERIFY PROM

The program and verify PROM routine copies the contents of the external 8K EPROM into the MCU PROM, with direct correspondence between the addresses. Memory addresses in the 8K EPROM being copied should contain $00 bytes to speed up the programming process.

To run the program and verify PROM routine on the PROM MCU, take the following steps:

1. Set switch 1 in the ON position (restores $V_{DD}$).

2. Restore the $V_{PP}$ power source.

3. Set switches S2, S3, S4, S5 in the OFF position and S6 in the ON position (selects proper routine).

4. Set switch 2 in the OUT position (routine is activated).

The red light emitting diode (LED) is illuminated, showing that the programming part of the routine is running. The LED goes out when programming is finished. The verification part of the routine now begins. When the green LED is illuminated, verification is successfully completed and the routine is finished.

5. Set switch 2 in the RESET position.

At this point, if no other MCU is to be programmed, remove $V_{PP}$ power from the board. If another routine is to be performed on the MCU being programmed, the user can then set switches S3, S4, S5 and S6 to the positions necessary to select the next routine, and begin it by setting switch 2 to the OUT position. If no other routine is to be performed, remove $V_{DD}$ from the board and remove the MCU from the programming socket.

## 2.4.4    VERIFY PROM CONTENTS

The verify PROM contents routine is normally run automatically after the PROM is programmed. Direct entry to this routine causes the PROM contents of the MCU to be compared to the contents of the external memory locations of the 8K EPROM at the same addresses.

To invoke the verify PROM contents routine of the MCU, take the following steps:

1. Set switch 1 in the ON position (restores $V_{DD}$).

2. Connect $V_{PP}$ to $V_{DD}$.

3. Set switches S2, S3, and S4 to the OFF position.

4. Set S5 and S6 in the ON position.

5. Set S2 in the OUT position (routine is activated).

The red LED is not illuminated during this routine, since no programming takes place. If verification fails, the routine halts with the failing address on the external memory bus. When the green LED is illuminated, verification is successfully completed and the routine is finished.

6. Set S2 in the RESET position.

At this point, if another routine is to be performed on the MCU being programmed, the user can set switches S3, S4, S5 and S6 to the positions necessary to select the next routine, and move switch S2 to the OUT position to start it. If no other routine is to be performed, remove $V_{DD}$ from the board and remove the MCU from the programming socket.

### 2.4.4.1    MASK OPTION REGISTER (MOR) $1E80

This register is latched upon RESET. It is an EPROM byte located at $1E80 and holds several option bits for the SIOP data format, interrupt sensitivity, LVPI and COP enable. See **Figure 2-4: Mask Option Register**.

| $1E80 | -- | -- | -- | -- | LVPI | SIOP | IRQ | COP |
|---|---|---|---|---|---|---|---|---|

Figure 2-4:  Mask Option Register

Reading and writing the MOR is done according to **Table 2-4**.

**Table 2-5:  MOR Read/Write Based on Mode and LATCH Bit**

| LATCH | TEST | READ | WRITE |
|-------|------|------|-------|
| 0 | 0 | E | N |
| 1 | 0 | N | PE |
| 0 | 1 | E | R |
| 1 | 1 | R | PE |

PE = Program EPROM Byte
R   = Register Latch
N   = Nothing
E   = EPROM Byte

LVPI - Low Voltage Program Inhibit
    READ: Any time.
    WRITE:  In special modes: Writing this bit activates the function without having
        to program it and can be written many times.
        In user mode: Writing has no effect.  It has to be programmed.

    0 - (erased state) Disable LVPI circuitry.
    1 - Enable LVPI circuitry

SIOP - SIOP MSB or LSB first
    READ:  Any time.
    WRITE:  In special modes: Writing this bit activates the function without having
        to program it and can be written many times.
        In user mode: Writing has no effect.  It has to be programmed.

    0 - (erased state) The SIOP sends/receives MSB (bit 7) first.
    1 - The SIOP sends/receives LSB (bit 0) first.

IRQ - Interrupt Request option
    READ:  Any time.
    WRITE:  In special modes: Writing this bit activates the function without having
        to program it and can be written many times.
        In user mode: Writing has no effect.  It has to be programmed.

    0 - (erased state) The $\overline{\text{IRQ}}$ pin is Edge-sensitive.
    1 - The $\overline{\text{IRQ}}$ pin is Edge- and Level-sensitive.

COP - COP enable/disable
        READ:  Anytime.
        WRITE: In special modes: Writing this bit activates the function without having
                    to program it and can be written many times.
                In user mode: Writing has no effect.  It has to be programmed.


        0 - (erased state) The COP is disabled.
        1 - The COP is enabled.

Section 2: Operating Modes

# SECTION 3        MEMORY

The MC68HC705C5 has an 8K-byte memory map consisting of user EPROM, user RAM, Boot ROM, EEPROM, and I/O.  See **Figure 3-1** and **Figure 3-5.**

| | | |
|---|---|---|
| $0000 | I/O 32 Bytes | 0000 |
| $0020 | User EPROM 48 Bytes | 0032 |
| $0050 | RAM 176 Bytes | 0080 |
| $00C0 | Stack 64 Bytes | 0192 |
| $0100 | EEPROM 128 Bytes | 0256 |
| $0180 | Unused 2304 Bytes | 0384 |
| $0A80 | User EPROM 5120 Bytes | 2688 |
| $1E80 | MOR (EPROM) | 7808 |
| $1E81 | Boot ROM 367 Bytes | 7809 |
| $1FE0 | Boot Vectors | 8160 |
| $1FF0 | User Vectors (EPROM) 16 Bytes | 8176 |
| $1FFF | | 8191 |
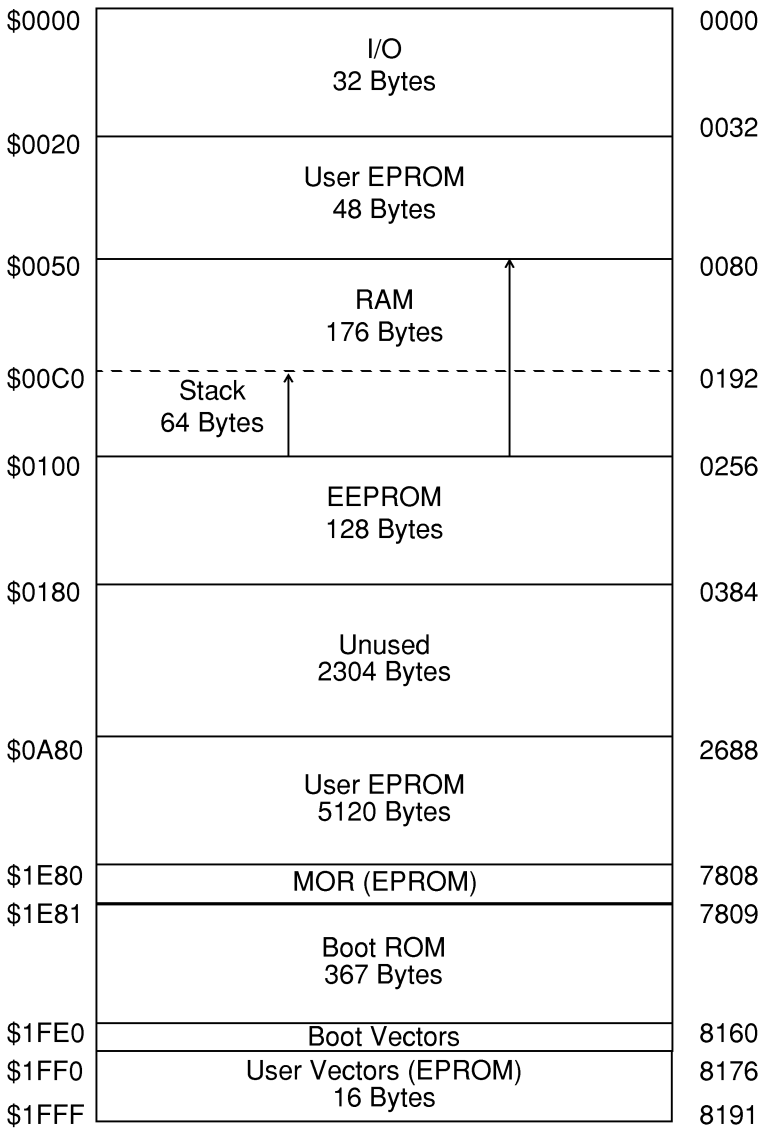
**Figure 3-1:  The 8K Memory Map of the MC68HC705C5**

NOTE:    See **Table 4-1: Vector Address for Interrupts and Reset**.

## 3.1 EPROM

The user EPROM consists of 48 bytes of page zero EPROM from $0020 to $004F, 5120 bytes of EPROM from $0A80 to $1E7F and 16 bytes of user vectors from $1FF0 to $1FFF. The Boot ROM and vectors are located from $1E81 to $1FEF.

Eight of the user vectors, $1FF8 through $1FFF, are dedicated to reset and interrupt vectors. The remaining eight locations, $1FF0 through $1FF7, are general purpose user EPROM locations.

### 3.1.1 EPROM ERASING

In the MC68HC705C5, the erased state of an OTPROM or EPROM byte is $00. EPROM devices can be erased by exposure to a high intensity ultraviolet (UV) light via the EPROM window with a wavelength of 2537 A. The recommended erasure dosage (UV intensity on a given surface area X exposure time) is 15 W/cm$^2$. UV lamps should be used without shortwave filters, and the EPROM device should be positioned about 1 inch (2.5 cm) from the UV source.

---

NOTE: EEPROM contents are not guaranteed after exposure to UV lamps. If EPROM is erased using UV lamps, the user should reprogram the EEPROM.

---

### 3.1.2 EPROM REGISTERS

#### 3.1.2.1 PROGRAMMING REGISTER (PROG) $1D

This register is used to program the EPROM array. Only the LATCH and EPGM bits are available in user mode. To program a byte of EPROM, set LATCH, then write data to the desired address, then set EPGM for $t_{EPGM}$. EPTST, MBE, TS1, and TS0 are available only in the special modes (BOOT and TEST).

| $001D | 0 | EPTST | MBE | TS1 | TS0 | LATCH | 0 | EPGM |
|-------|---|-------|-----|-----|-----|-------|---|------|
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Figure 3-2: PROG Register

EPTST - EPROM test mode enable
    READ:  Not readable, returns zero.
    WRITE:  Only in special modes.

    1 - Enables the functions of  TS1 and TS0.
    0 - TS1 and TS0 functions are disabled.

MBE - Multiple Byte Program enable
    READ:  Not readable, returns zero
    WRITE: Only in special modes

    0 - Normal programming
    1 - Program 2 bytes with the same data; address bit 4 is don't care
    so that the bytes with address 4 = 0 and = 1 both get programmed.

TS1,TS0   EPROM test select bits
    READ: Not readable, returns zero
    WRITE: Only in special modes

    These bits have effect only when EPTST is set.  They are used to select the
    extra rows, extra columns, perform gate stress, and drain stress.

    GATE STRESS holds the drain at $V_{SS}$ and $V_{PP}$ is applied to all of the control gates.

    DRAIN STRESS holds the control gate at  $V_{SS}$ and applies $V_{PP}$ to all of the drains.
    GATE and DRAIN stresses are performed on the entire array.

    To initiate gate or drain test, the mode is selected, and then the EPROM
    programming sequence is executed.  The data written to the array in drain stress
    mode MUST be $FF.

    SELECT EXTRA COLUMNS disables the user array and enables access to the
    extra columns.  A bit in the extra column is accessed by addressing the row which
    contains the bit.  When the extra columns are selected, the user array and vectors
    are disabled.  The extra columns are for bit 0 and bit 7 only.  Bits 1 through 6 read
    as ones when in this mode.  The extra columns are accessed by addressing the
    EPROM.  The same extra column bits are accessed for a block of 32 addresses.
    The MOR also is not readable in this mode and must not be addressed because it
    will interfere with the extra column data.

SELECT EXTRA ROW disables the user array and enables access to the extra rows. A bit in the extra row is accessed by addressing the column which contains the bit. When the extra row is selected, the user array and vectors are disabled. The extra row is accessed by addressing the EPROM. The 32 bytes of this row are redundantly mapped throughout the EPROM address space. The MOR is also not readable in this mode and must not be addressed because it will interfere with the extra row data.

The extra columns and row are located at the opposite side of the array from the row drivers and the sense amplifiers.

LATCH - EPROM Latch Control

READ: Anytime.

WRITE: Anytime.

0 - EPROM address and data bus configured for normal reads.

1 - EPROM address and data bus configured for programming. Causes address and data bus to be latched when a write to EPROM is done. EPROM cannot be read if LATCH = 1.

EPGM - EPROM Program Control

READ: Anytime

WRITE: Cleared anytime, Set only if LATCH=1

0 - Programming power switched off the EPROM array

1 - Programming power switched on to the EPROM array. If LATCH=0, EPGM is automatically cleared. EPGM cannot be set if LATCH is not set. LATCH and EPGM can not both be set on the same write.

## 3.2    RAM

The user RAM consists of 176 bytes of a shared stack area. The stack begins at address $00FF. The stack pointer can access 64 bytes of RAM in the range $00FF to $00C0. See **4.1.4 STACK POINTER (SP)**.

---

NOTE:    Using the stack area for data storage or temporary work locations requires care to prevent it from being overwritten due to stacking from an interrupt or subroutine call.

---

## 3.3    EEPROM

The EEPROM on this device is 128 bytes long and is located at address $0100. The erased state of an EEPROM byte is $FF. Programming the EEPROM can be done by the user on a single-byte basis by manipulating the Programming Register, located at address $001C.

### 3.3.1    PROGRAMMING REGISTER $1C

The contents and use of the programming register are discussed below. This device includes low-voltage programming inhibit (LVPI) circuitry which inhibits the use of the programming register when the supply voltage ($V_{DD}$) falls below $V_{LVPI}$.

| $1C | LVPI | CPEN | 0 | ER1 | ER0 | LATCH | EERC | EEPGM |
|---|---|---|---|---|---|---|---|---|
| RESET: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 3-3:  Programming Register**

#### 3.3.1.1    LVPI - Low-Voltage Programming Inhibit

LVPI is automatically set and cleared by the LVPI circuit and is not writable. The bit is set when $V_{DD}$ falls below $V_{LVPI}$ and is cleared when $V_{DD}$ is above $V_{LVPR}$. Note that the $V_{DD}$ rise and fall slew rates ($t_{VDDR}$ and $t_{VDDF}$) must be within the specification for proper LVPI operation. If the specification is not met, the circuit will operate properly following a delay of $V_{DD}$/Slew rate. When set, LVPI clears bits 0-6 in the programming register to disable the charge pump and prevent programming.

CPEN cannot be set when LVPI is set. During reset, LVPI is set until $V_{DD}$ reaches $V_{LVPI}$, at which time it is cleared. The LVPI circuitry continues to function while the processor is in STOP mode.

The LVPI function is controlled by a bit in the MOR register. If this function is disabled, bit 7 will be set to a value of "0".

#### 3.3.1.2    CPEN - Charge Pump Enable

When set, CPEN enables the charge pump which produces the internal EEPROM programming voltage. This bit should be set with the LATCH bit. The programming voltage will not be available until EEPGM is set. The charge pump should be disabled when not in use. This bit is automatically cleared by the LVPI circuit when LVPI is set, and cannot be set until LVPI is cleared. CPEN is readable and writable and is cleared by reset.

#### 3.3.1.3    ER1:ER0 - Erase Select Bits

ER1 and ER0 form a 2-bit field which is used to select one of three erase modes:  byte, block, or bulk. **Table 3-1: Erase Mode Select** shows the modes selected for each bit configuration. These bits are automatically cleared when LVPI is set. These bits are readable and writable and are cleared by reset.

Section 3:  Memory

MOTOROLA

In byte erase mode, only the selected byte is erased. In block mode, a 32-byte block of EEPROM is erased. The EEPROM memory space is divided into four 32-byte blocks ($100-$11F, $120-$13F, $140-$15F, $160-$17F), and doing a block erase to any address within a block will erase the entire block. In bulk erase mode, the entire 128-byte EEPROM section is erased.

**Table 3-1: Erase Mode Select**

| ER1 | ER0 | MODE |
|-----|-----|------|
| 0 | 0 | Program (No Erase) |
| 0 | 1 | Byte Erase |
| 1 | 0 | Block Erase |
| 1 | 1 | Bulk Erase |

### 3.3.1.4 LATCH

When set, LATCH configures the EEPROM address and data bus for programming. When LATCH is set, writes to the EEPROM array cause the data bus and the address bus to be latched. This bit is readable and writable, but reads from the array are inhibited if the LATCH bit is set and a write to the EEPROM space has taken place. When clear, address and data buses are configured for normal operation. LATCH is automatically cleared when LVPI is set. Reset clears this bit.

### 3.3.1.5 EERC - EEPROM RC Oscillator Control

When this bit is set, the EEPROM section uses the internal RC oscillator instead of the CPU clock. After setting the EERC bit, delay a time $t_{RCON}$ to allow the RC oscillator to stabilize.

This bit is readable and writable and should be set by the user when the internal bus frequency falls below 1.5 MHz. EERC is automatically cleared when LVPI is set. Reset clears this bit.

### 3.3.1.6 EEPGM - EEPROM Programming Power Enable

EEPGM must be written to enable (or disable) the EEPGM function. When set, EEPGM turns on the charge pump and enables the programming (or erasing) power to the EEPROM array. When clear, this power is switched off. This will enable pulsing of the programming voltage to be controlled internally. This bit can be read at any time, but can only be written to if LATCH = 1. If LATCH is not set, then EEPGM cannot be set. LATCH and EEPGM cannot both be set with one write if LATCH is cleared. EEPGM is automatically cleared when LVPI is set. EEPGM is automatically cleared when LATCH is cleared. Reset clears this bit.

### 3.3.1.7 PROGRAMMING/ERASING PROCEDURES

To program a byte of EEPROM, set LATCH = CPEN = 1, set ER1 = ER0 = 0, write data to the desired address and then set EEPGM for a time $t_{EPGM}$.

---

NOTE: Any bit should be erased before it is programmed. However, if write/erase cycling is a concern, a procedure can be followed to minimize the cycling of each bit in each EEPROM byte. Here is the procedure:

- If PB•EB* = 0, program the new data over the existing data without erasing it first.

- If PB•EB* ≠ 0, erase byte before programming

- Where PB = Byte data to be programmed and EB = Existing EEPROM byte data

---

To erase a **byte** of EEPROM, set LATCH = 1, CPEN = 1, ER1 = 0 and ER0 = 1, write to the address to be erased, and set EEPGM for a time $t_{EBYT}$.

To erase a **block** of EEPROM, set LATCH = 1, CPEN = 1, ER1 = 1 and ER0 = 0, write to any address in the block, and set EEPGM for a time $t_{EBLOCK}$.

For a **bulk** erase, set LATCH = 1, CPEN = 1, ER1 = 1, and ER0 = 1, write to any address in the array, and set EEPGM for a time $t_{EBULK}$.

To terminate the programming or erase sequence, clear EEPGM, delay for a time $t_{FPV}$ to allow the program voltage to fall, and then clear LATCH and CPEN to free up the buses. Following each erase or programming sequence, clear all programming control bits.

| ADDRESS 0000 TO 001F | DATA | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| $00 PORT A DATA | | | | | | | | |
| $01 PORT B DATA | | | | | | | | |
| $02 PORT C DATA | | | | | | | | |
| $03 PORT D DATA | | | | | | | | |
| $04 PORT A DDR | | | | | | | | |
| $05 PORT B DDR | | | | | | | | |
| $06 PORT C DDR | | | | | | | | |
| $07 PORT D DDR | | | | | | | | |
| $08 UNUSED | -- | -- | -- | -- | -- | -- | -- | -- |
| $09 UNUSED | -- | -- | -- | -- | -- | -- | -- | -- |
| $0A SERIAL CTRL | 0 | SPE | 0 | MSTR | CPOL | 0 | 0 | 0 |
| $0B SERIAL STAT | SPIF | DCOL | 0 | 0 | 0 | 0 | 0 | 0 |
| $0C SERIAL DATA | | | | | | | | |
| $0D UNUSED | -- | -- | -- | -- | -- | -- | -- | -- |
| $0E UNUSED | -- | -- | -- | -- | -- | -- | -- | -- |
| $0F UNUSED | -- | -- | -- | -- | -- | -- | -- | -- |
| $10 UNUSED | -- | -- | -- | -- | -- | -- | -- | -- |
| $11 UNUSED | -- | -- | -- | -- | -- | -- | -- | -- |
| $12 TIMER CONTROL | ICIE | OCIE | TOIE | 0 | 0 | COE | IEDG | OLVL |
| $13 TIMER STATUS | ICF | OCF | TOF | 0 | 0 | 0 | 0 | 0 |
| $14 CAPTURE HIGH | | | | | | | | |
| $15 CAPTURE LOW | | | | | | | | |
| $16 COMPARE HIGH | | | | | | | | |
| $17 COMPARE LOW | | | | | | | | |
| $18 COUNTER HIGH | | | | | | | | |
| $19 COUNTER LOW | | | | | | | | |
| $1A DUAL TM HIGH | | | | | | | | |
| $1B DUAL TM LOW | | | | | | | | |
| $1C PROGRAM REG | LVPI | CPEN | 0 | ER1 | ER0 | LATCH | EERC | EEPGM |
| $1D PROG | -- | EPTST | MBE | TS1 | TS0 | LATCH | 0 | EPGM |
| $1E UNUSED | -- | -- | -- | -- | -- | -- | -- | -- |
| $1F TEST REGISTER | -- | MSCAN | ROMON | IOOFF | -- | -- | TCNT | RAMON |

**Figure 3-4: I/O Registers**

# SECTION 4                    CPU CORE

## 4.1    REGISTERS

The MCU contains five registers, as shown in the programming models of **Figure 4-1** and **Figure 4-2**. The interrupt stacking order is shown in **Figure 4-2: Stacking Order**.
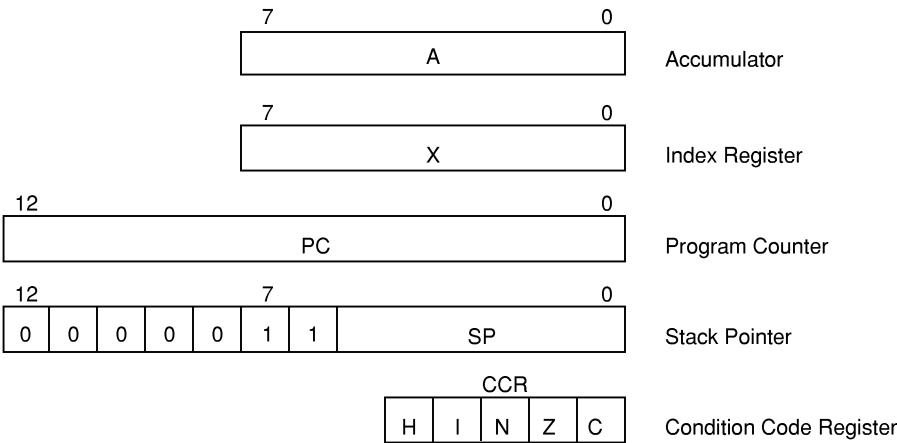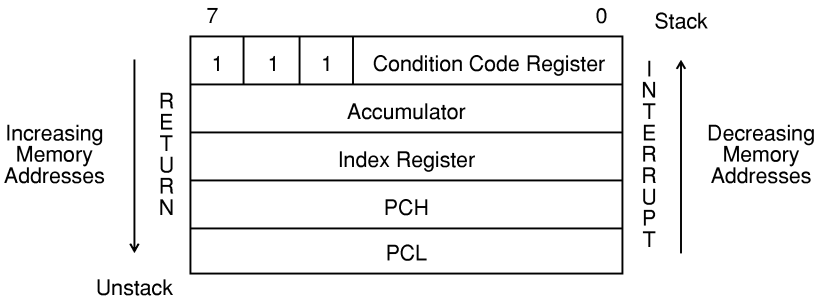


**Figure 4-1:  Programming Model**



**Figure 4-2:  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.

---

www.DataSheet4U.com

```
                    ┌─────────┐              ┌─────────┐
                    │  STOP   │              │  WAIT   │
                    └────┬────┘              └────┬────┘
                         │                        │
              ┌──────────┴──────────┐  ┌──────────┴──────────┐
              │  Stop Oscillator    │  │  Oscillator Active  │
              │  And All Clocks     │  │  Timer Clock Active │
              │                     │  │ Processor Clocks    │
              │  Clear I Bit        │  │   Stopped           │
              │                     │  │  Clear I Bit        │
              └─────────────────────┘  └─────────────────────┘
```
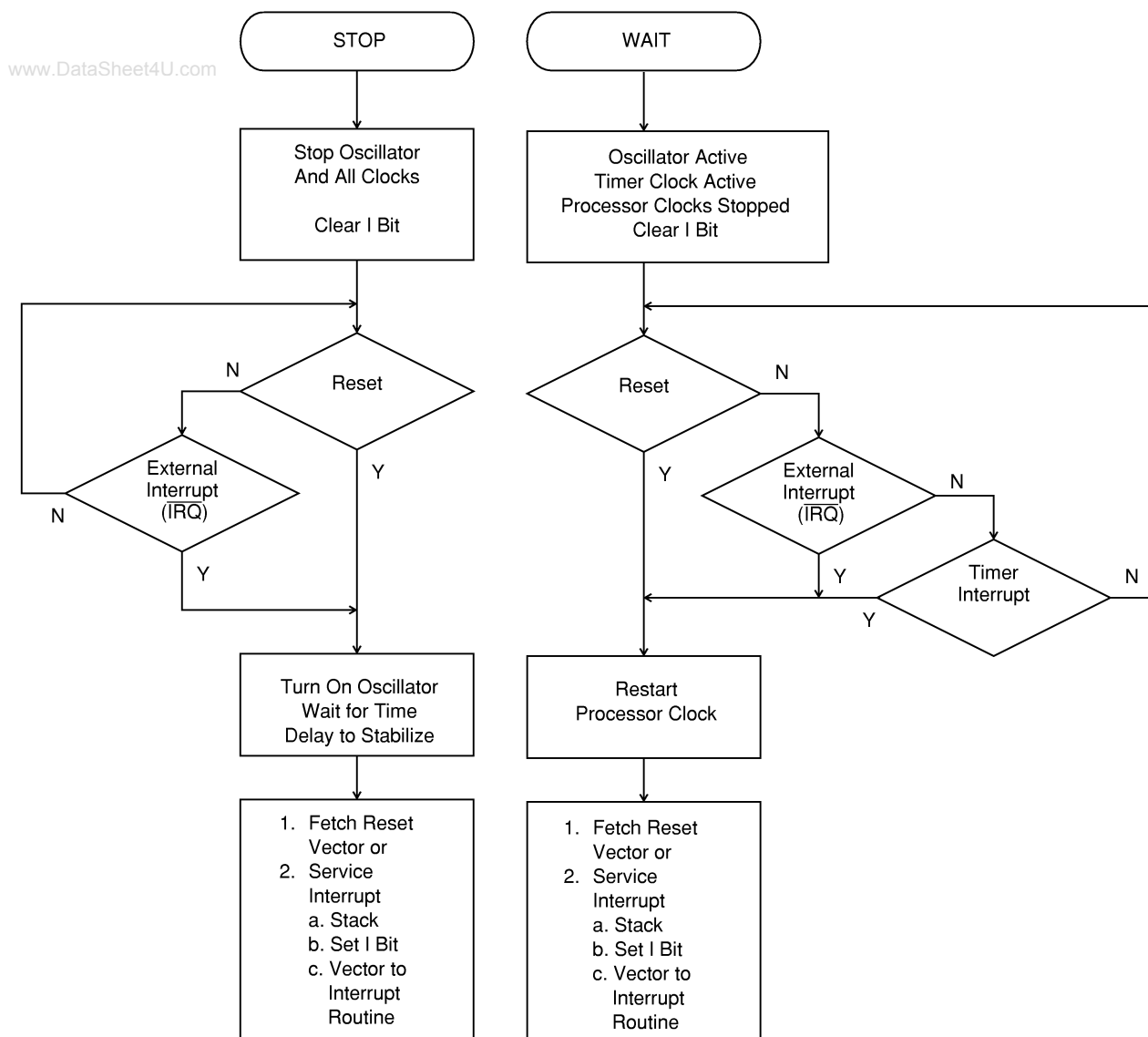
**Figure 4-3: STOP/WAIT Flowcharts**

## 4.1.1    ACCUMULATOR (A)

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.

## 4.1.2    INDEX REGISTER (X)

The index register is an 8-bit register used by the indexed addressing value to create an effective address.  The index register may also be used as a temporary storage area.

## 4.1.3    PROGRAM COUNTER (PC)

The program counter is a 13-bit register that contains the address of the next byte to be fetched.

www.DataSheet4U.com

### 4.1.4 STACK POINTER (SP)

The stack pointer contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set to location $0FF. 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 seven most significant bits are permanently set to 0000011. These seven bits are appended to the six least significant register bits to produce and address within the range of $00FF to $00C0. Subroutines and interrupts may use up to 64 (decimal) locations. If 64 locations are exceeded, the stack pointer wraps around and loses the previously stored information. A subroutine call occupies two locations on the stack; an interrupt uses five locations.

### 4.1.5 CONDITION CODE REGISTER (CCR)

The CCR is a 5-bit register in which 4 bits are used to indicate the results of the instruction just executed, and the fifth bit indicates whether interrupts are masked. These bits can be individually tested by a program, and specific actions can be taken as a result of their state. Each bit is explained in the following paragraphs.

#### 4.1.5.1 Half Carry (H)

This bit is set during ADD and ADC operations to indicate that a carry occurred between bits 3 and 4.

#### 4.1.5.2 Interrupt (I)

When this bit is set, the timer and external interrupt is masked (disabled). If an interrupt occurs while this bit is set, the interrupt is latched and processed as soon as the interrupt bit is cleared.

#### 4.1.5.3 Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was negative.

#### 4.1.5.4 Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation was zero.

#### 4.1.5.5 Carry/Borrow (C)

When set, this bit indicates that a carry or borrow out of the arithmetic logical unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions and during shifts and rotates.

## 4.2 INSTRUCTION SET

The MCU has a set of 62 basic instructions. They can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type. For more information on the instruction set, refer to the *M6805 Family User's Manual* (M6805UM/AD2) or the *MC68HC05C4 Technical Data* (MC68HC05C4/D).

Section 4: CPU Core

MOTOROLA

### 4.2.1 REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following instruction list.

| Function | Mnemonic |
|---|---|
| Load A from Memory | LDA |
| Load X from Memory | LDX |
| Store A in Memory | STA |
| Store X in Memory | STX |
| Add Memory to A | ADD |
| Add Memory and Carry to A | ADC |
| Subtract Memory | SUB |
| Subtract Memory from A with Borrow | SBC |
| AND Memory to A | AND |
| OR Memory with A | ORA |
| Exclusive OR Memory with A | EOR |
| Arithmetic Compare A with Memory | CMP |
| Arithmetic Compare X with Memory | CPX |
| Bit Test Memory with A (Logical Compare) | BIT |
| Jump Unconditional | JMP |
| Jump to Subroutine | JSR |
| Multiply | MUL |

## 4.2.2 READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Do not use these read-modify-write instructions on write-only locations. Refer to the following list of instructions.

| Function | Mnemonic |
|---|---|
| Increment | INC |
| Decrement | DEC |
| Clear | CLR |
| Complement | COM |
| Negate (Twos Complement) | NEG |
| Rotate Left Through Carry | ROL |
| Rotate Right Through Carry | ROR |
| Logical Shift Left | LSL |
| Logical Shift Right | LSR |
| Arithmetic Shift Right | ASR |
| Test for Negative or Zero | TST |

### 4.2.3 BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are 2-byte instructions. Refer to the following list for branch instructions.

| Function | Mnemonic |
|---|---|
| Branch Always | BRA |
| Branch Never | BRN |
| Branch if Higher | BHI |
| Branch if Lower or Same | BLS |
| Branch if Carry Clear | BCC |
| Branch if Higher or Same | BHS |
| Branch if Carry Set | BCS |
| Branch if Lower | BLO |
| Branch if Not Equal | BNE |
| Branch if Equal | BEQ |
| Branch if Half Carry Clear | BHCC |
| Branch if Half Carry Set | BHCS |
| Branch if Plus | BPL |
| Branch if Minus | BMI |
| Branch if Interrupt Mask Bit is Clear | BMC |
| Branch if Interrupt Mask Bit is Set | BMS |
| Branch if Interrupt Line is Low | BIL |
| Branch if Interrupt Line is High | BIH |
| Branch to Subroutine | BSR |

### 4.2.3.1 BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any writable bit which resides in the first 256 bytes of the memory space where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 256 locations. The bit set, bit clear, and bit test and branch functions are all implemented with a single instruction. For test and branch instructions, the value of the bit tested is also placed in the carry bit of the condition code register. These instructions are also read-modify-write instructions. Do not bit manipulate write-only locations. Refer to the following list for bit manipulation instructions.

| Function | Mnemonic |
|---|---|
| Branch if Bit n is Set | BRSET n (n = 0. . .7) |
| Branch if Bit n is Clear | BRCLR n (n = 0. . .7) |
| Set Bit n | BSET n (n = 0. . .7) |
| Clear Bit n | BCLR n (n = 0. . .7) |

## 4.2.4 CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to the following list for control instructions.

| Function | Mnemonic |
|---|---|
| Transfer A to X | TAX |
| Transfer X to A | TXA |
| Set Carry Bit | SEC |
| Clear Carry Bit | CLC |
| Set Interrupt Mask Bit | SEI |
| Clear Interrupt Mask Bit | CLI |
| Software Interrupt | SWI |
| Return from Subroutine | RTS |
| Return from Interrupt | RTI |
| Reset Stack Pointer | RSP |
| No-Operation | NOP |
| Stop | STOP |
| Wait | WAIT |

## 4.3 ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single-byte instructions; the longest instructions (3 bytes) permit accessing tables throughout memory. Short and long absolute addressing is also included. One or 2-byte direct addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory.

The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

### 4.3.1 IMMEDIATE

In the immediate addressing mode, the operand is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution (for example, a constant used to initialize a loop counter).

### 4.3.2 DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single 2-byte instruction.

### 4.3.3 EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the 2 bytes following the opcode byte. Instructions with extended addressing mode are capable of referencing arguments anywhere in memory with a single 3-byte instruction. When using the Motorola assembler, the user need not specify whether an instruction uses direct or extended addressing. The assembler automatically selects the shortest form of the instruction.

### 4.3.4 RELATIVE

The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed offset byte, which is the last byte of the instruction, is added to the PC if, and only if, the branch conditions are true. Otherwise control proceeds to the next instruction. The span of relative addressing is from -128 to +127 from the address of the next opcode. The programmer need not calculate the offset when using the Motorola assembler, since it calculates the proper offset and checks to see that it is within the span of the branch.

### 4.3.5 INDEXED, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. This addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is often used to

move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

### 4.3.6 INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the unsigned byte following the opcode.

This addressing mode is useful for selecting the $K^{th}$ element in an n element table. With this 2-byte instruction, K would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE). This is the last location which can be accessed in this way.

### 4.3.7 INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the 2 unsigned bytes following the opcode. This address mode can be used in a manner similar to indexed, 8-bit offset except that this 3-byte instruction allows tables to be anywhere in memory. As with direct and extended addressing, the Motorola assembler determines the shortest form of indexed addressing.

### 4.3.8 BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode, and the byte following the opcode specifies the direct addressing of the byte in which the specified bit is to be set or cleared. Any read/write bit in the first 256 locations of memory, including I/O, can by selectively set or cleared with a single 2-byte instruction.

### 4.3.9 BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit that is to be tested and its condition (set or clear), is included in the opcode. The address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single 3-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from -128 to +127 from the address of the next opcode. The state of the tested bit is also transferred to the carry bit of the condition code register.

### 4.3.10 INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register and/or accumulator as well as the control instructions with no other arguments are included in this mode. These instructions are 1 byte long.

## 4.4 RESETS

The MCU can be reset three ways: by the initial power-on reset function, by an active low input to the RESET pin, and by a COP watchdog-timer reset.

### 4.4.1 POWER-ON RESET (POR)

An internal reset is generated on power-up to allow the internal clock generator to stabilize. The power-on reset is strictly for power turn-on conditions and should not be used to detect a drop in the power supply voltage. There is a 4064 internal processor clock cycle ($t_{CYC}$) oscillator stabilization delay after the oscillator becomes active. If the RESET pin is low at the end of this 4064-cycle delay, the MCU will remain in the reset condition until RESET goes high.



**Figure 4-4: Power-On Reset and RESET**
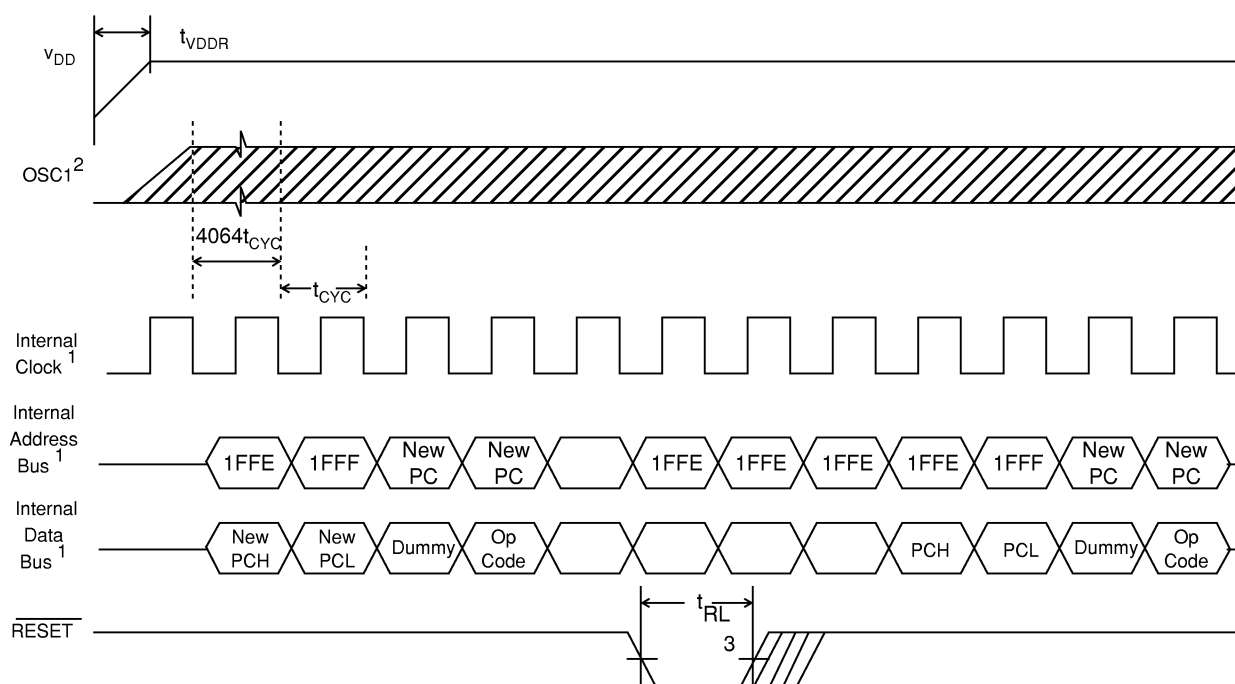
NOTES:
  1. Internal timing signal and bus information not available externally.
  2. OSC1 line is not meant to represent frequency. It is only used to represent time.
  3. The next rising edge of the internal processor clock following the rising edge of RESET initiates the reset sequence.

### 4.4.2 RESET PIN

The MCU is reset when a logic zero is applied to the RESET input for a period of one and one-half machine cycles ($t_{CYC}$).

### 4.4.3    COMPUTER OPERATING PROPERLY (COP) RESET

The MCU contains a watchdog timer that automatically times out if not reset (cleared) within a specific time by a program reset sequence. If the COP watchdog timer is allowed to time-out, an internal reset is generated to reset the MCU. Because the internal $\overline{RESET}$ signal is used, the MCU comes out of a COP reset in the same operating mode it was in when the COP time-out was generated.

The COP reset function is enabled or disabled by a bit in the MOR register. Refer to **SECTION 7  COMPUTER OPERATING PROPERLY** for more information on the COP Watchdog timer.

## 4.5    INTERRUPTS

The MCU can be interrupted three different ways: the two maskable hardware interrupts ($\overline{IRQ}$ and timer) and the non-maskable software interrupt instruction (SWI).

Interrupts cause the processor to save register contents on the stack and to set the interrupt mask (I bit) to prevent additional interrupts. The RTI instruction causes the register contents to be recovered from the stack and normal processing to resume.

Unlike RESET, hardware interrupts do not cause the current instruction execution to be halted, but are considered pending until the current instruction is complete.

---

NOTE:    The current instruction is the one already fetched and being operated on.

---

When the current instruction is complete, the processor checks all pending hardware interrupts. If interrupts are not masked (CCR I bit clear) and if the corresponding interrupt enable bit is set, the processor proceeds with interrupt processing; otherwise, the next instruction is fetched and executed.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction, regardless of the I-bit state.

**Table 4-1** lists vector addresses for all interrupts including reset.

## Table 4-1: Vector Address for Interrupts and Reset

| Register | Flag Name | Interrupts | CPU Interrupt | Vector Address |
|----------|-----------|------------|---------------|----------------|
| N/A | N/A | Reset | RESET | $1FFE-$1FFF |
| N/A | N/A | Software | SWI | $1FFC-$1FFD |
| N/A | N/A | External Interrupt | IRQ | $1FFA-$1FFB |
| TSR | ICF | Timer Input Capture | TIMER | $1FF8-$1FF9 |
| TSR | OCF | Timer Output Compare | TIMER | $1FF8-$1FF9 |
| TSR | TOF | Timer Overflow | TIMER | $1FF8-$1FF9 |

### 4.5.1    HARDWARE CONTROLLED INTERRUPT SEQUENCE

The following three functions ($\overline{\text{RESET}}$, STOP, and WAIT) are not in the strictest sense an interrupt; however, they are acted upon in a similar manner.  Flowcharts for hardware interrupts are shown in **Figure 4-5: Interrupt Flowchart** and **Figure 4-2: Stacking Order**

. A discussion is provided below.

1. $\overline{\text{RESET}}$ - A low input on the $\overline{\text{RESET}}$ input pin 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.  Much of the MCU is configured to a known state during this type of reset as previously described in **4.4 RESETS**.

2. STOP -  The STOP instruction causes the oscillator to be turned off and the processor to "sleep" until an external interrupt ($\overline{\text{IRQ}}$) or reset occurs.

3. WAIT - The WAIT instruction causes all processor clocks to stop, but leaves the timer clock running.  This "rest" state of the processor can be cleared by reset, an external interrupt ($\overline{\text{IRQ}}$), or Timer interrupt.  There are no special wait vectors for these individual interrupts.

### 4.5.2    SOFTWARE INTERRUPT (SWI)

The SWI is an executable instruction and a non-maskable interrupt: It is executed regardless of the state of the I bit in the CCR.  If the I bit is 0 (interrupts enabled), SWI executes after interrupts which were pending when the SWI was fetched, but before interrupts generated after the SWI was fetched.  The interrupt service routine address is specified by the contents of memory locations $1FFC and $1FFD.

### 4.5.3    EXTERNAL INTERRUPT

If the interrupt mask bit (I bit) of the CCR is set, all maskable interrupts (internal and external) are disabled. Clearing the I bit enables interrupts. The interrupt request is latched immediately following the falling edge of $\overline{\text{IRQ}}$.  It is then synchronized internally and serviced as specified by the contents of $1FFA and $1FFB.

Either a level-sensitive and edge-sensitive trigger or an edge-sensitive-only trigger is controlled by the IRQ bit in the MOR register.

---

NOTE: The internal interrupt latch is cleared in the first part of the interrupt service routine; therefore, one external interrupt pulse could be latched and serviced as soon as the I bit is cleared.

---

### 4.5.4    TIMER INTERRUPT

There are three different timer interrupt flags that cause a timer interrupt whenever they are set and enabled.  The interrupt flags are in the Timer Status Register (TSR), and the enable bits are in the Timer Control Register (TCR).  Any of these interrupts will vector to the same interrupt service routine, located at the address specified by the contents of memory locations $1FF8 and $1FF9.
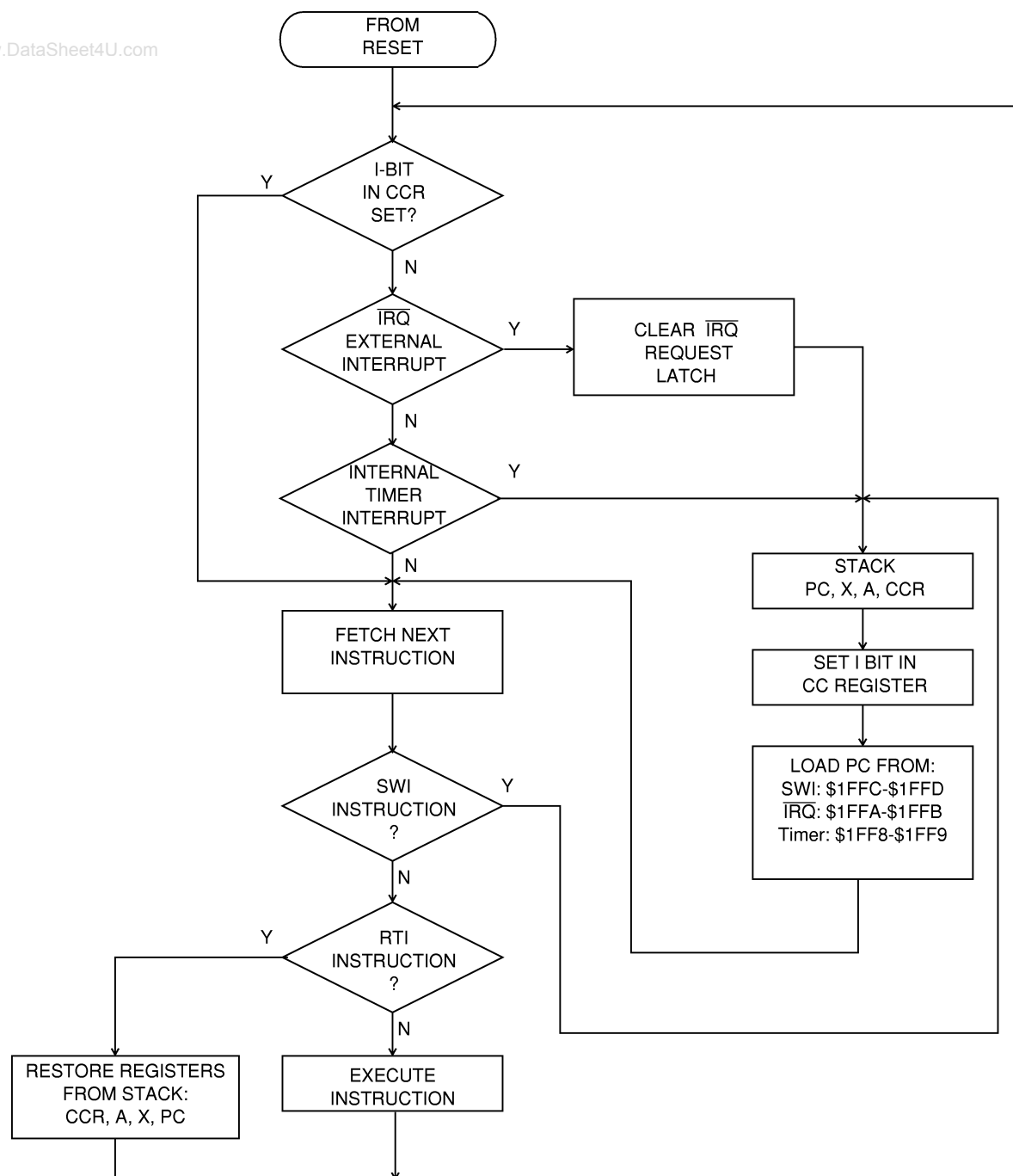
**Figure 4-5: Interrupt Flowchart**

## 4.6 LOW-POWER MODES

### 4.6.1 STOP

The STOP instruction places the MCU in its lowest power consumption mode. In STOP mode, the internal oscillator is turned off, halting all internal processing, including timer and COP Watchdog operation. The RC oscillator is also turned off during STOP mode, and is not available for use by the EEPROM system.

During the STOP mode, the TCR bits are altered to remove any pending timer interrupt request and to disable any further timer interrupts. The timer prescaler is cleared. The I bit in the CCR is cleared to enable external interrupts. All other registers and memory remain unaltered. All input/output lines remain unchanged. The processor can be brought out of the STOP mode only by an external interrupt or reset.

### 4.6.2 STOP RECOVERY

The processor can be brought out of the STOP mode only by an external interrupt or $\overline{\text{RESET}}$. See **Figure 4-6: Stop Recovery Timing Diagram**.



Notes:
1. Represents the internal gating of the OSC1 pin.
2. $\overline{\text{IRQ}}$ pin edge-sensitive controlled by the IRQ bit in the MOR register.
3. $\overline{\text{IRQ}}$ pin level- and edge-sensitive controlled by the IRQ bit in the MOR register.
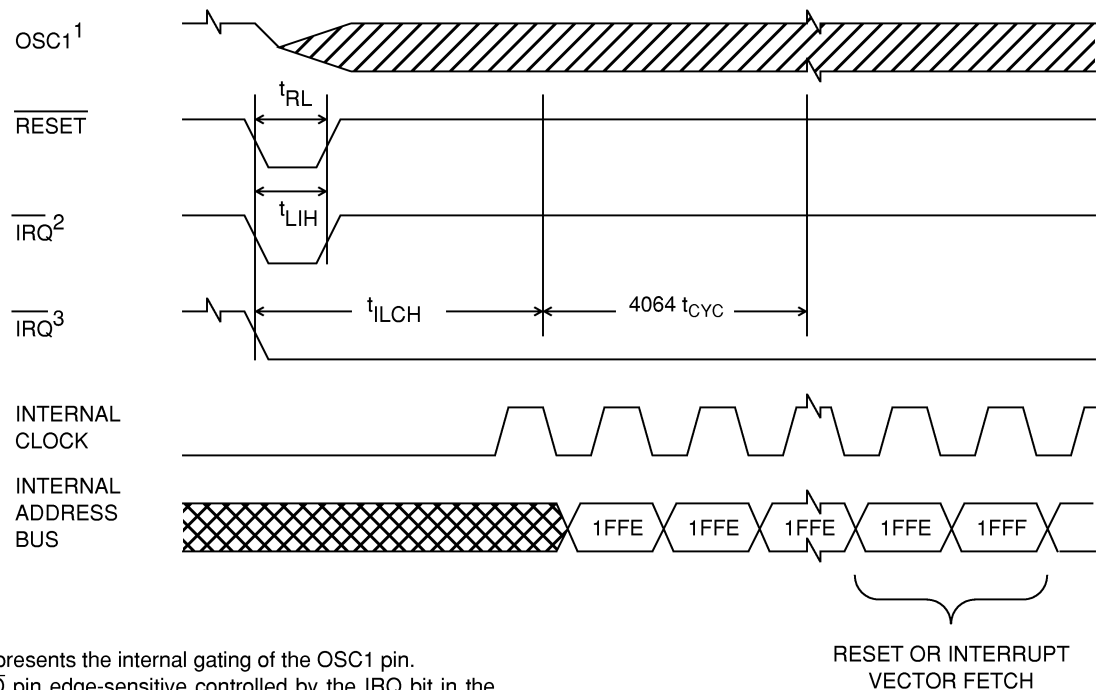
**Figure 4-6: Stop Recovery Timing Diagram**

### 4.6.3 WAIT

The WAIT instruction places the MCU in a low-power consumption mode, but the WAIT mode consumes more power than the STOP mode.  All CPU action is suspended, but the timer and the oscillator remain active.  Any interrupt or reset (including a COP reset) will cause the MCU to exit the WAIT mode.

During the WAIT mode, the I bit in the CCR is cleared to enable interrupts.  All other registers, memory, and input/output lines remain in their previous state.  The timer may be enabled to allow a periodic exit from the WAIT mode.

# SECTION 5                    INPUT/OUTPUT PORTS

In Single-Chip mode, there are 32 lines arranged as four 8-bit I/O ports. These ports are programmable as either inputs or outputs under software control of the data direction registers.

---

NOTE:    To avoid a glitch on the output pins, write data to the I/O Port Data Register before writing a one to the corresponding Data Direction Register.

---

## 5.1    PORT A

Port A is an 8-bit bidirectional port which does not share any of its pins with other subsystems. The port A data register is at $0000 and the data direction register (DDR) is at $0004. Reset does not affect the data registers, but clears the data direction registers, thereby returning the ports to inputs. Writing a one to a DDR bit sets the corresponding port bit to output mode.

## 5.2    PORT B

Port B is an 8-bit bidirectional port. Three of the port B pins (PB5-PB7) are shared with the SIOP subsystem. Refer to **SECTION 8  SIMPLE SERIAL INPUT/OUTPUT PORT** for a detailed description of the SIOP. The port B data register is at $0001 and the data direction register (DDR) is at $0005. Reset does not affect the data registers, but clears the data direction registers, thereby returning the ports to inputs. Writing a one to a DDR bit sets the corresponding port bit to output mode.

## 5.3    PORT C

Port C is an 8-bit bidirectional port which does not share any of its pins with other subsystems. The port C data register is at $0002 and the data direction register (DDR) is at $0006. Reset does not affect the data registers, but clears the data direction registers, thereby returning the ports to inputs. Writing a one to a DDR bit sets the corresponding port bit to output mode. Port C has a high current sink capability. To minimize current spikes, these pins should be switched one at a time.

## 5.4    PORT D

Port D is an 8-bit bidirectional port. PD6 is shared with TCMP. If the PD6 pin is configured as TCMP by setting the COE bit in the Timer Control Register, this pin will become an output controlled by the Timer subsection. Refer to **SECTION 6  TIMER** for more information. The port D data register is at $0003 and the data direction register (DDR) is at $0007. Reset does not affect the data registers, but clears the data direction registers,

thereby returning the ports to inputs. Writing a one to a DDR bit sets the corresponding port bit to output mode.

## 5.4.1 INPUT/OUTPUT PROGRAMMING

Port pins may be programmed as inputs or outputs under software control. The direction of the pins is determined by the state of the corresponding bit in the port data direction register (DDR). Each port has an associated DDR. Any port pin is configured as an output if its corresponding DDR bit is set to a logic one. A pin is configured as an input if its corresponding DDR bit is cleared to a logic zero.

At power-on or reset, all DDRs are cleared, which configures all pins as inputs. The data direction registers are capable of being written to or read by the processor. During the programmed output state, a read of the data register actually reads the value of the output data latch and not the I/O pin. For further information see **Table 5-1: I/O Pin Functions** and **Figure 5-1: I/O Circuitry**.

### Table 5-1:  I/O Pin Functions

| R/$\overline{\text{W}}$ | DDR | I/O Pin Function |
|---|---|---|
| 0 | 0 | The I/O pin is in input mode.  Data is written into the output data latch. |
| 0 | 1 | Data is written into the output data latch and output of the I/O pin. |
| 1 | 0 | The state of the I/O pin is read. |
| 1 | 1 | The I/O pin is in an output mode.  The output data latch is read. |

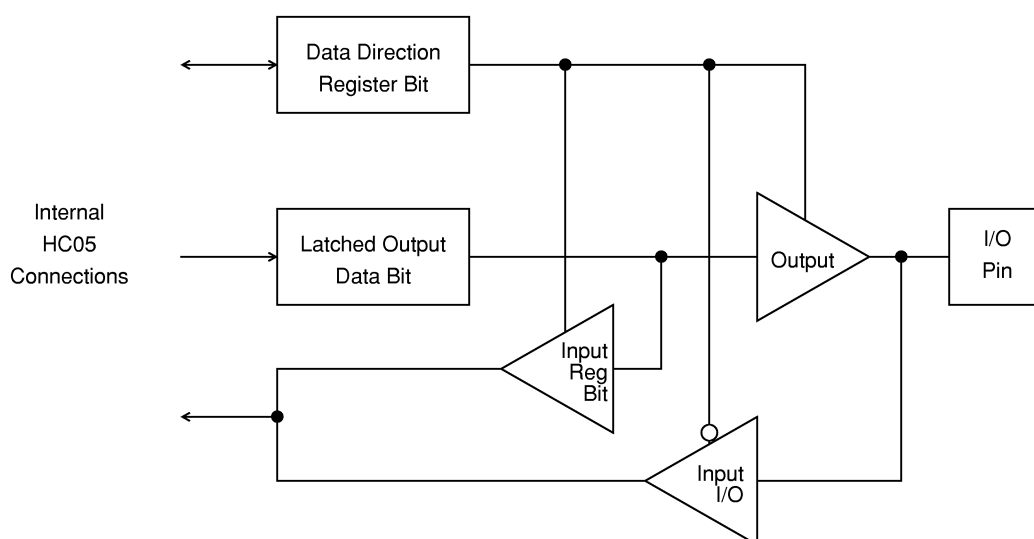R/W is an internal signal.



**Figure 5-1:  I/O Circuitry**

# SECTION 6                                    TIMER

## 6.1    INTRODUCTION

The timer consists of a 16-bit, software-programmable counter driven by a fixed divide-by-4 prescaler.  This timer can be used for many purposes, including input waveform measurements while simultaneously generating an output waveform.  Pulse widths can vary from several microseconds to many seconds.  Refer to **Figure 6-1: Timer Block Diagram** for a timer block diagram.

Because the timer has a 16-bit architecture, each specific functional segment (capability) is represented by two registers.  These registers contain the high and low byte of that functional segment.  Generally, accessing the low byte of a specific timer function allows full control of that function; however, an access of the high byte inhibits that specific timer function until the low byte is also accessed.

Note:  The I bit in the CCR should be set while manipulating both the high and low byte register of a specific timer function to ensure that an interrupt does not occur.
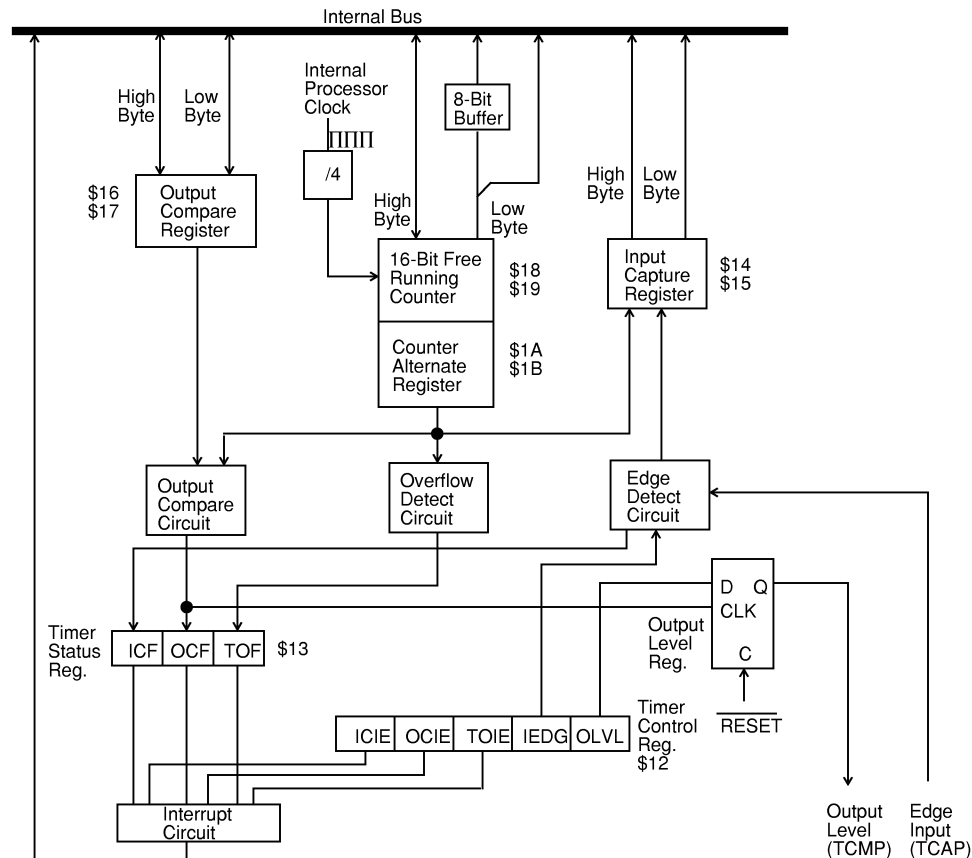


**Figure 6-1:  Timer Block Diagram**

## 6.2    COUNTER

The key element in the programmable timer is a 16-bit, free-running counter or counter register, preceded by a prescaler that divides the internal processor clock by 4. The prescaler gives the timer a resolution of 2.0 microseconds if the internal bus clock is 2.0 MHz. The counter is incremented during the low portion of the internal bus clock. Software can read the counter at any time without affecting its value.

The double-byte, free-running counter can be read from either of two locations, $18-$19 (counter register) or $1A-$1B (counter alternate register). A read from only the least significant byte (LSB) of the free-running counter ($19, $1B) receives the count value at the time of the read. If a read of the free-running counter or counter alternate register first addresses the most significant byte (MSB) ($18, $1A), the LSB ($19, $1B) is transferred to a buffer. This buffer value remains fixed after the first MSB read, even if the user reads the MSB several times. This buffer is accessed when reading the free-running counter or counter alternate register LSB ($19 or $1B) and, thus, completes a read sequence of the total counter value. In reading either the free-running counter or counter alternate register, if the MSB is read, the LSB must also be read to complete the sequence.

The counter alternate register differs from the counter register in one respect: A read of the counter register MSB can clear the timer overflow flag (TOF). Therefore, the counter alternate register can be read at any time without the possibility of missing timer overflow interrupts due to clearing of the TOF.

The free-running counter is configured to $FFFC during reset and is always a read-only register. During a power-on reset, the counter is also preset to $FFFC and begins running after the oscillator start-up delay. Because the free-running counter is 16 bits preceded by a fixed divided-by-4 prescaler, the value in the free-running counter repeats every 262,144 internal bus clock cycles. When the counter rolls over from $FFFF to $0000, the TOF bit is set. When counter rollover occurs, an interrupt can also be enabled by setting its interrupt enable bit (TOIE).

## 6.3    OUTPUT COMPARE REGISTER

The 16-bit output compare register is made up of two 8-bit registers at locations $16 (MSB) and $17 (LSB). The output compare register is used for several purposes, such as indicating when a period of time has elapsed. All bits are readable and writable and are not altered by the timer hardware or reset. If the compare function is not needed, the 2 bytes of the output compare register can be used as storage locations.

The output compare register contents are compared continually with the contents of the free-running counter and, if a match is found, the corresponding output compare flag (OCF) bit is set and the corresponding output level (OLVL) bit is clocked to an output level register. The output compare register values and the output level bit should be changed after each successful comparison to establish a new elapsed time-out. An interrupt can also accompany a successful output compare provided the corresponding interrupt enable bit (OCIE) is set.

After a processor write cycle to the output compare register containing the MSB ($16), the output compare function is inhibited until the LSB ($17) is also written. The user must write

both bytes (locations) if the MSB is written first. A write made only to the LSB ($17) will not inhibit the compare function. The free-running counter is updated every four internal bus clock cycles. The minimum time required to update the output compare register is a function of the program rather than the internal hardware.

The processor can write to either byte of the output compare register without affecting the other byte. The output level (OLVL) bit is clocked to the output level register regardless of whether the output compare flag (OCF) is set or clear.

## 6.4    INPUT CAPTURE REGISTER

Two 8-bit registers, which make up the 16-bit input capture register, are read-only and are used to latch the value of the free-running counter after the corresponding input capture edge detector senses a defined transition. The level transition which triggers the counter transfer is defined by the corresponding input edge bit (IEDG). Reset does not affect the contents of the input capture register except when exiting stop mode.

The result obtained by an input capture will be one more than the value of the free-running counter on the rising edge of the internal bus clock preceding the external transition. This delay is required for internal synchronization. Resolution is one count of the free-running counter, which is four internal bus clock cycles.

The free-running counter contents are transferred to the input capture register on each proper signal transition regardless of whether the input capture flag (ICF) is set or clear. The input capture register always contains the free-running counter value that corresponds to the most recent input capture.

After a read of the input capture register ($14) MSB, the counter transfer is inhibited until the LSB ($15) is also read. This characteristic causes the time used in the input capture software routine and its interaction with the main program to determine the minimum pulse period.

A read of the input capture register LSB ($15) does not inhibit the free-running counter transfer since they occur on opposite edges of the internal bus clock.

## 6.5    TIMER CONTROL REGISTER (TCR) $12

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

| $12 | ICIE | OCIE | TOIE | 0 | 0 | COE | IEDG | OLVL |
|-----|------|------|------|---|---|-----|------|------|
| RESET: | 0 | 0 | 0 | 0 | 0 | 0 | U | 0 |

**Figure 6-2:  Timer Control Register**

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

COE - TCMP Pin Enable

1 = TCMP pin enabled

0 = TCMP pin disabled (pin 35 is PD6)

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

Reset does not affect the IEDG bit (U=unaffected).

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 3 and 4 - Not used

Always read zero

## 6.6     TIMER STATUS REGISTER (TSR) $13

The TSR is a read-only register containing 3 status flag bits.

| $13 | ICF | OCF | TOF | 0 | 0 | 0 | 0 | 0 |
|------|-----|-----|-----|---|---|---|---|---|
| RESET: | U | U | U | 0 | 0 | 0 | 0 | 0 |

**Figure 6-3:  Timer Status Register**

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 0-4 - Not used

Always read zero

Accessing the timer status register satisfies the first condition required to clear status bits. The remaining step is to access the register corresponding to the status bit.

A problem can occur when using the timer overflow function and reading the free-running counter at random times to measure an elapsed time. Without incorporating the proper precautions into software, the timer overflow flag could unintentionally be cleared if:

1. The timer status register is read or written when TOF is set, and

2. The LSB of the free-running counter is read but not for the purpose of servicing the flag.

The counter alternate register at address $1A and $1B contains the same value as the free-running counter (at address $18 and $19); therefore, this alternate register can be read at any time without affecting the timer overflow flag in the timer status register.

## 6.7  TIMER DURING WAIT MODE

The CPU clock halts during the WAIT mode, but the timer remains active. If interrupts are enabled, a timer interrupt will cause the processor to exit the WAIT mode.

## 6.8  TIMER DURING STOP MODE

In the STOP mode, the timer stops counting and holds the last count value if STOP is exited by an interrupt. If RESET is used, the counter is forced to $FFFC. During STOP, if at least one valid input capture edge occurs at the TCAP pin, the input capture detect circuit is armed. This does not set any timer flags or wake up the MCU, but when the MCU does wake up, there is an active input capture flag and data from the first valid edge that occurred during the STOP mode. If RESET is used to exit STOP mode, then no input capture flag or data remains, even if a valid input capture edge occurred.

# SECTION 7 COMPUTER OPERATING PROPERLY

## 7.1 INTRODUCTION

This device includes a "Watchdog" COP feature enabled with the COP bit in the MOR register. The COP is implemented with an 18-bit ripple counter. This provides a time-out period of 64 milliseconds at a bus rate of 2 Mhz. If the COP should time-out, a system reset will occur and the device will be re-initialized in the same fashion as a POR or external Reset.
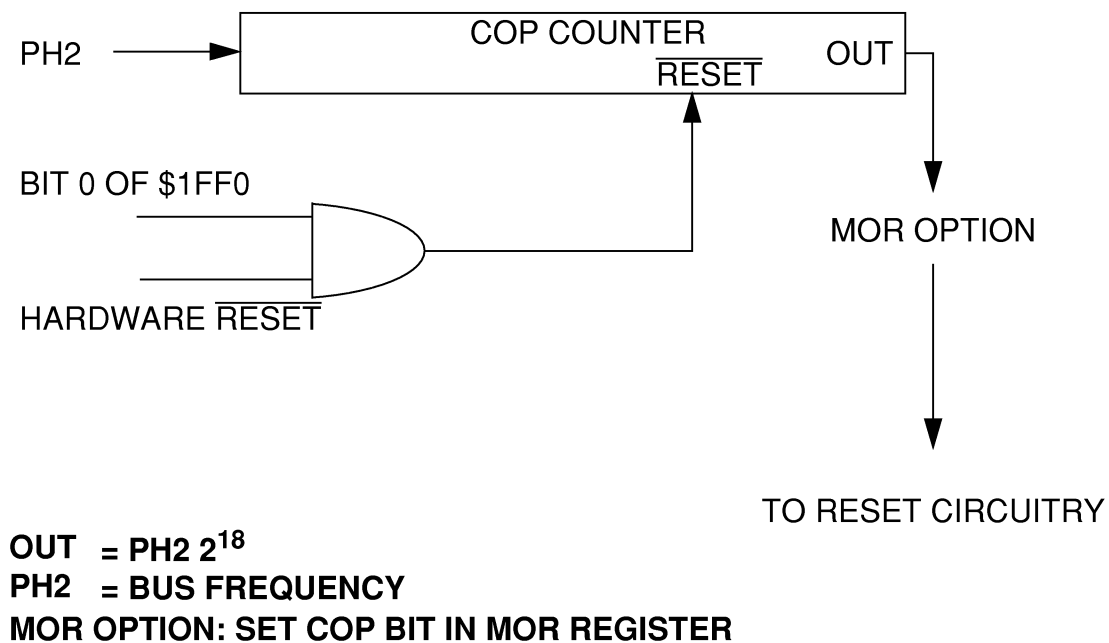


$OUT = PH2\ 2^{18}$
$PH2 = BUS\ FREQUENCY$
**MOR OPTION: SET COP BIT IN MOR REGISTER**

**Figure 7-1: COP Block Diagram**

## 7.2 RESETTING THE COP

Preventing a COP reset is done by writing a "0" to the COPF bit. This action will reset the counter and begin the time-out period again. The COPF bit is bit 0 of address $1FF0. A read of address $1FF0 will result in the user-defined EPROM data at that location.

## 7.3 COP DURING WAIT MODE

The COP will continue to operate normally during WAIT mode. The software should pull the device out of WAIT mode periodically and reset the COP by writing to the COPF bit to prevent a COP reset.

## 7.4     COP DURING STOP MODE

STOP mode disables the oscillator circuit and thereby turns the clock off for the entire device.  The COP counter will be reset when STOP mode is entered.  If a reset is used to exit STOP mode, the COP counter will be reset after the 4064 cycles of delay after STOP mode.  If an IRQ is used to exit STOP mode, the COP counter will not be reset after the 4064 cycle delay and will have that many cycles already counted when control is returned to the program.

## 7.5     COP AFTER POR

After POR the COP counter will not be reset and will have 4064 cycles already counted when control is returned to the program. If the RC time constant on the RESET pin is longer than the POR start-up time, the counter will be reset to zero cycles counted.

## 7.6     COP DURING SELF-CHECK MODE

The COP is disabled during Self-Check Mode.

# SECTION 8     SIMPLE SERIAL INPUT/OUTPUT PORT

This device includes a simple synchronous Serial I/O Port (SIOP).  The SIOP is a three-wire Master/Slave system including Serial Clock (SCK), Serial Data Input (SDI), and Serial Data Output (SDO).  The SIOP bit in the MOR register determines whether the SIOP is MSB or LSB first. See **Figure 2-4: Mask Option Register**
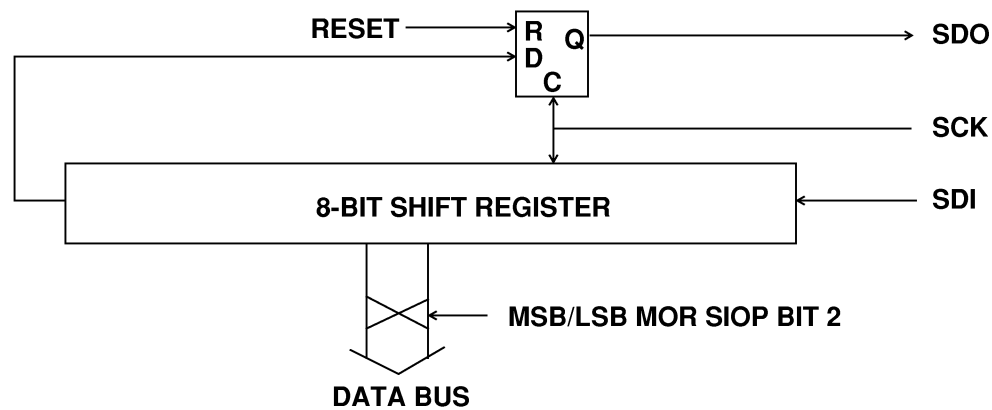
**Figure 8-1:  SIOP Block Diagram**

## 8.1     SIGNAL FORMAT

### 8.1.1     SCK

The state of SCK between transmissions must be logic one for CPOL set and logic zero for CPOL clear.  The first transition of SCK signals the beginning of a transmission.  At this time, the first bit of received data is accepted at the SDI pin and the first bit of transmitted data is presented at the SDO pin.  Data is captured at the SDI pin on the rising edge of SCK.  Subsequent falling edges shift the data and accept or present the next bit.  The transmission is ended upon the eighth rising edge of SCK.  The maximum frequency of SCK in slave mode is equal to E (bus clock) divided by 4.  That is, for a 4 MHz oscillator, input E becomes 2 MHz and the maximum SCK frequency is 500 KHz.  There is no minimum SCK frequency.

In Master Mode, the format is identical except that the SCK pin is an output and the shift clock now originates internally.  The Master Mode transmission frequency is fixed at E/4.

### 8.1.2     SDO

A bit in the MOR Register will be included to allow data to be transmitted in either MSB first format or LSB first format.  In either case, the state of the SDO pin will always reflect the value of the first bit received on the previous transmission if there was one.  Prior to enabling the SIOP, PB5 can be initialized to determine the beginning state if necessary. While the SIOP is enabled, PB5 can not be used as a standard output since that pin is coupled to the last stage of the serial shift register.  If CPOL is set, the first falling edge of SCK will shift the first data bit out to the output pin.  If CPOL is clear, the first data bit will be on the SDO pin waiting for the transmission.

Section 8:  Simple Serial Input/Output Port                              MOTOROLA

### 8.1.3    SDI

The SDI pin becomes an input as soon as the SIOP is enabled.  New data may be presented to the SDI pin on the falling edge of SCK.  Valid data must be present at least 100 ns before the rising edge of the clock and remain valid for 100 ns after the edge
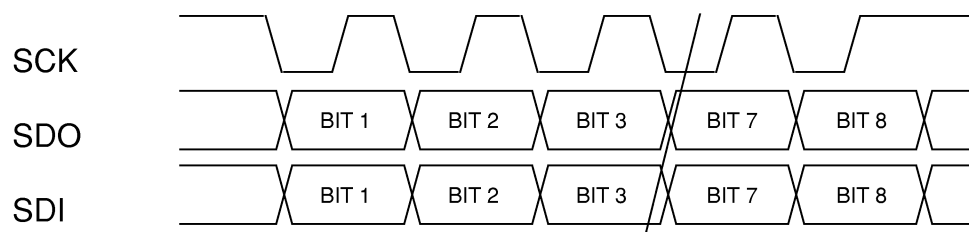


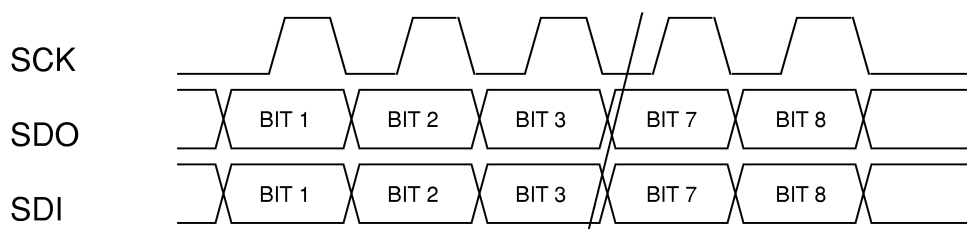**Figure 8-2:  Serial I/O Port Timing (CPOL = 1)**



**Figure 8-3:  Serial I/O Port Timing (CPOL = 0)**

## 8.2    SIOP REGISTERS

### 8.2.1    SIOP CONTROL REGISTER (SCR)

This register is located at address $000A and contains 3 bits.

| $0A | 0 | SPE | 0 | MSTR | CPOL | 0 | 0 | 0 |
|-----|---|-----|---|------|------|---|---|---|
| RESET: | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

**Figure 8-4:  SIOP Control Register**

### 8.2.2    SPE - SERIAL PERIPHERAL ENABLE

When set, this bit enables the Serial I/O Port and initializes the Port B DDR such that PB5 (SDO) is output, PB6 (SDI) is input and PB7 (SCK) is input (Slave Mode only).  The Port B DDR can be subsequently altered as the application requires and the Port B data register (except for PB5) can be manipulated as usual; however, these actions could affect the transmitted or received data.  When SPE is cleared, Port B reverts to standard parallel I/O without affecting the Port B data register or DDR.  SPE is readable and writable any time but clearing SPE while a transmission is in progress will abort the transmission, reset the bit counter, and return Port B to its normal I/O function.  Reset clears this bit.

### 8.2.3    MSTR - MASTER MODE

When set, this bit configures the SIOP for Master Mode.  This means that the transmission is initiated by a write to the data register and the SCK pin becomes an output providing a synchronous data clock at a fixed rate of E (bus clock) divided by 4.  While the device is in Master Mode, the SDO and SDI pins do not change function.  These pins behave exactly as they would in Slave Mode.  Reset clears this bit and configures the SIOP for Slave operation.  MSTR may be set at any time regardless of the state of SPE.  Clearing  MSTR will abort any transmission in progress.

### 8.2.4    CPOL - CLOCK POLARITY

The Clock Polarity bit controls the SCK polarity between transmissions.  When this bit is cleared, SCK will be low between transmissions.  When this bit is set, SCK will be high between transmissions.  In both cases, the data is latched on the rising edge of SCK for serial input and is valid on the rising edge of SCK for serial output.  Reset sets this bit.

When using the Clock Polarity low mode (CPOL=0), the proper mode should be entered before enabling the serial system.  The CPOL bit should be cleared first.  Then the SPE bit should be set during a second write to the SCR.  The following example shows a proper sequence.

* For Master Mode CPOL=0

LDA#$00

STASCRclear CPOL

LDA#$50

STASCRset Mstr, set SPE


* For Slave Mode CPOL=0

LDA#$00

STASCRclear CPOL

LDA#$40

STASCRset SPE


### 8.2.5    SIOP STATUS REGISTER (SSR)

This register is located at address $000B and contains only 2 bits.

| $0B | SPIF | DCOL | 0 | 0 | 0 | 0 | 0 | 0 |
|-----|------|------|---|---|---|---|---|---|
| RESET: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 8-5:  SIOP Status Register**

Section 8:  Simple Serial Input/Output Port

### 8.2.6    SPIF - SERIAL PERIPHERAL INTERFACE FLAG

To indicate that a data transfer has taken place, this bit is set upon occurrence of the last rising clock edge if CPOL is set and the last falling clock edge if CPOL is clear. It has no effect on any further transmissions and can be ignored without problem.  SPIF is cleared by reading the SSR with SPIF set followed by a read or write of the serial data register.  If SPIF is cleared before the last edge of the next byte, it will be set again.  Reset clears this bit.

### 8.2.7    DCOL - DATA COLLISION

This is a read-only status bit which indicates that an invalid access to the data register has been made.  This can occur any time after the first falling edge of SCK if CPOL is set and after the first rising edge of SCK if CPOL is clear and before SPIF is set.  A read or write of the data register during this time will result in invalid data being transmitted or received.

DCOL is cleared by reading the status register with SPIF set followed by a read or write of the data register.  If the last part of the clearing sequence is done after another transmission has been started, DCOL will be set again.  Reset also clears this bit.

### 8.2.8    SIOP DATA REGISTER (SDR)

This register is located at address $000C and is both the transmit and receive data register. This system is not double buffered and any write to this register will destroy the previous contents.  The SDR can be read at any time but if a transmission is in progress the results may be ambiguous.  Writes to the SDR while a transmission is in progress can cause invalid data to be transmitted and/or received.  This register can be read and written only when the SIOP is enabled (SPE=1).
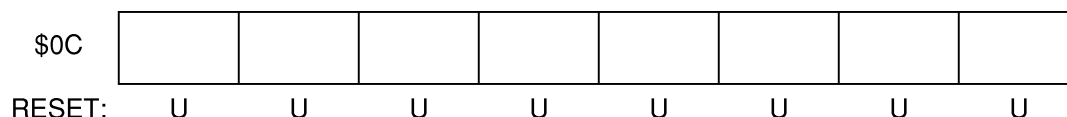
| $0C | | | | | | | | |
|------|---|---|---|---|---|---|---|---|
| RESET: | U | U | U | U | U | U | U | U |

**Figure 8-6:  SIOP Data Register**

# SECTION 9    ELECTRICAL SPECIFICATIONS

## 9.1    MAXIMUM RATINGS

(Voltages referenced to $V_{SS}$)

| Rating | Symbol | Value | Unit |
|---|---|---|---|
| Supply Voltage | $V_{DD}$ | -0.3 to +7.0 | V |
| Input Voltage | $V_{IN}$ | $V_{SS}$ - 0.3 to $V_{DD}$ + 0.3 | V |
| EPROM Programming Voltage/PE | $V_{PP}$ | $V_{SS}$ - 0.3 to 16 | V |
| Self-Check Mode ($\overline{IRQ}$ Pin Only) | $V_{IN}$ | $V_{SS}$ - 0.3 to $2 \times V_{DD}$ + 0.3 | V |
| Current Drain Per Pin Excluding $V_{DD}$ and $V_{SS}$ | I | 25 | mA |
| Operating Temperature Range<br>    MC68HC705C5P (Standard)<br>    MC68HC705C5CP (Extended) | $T_A$ | $T_L$ to $T_H$<br>0 to +70<br>-40 to +85 | °C |
| Storage Temperature Range | $T_{STG}$ | -65 to +150 | °C |

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit.  For proper operation, it is recommended that $V_{IN}$ and $V_{OUT}$ be constrained to the range $V_{SS} \leq (V_{IN}$ or $V_{OUT}) \leq V_{DD}$.  Reliability of operation is enhanced if unused inputs are connected to an appropriate logic voltage level (for example, either $V_{SS}$ or $V_{DD}$).

## 9.2    THERMAL CHARACTERISTICS

| Characteristic | Symbol | Value | Unit |
|---|---|---|---|
| Thermal Resistance<br>    Plastic | $\theta_{JA}$ | 60 | °C/W |
| Plastic Leaded Chip Carrier | $\theta_{JA}$ | 70 | °C/W |
| Cerdip | $\theta_{JA}$ | 60 | °C/W |

## 9.3 DC ELECTRICAL CHARACTERISTICS

($V_{DD}$ = 5.0 Vdc $\pm$10%, $V_{SS}$ = 0 Vdc, $T_A$ = -40$^{\circ}$C to +85$^{\circ}$C, unless otherwise noted)

| Characteristic | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| Output voltage <br> $I_{Load}$ = 10.0 $\mu$A <br> $I_{Load}$ = -10.0 $\mu$A | $V_{OL}$ <br> $V_{OH}$ | — <br> $V_{DD}$-0.1 | — <br> — | 0.1 <br> — | V |
| Output High Voltage <br> ($I_{Load}$ = -0.8 mA) PA0-7, PB0-7,PC0-7,PD0-7 | $V_{OH}$ | $V_{DD}$-0.8 | — | — | V |
| Output Low Voltage <br> ($I_{Load}$ = 1.6 mA) PA0-7, PB0-7,PD0-7 <br> ($I_{Load}$ = 10 mA) PC0-7 | $V_{OL}$ | — | — | 0.4 | V |
| Input High Voltage <br> PA0-7, PB0-7,PC0-7, PD0-7, $\overline{IRQ}$,$\overline{RESET}$,OSC1, TCAP | $V_{IH}$ | $0.7 \times V_{DD}$ | — | $V_{DD}$ | V |
| Input Low Voltage <br> PA0-7, PB0-7,PC0-7, PD0-7, $\overline{IRQ}$,$\overline{RESET}$,OSC1, TCAP | $V_{IL}$ | $V_{SS}$ | — | $0.2 \times V_{DD}$ | V |
| Low Voltage Programming Inhibit | $V_{LVPI}$ | 3.5 | — | — | V |
| Low Voltage Programming Recover | $V_{LVPR}$ | — | — | 4.5 | V |
| Low Voltage Programming Inhibit/Recover Hysteresis | $H_{LVPI}$ | 0.1 | — | 1.0 | V |
| Supply Current (see Notes) <br> Run <br> Wait <br> Stop with LVPI Enabled <br>   25 $^{\circ}$C <br>   -40 $^{\circ}$C to +85 $^{\circ}$C <br> Stop with LVPI Disabled <br>   25 $^{\circ}$C <br>   -40 $^{\circ}$C to +85 $^{\circ}$C | $I_{DD}$ <br> $I_{DD}$ <br><br> $I_{DD}$ <br> $I_{DD}$ <br><br> $I_{DD}$ <br> $I_{DD}$ | — <br> — <br><br> — <br> — <br><br> — <br> — | 3.0 <br> 3.0 <br><br> 250 <br> 250 <br><br> 5 <br> 15 | 7.0 <br> 4.0 <br><br> 400 <br> 400 <br><br> 90 <br> 210 | mA <br> mA <br><br> $\mu$A <br> $\mu$A <br><br> $\mu$A <br> $\mu$A |
| I/O Ports Hi-Z Leakage Current <br> PA0-7, PB0-7,PC0-7,PD0-7 | $I_{OZ}$ | — | — | $\pm$10 | $\mu$A |
| Input Current <br> $\overline{RESET}$, $\overline{IRQ}$, OSC1,TCAP <br> PE | $I_{IN}$ <br> $I_{IN}$ | — <br> — | — <br> — | $\pm$1 <br> TBD | $\mu$A <br> $\mu$A |
| Capacitance <br> Ports (as Input or Output) <br> $\overline{RESET}$, $\overline{IRQ}$, TCAP | $C_{OUT}$ <br> $C_{IN}$ | — <br> — | — <br> — | 12 <br> 8 | pF <br> pF |
| EPROM Programming Voltage | $V_{PP}$ | 14 | 15 | 16 | V |
| $V_{PP}$ Supply Current ($V_{PP}$ = Programming Voltage) | $I_{PP}$ | 2.25 | 5 | 10 | mA |

NOTES:
1. All values shown reflect average measurements.
2. Typical values at midpoint of voltage range, 25$^{\circ}$C only
3. Wait $I_{DD}$: Only timer system active.
4. Run (Operating) $I_{DD}$, Wait $I_{DD}$: Measured using external square wave clock source ($f_{osc}$ = 4.2 MHz), all inputs 0.2V from rail; no dc loads, less than 50 pF on all outputs, $C_L$ = 20 pF on OSC2.
5. Wait, Stop $I_{DD}$: All ports configured as inputs, $V_{IL}$ = 0.2 V, $V_{IH}$ = $V_{DD}$-0.2 V.
6. Stop $I_{DD}$ measured with OSC1 = $V_{SS}$.
7. Wait $I_{DD}$ is affected linearly by the OSC2 capacitance.

## 9.4    CONTROL TIMING

($V_{DD}$ = 5.0 Vdc±10%, $V_{SS}$ = 0 Vdc, $T_A$ = -40°C to +85°C, unless otherwise noted)

| Characteristic | Symbol | Min | Max | Unit |
|---|---|---|---|---|
| Frequency of Operation<br>  Crystal Option<br>  External Clock Option | $f_{osc}$<br>$f_{osc}$ | —<br>dc | 4.2<br>4.2 | MHz<br>MHz |
| Internal Operating Frequency<br>  Crystal ($f_{osc} \div 2$)<br>  External Clock ($f_{osc} \div 2$) | $f_{OP}$<br>$f_{OP}$ | —<br>dc | 2.1<br>2.1 | MHz<br>MHz |
| Cycle Time | $t_{CYC}$ | 480 | — | ns |
| Crystal Oscillator Start-up Time | $t_{OXOV}$ | — | 100 | ms |
| Stop Recovery Start-up Time (Crystal Oscillator) | $t_{ILCH}$ | — | 100 | ms |
| $\overline{RESET}$ Pulse Width | $t_{RL}$ | 1.5 | — | $t_{CYC}$ |
| Interrupt Pulse Width Low (Edge-Triggered) | $t_{ILIH}$ | 125 | — | ns |
| Interrupt Pulse Period | $t_{ILIL}$ | * | — | $t_{CYC}$ |
| OSC1 Pulse Width | $t_{OH},t_{OL}$ | 90 | — | ns |
| EEPROM Byte Programming Time | $t_{EPGM}$ | — | 15.0 | ms |
| EEPROM Byte Erase Time | $t_{EBYT}$ | — | 15.0 | ms |
| EEPROM Block  Erase Time | $t_{EBLOCK}$ | — | 30.0 | ms |
| EEPROM Bulk Erase Time | $t_{EBULK}$ | — | 100.0 | ms |
| EEPROM Programming Voltage Fall Time<br>  Normal Operation<br>  After LVPI set | $t_{FPV}$<br>$t_{FPVL}$ | —<br>— | 10.0<br>10.0 | μs<br>μs |
| EPROM Byte Programming Time | $t_{PROG}$ | 2 | -- | ms |
| $V_{DD}$ Slew Rate<br>  Rising<br>  Falling | $t_{VDDR}$<br>$t_{VDDF}$ | —<br>— | 0.05<br>0.1 | V/μs<br>V/μs |
| RC Oscillator Stabilization Time (EEPROM) | $t_{RCON}$ | — | 5.0 | μs |

\* The minimum period $T_{ILIL}$ should not be less than the number of cycle times it takes to execute the interrupt service routine plus 21 $t_{CYC}$.
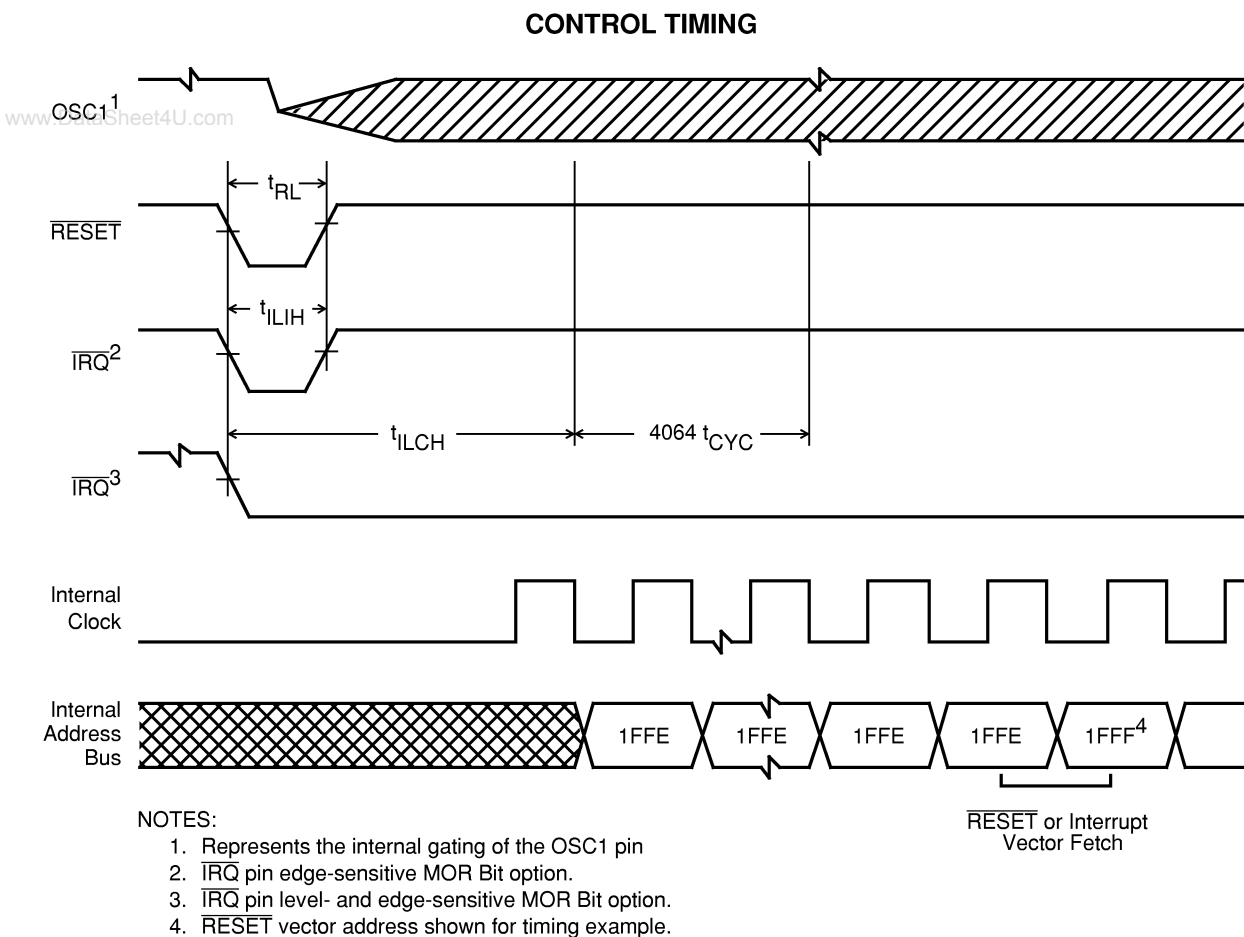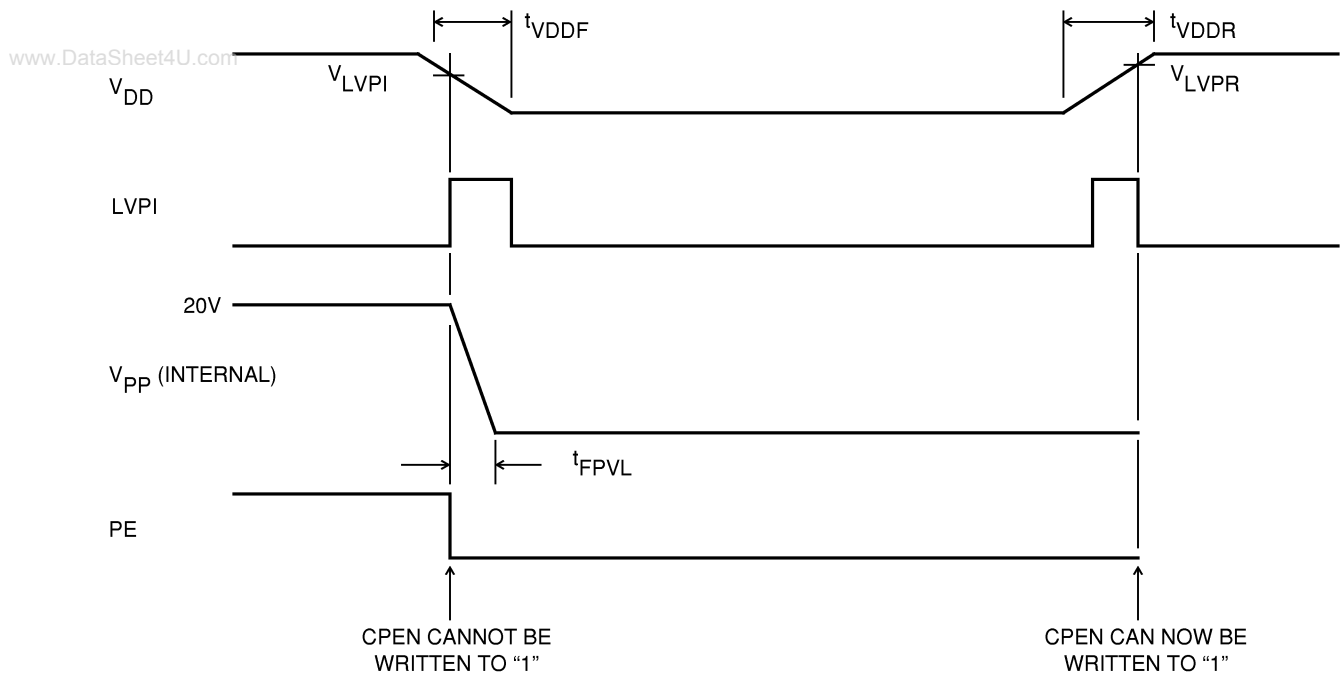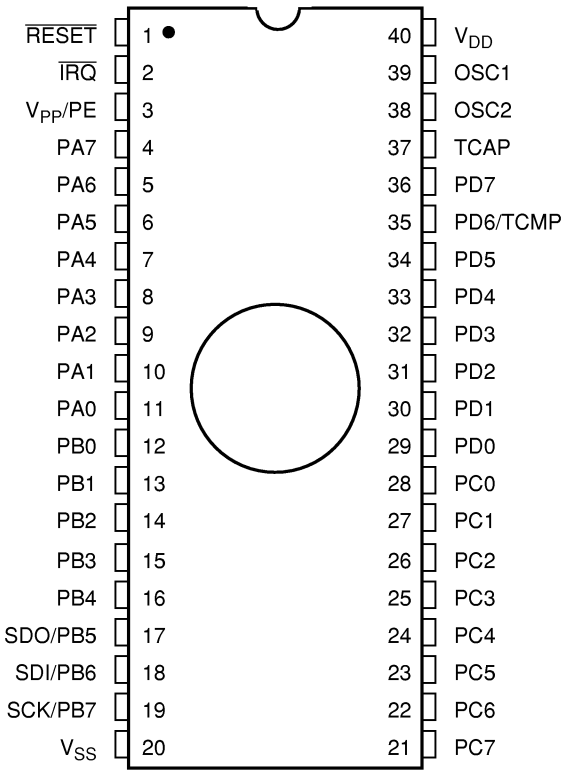
**CONTROL TIMING**



NOTES:
1. Represents the internal gating of the OSC1 pin
2. $\overline{IRQ}$ pin edge-sensitive MOR Bit option.
3. $\overline{IRQ}$ pin level- and edge-sensitive MOR Bit option.
4. $\overline{RESET}$ vector address shown for timing example.

**Figure 9-1: Stop Recovery Timing Diagram**

**Figure 9-2: LVPI Timing Diagram**
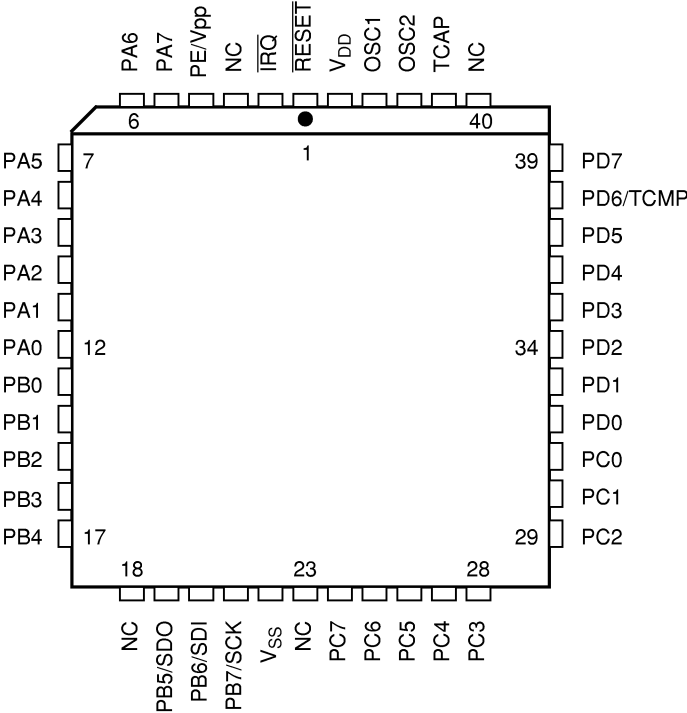
Section 9: Electrical Specifications

# SECTION 10     MECHANICAL SPECIFICATIONS

## 10.1    40-PIN DUAL INLINE PACKAGE

| | | |
|---|---|---|
| $\overline{\text{RESET}}$ | 1 | 40 $V_{DD}$ |
| $\overline{\text{IRQ}}$ | 2 | 39 OSC1 |
| $V_{PP}$/PE | 3 | 38 OSC2 |
| PA7 | 4 | 37 TCAP |
| PA6 | 5 | 36 PD7 |
| PA5 | 6 | 35 PD6/TCMP |
| PA4 | 7 | 34 PD5 |
| PA3 | 8 | 33 PD4 |
| PA2 | 9 | 32 PD3 |
| PA1 | 10 | 31 PD2 |
| PA0 | 11 | 30 PD1 |
| PB0 | 12 | 29 PD0 |
| PB1 | 13 | 28 PC0 |
| PB2 | 14 | 27 PC1 |
| PB3 | 15 | 26 PC2 |
| PB4 | 16 | 25 PC3 |
| SDO/PB5 | 17 | 24 PC4 |
| SDI/PB6 | 18 | 23 PC5 |
| SCK/PB7 | 19 | 22 PC6 |
| $V_{SS}$ | 20 | 21 PC7 |

## 10.2    44-PIN PLCC PACKAGE

Section 10:  Mechanical Specifications

**How to reach us:**

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

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

    **INTERNET:** http://www.mot.com/SPS/

    **JAPAN:** Nippon Motorola Ltd.; Tatsumi-SPD-JLDC, 6F Seibu-Butsuryu-Center, 3-14-2 Tatsumi Koto-Ku, Tokyo 135, Japan.
        81-3-3521-8315

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

Mfax is a trademark of Motorola, Inc.

**MOTOROLA**

HC705C5GRS/D