# HD63P05Y1, HD63PA05Y1, HD63PB05Y1

# CMOS MCU (Microcomputer Unit)

## —PRELIMINARY—

The HD63P05Y1 is a CMOS 8-bit single-chip microcomputer unit which has a 4k-byte or 8k-byte EPROM on the package. It is compatible with the HD6305Y1 except for ROM which is not included in the HD63P05Y1. It can be used not only for debugging and evaluating the internal program of HD6305X1 or HD6305Y1, but also for small-sized production preceding mask ROM.

■ **FEATURES**
● Pin compatible with HD6305X1 and HD6305Y1
● 256-byte of RAM
● A total of 31 terminals, including 24 I/O's, 7 inputs
● Two timers
  — 8-bit timer with a 7-bit prescaler (programmable prescaler; event counter)
  — 15-bit timer (commonly used with the SCI clock divider)
● On-chip serial interface circuit (synchronized with clock)
● Six interrupts (two external, two timer, one serial and one software)
● Low power dissipation modes — Wait, Stop and Standby Mode
● Minimum instruction cycle time
  HD63P05Y1 . . . . . . 1 $\mu$s (f = 1 MHz)
  HD63PA05Y1 . . . . 0.67 $\mu$s (f = 1.5 MHz)
  HD63PB05Y1 . . . . . 0.5 $\mu$s (f = 2 MHz)
● Similar to HD6800 instruction set
● Bit manipulation
● Bit test and branch
● Versatile interrupt handling
● Full set of conditional branches
● New instructions — STOP, WAIT, DAA
● Applicable to 4k or 8k bytes of EPROM
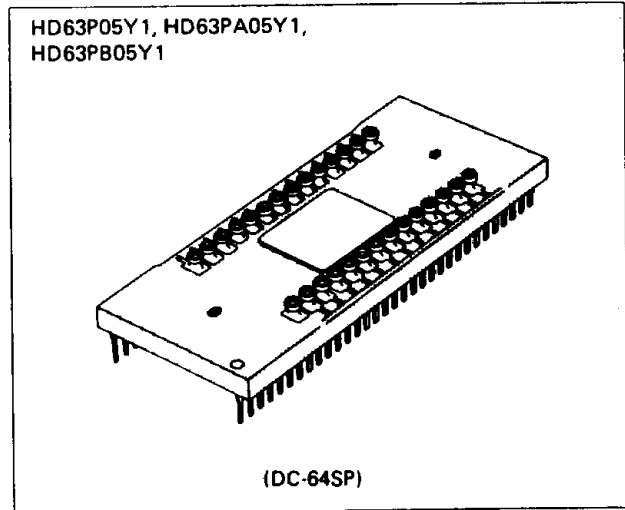  4k bytes: HN482732A
  8k bytes: HN482764, HN27C64

■ **TYPE OF PRODUCTS**

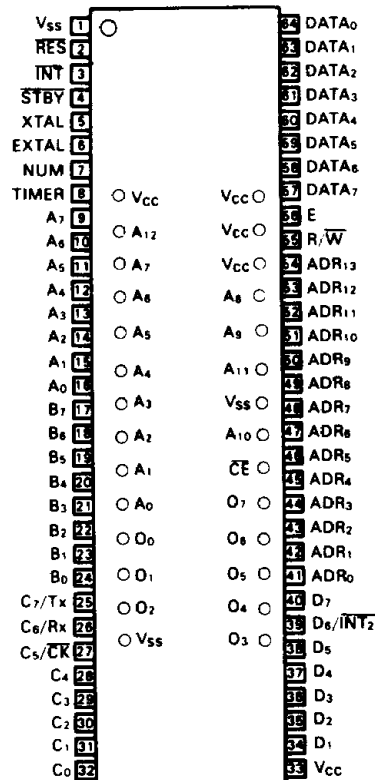| Type No. | Bus Timing | Applied EPROM |
|---|---|---|
| HD63P05Y1 | 1 MHz | HN482732A-30, HN482764-3, HN27C64-30 |
| HD63PA05Y1 | 1.5 MHz | HN482732A-30, HN482764-3, HN27C64-30 |
| HD63PB05Y1 | 2 MHz | HN482732A-25, HN482764, HN27C64-25 |

(Note) EPROM is not attached to the MCU.

■ **PROGRAM DEVELOPMENT SUPPORT TOOLS**
● Cross assembler software for use with IBM PCs and compatibles
● In circuit emulator for use with IBM PCs and compatibles

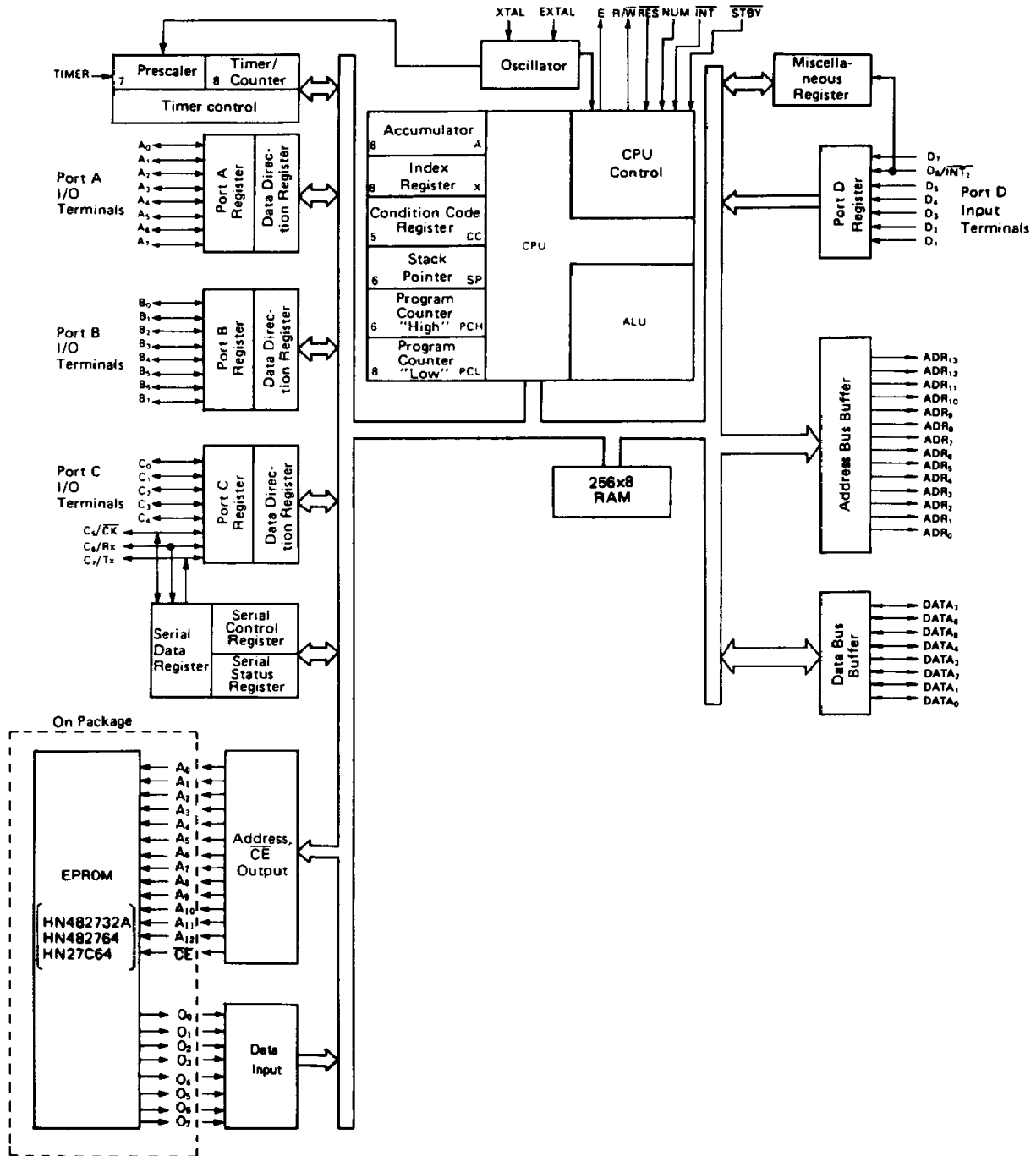HD63P05Y1, HD63PA05Y1, HD63PB05Y1



(DC-64SP)

■ **PIN ARRANGEMENT**



| Pin | Signal | | Pin | Signal |
|---|---|---|---|---|
| 1 | $V_{SS}$ | | 64 | $DATA_0$ |
| 2 | $\overline{RES}$ | | 63 | $DATA_1$ |
| 3 | $\overline{INT}$ | | 62 | $DATA_2$ |
| 4 | $\overline{STBY}$ | | 61 | $DATA_3$ |
| 5 | XTAL | | 60 | $DATA_4$ |
| 6 | EXTAL | | 59 | $DATA_5$ |
| 7 | NUM | | 58 | $DATA_6$ |
| 8 | TIMER | | 57 | $DATA_7$ |
| 9 | $A_7$ | | 56 | E |
| 10 | $A_6$ | | 55 | $R/\overline{W}$ |
| 11 | $A_5$ | | 54 | $ADR_{13}$ |
| 12 | $A_4$ | | 53 | $ADR_{12}$ |
| 13 | $A_3$ | | 52 | $ADR_{11}$ |
| 14 | $A_2$ | | 51 | $ADR_{10}$ |
| 15 | $A_1$ | | 50 | $ADR_9$ |
| 16 | $A_0$ | | 49 | $ADR_8$ |
| 17 | $B_7$ | | 48 | $ADR_7$ |
| 18 | $B_6$ | | 47 | $ADR_6$ |
| 19 | $B_5$ | | 46 | $ADR_5$ |
| 20 | $B_4$ | | 45 | $ADR_4$ |
| 21 | $B_3$ | | 44 | $ADR_3$ |
| 22 | $B_2$ | | 43 | $ADR_2$ |
| 23 | $B_1$ | | 42 | $ADR_1$ |
| 24 | $B_0$ | | 41 | $ADR_0$ |
| 25 | $C_7/Tx$ | | 40 | $D_7$ |
| 26 | $C_6/Rx$ | | 39 | $D_6/\overline{INT_2}$ |
| 27 | $C_5/\overline{CK}$ | | 38 | $D_5$ |
| 28 | $C_4$ | | 37 | $D_4$ |
| 29 | $C_3$ | | 36 | $D_3$ |
| 30 | $C_2$ | | 35 | $D_2$ |
| 31 | $C_1$ | | 34 | $D_1$ |
| 32 | $C_0$ | | 33 | $V_{CC}$ |

Inner (EPROM socket) pins: $V_{CC}$, $V_{CC}$, $V_{CC}$, $A_8$, $A_9$, $A_{11}$, $V_{SS}$, $A_{10}$, $\overline{CE}$, $O_7$, $O_8$, $O_5$, $O_4$, $O_3$, $V_{CC}$, $A_{12}$, $A_7$, $A_6$, $A_5$, $A_4$, $A_3$, $A_2$, $A_1$, $A_0$, $O_0$, $O_1$, $O_2$, $V_{SS}$

(Top View)

● BLOCK DIAGRAM

## ■ ABSOLUTE MAXIMUM RATINGS

| Item | Symbol | Value | Unit |
|---|---|---|---|
| Supply voltage | $V_{CC}$ | $-0.3 \sim +7.0$ | V |
| Input voltage | $V_{in}$ | $-0.3 \sim V_{CC} + 0.3$ | V |
| Operating temperature | $T_{opr}$ | $0 \sim +70$ | °C |
| Storage temperature | $T_{stg}$ | $-55 \sim +150$ | °C |

[NOTE]  These products have a protection circuit in their input terminals against high electrostatic voltage or high electric fields. Notwithstanding, be careful not to apply any voltage higher than the absolute maximum rating to these high input impedance circuits. To assure normal operation, we recommended $V_{in}$, $V_{out}$; $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{CC}$

## ■ ELECTRICAL CHARACTERISTICS

● DC Characteristics ($V_{CC}$ = 5.0V ± 10%, $V_{SS}$ = GND, $T_a$ = 0 ~ +70°C, unless otherwise specified.)

| Item | | Symbol | Test Condition | min. | typ. | max. | Unit |
|---|---|---|---|---|---|---|---|
| Input voltage "High" | $\overline{RES}$, $\overline{STBY}$ | $V_{IH}$ | | $V_{CC}-0.5$ | — | $V_{CC} + 0.3$ | V |
| | EXTAL | | | $V_{CC} \times 0.7$ | — | $V_{CC} + 0.3$ | V |
| | Others | | | 2.0 | — | $V_{CC} + 0.3$ | V |
| Input voltage "Low" | All Input | $V_{IL}$ | | $-0.3$ | — | 0.8 | V |
| Output "High" voltage | All Input | $V_{OH}$ | $I_{OH} = -200 \mu A$ | 2.4 | — | — | V |
| | | | $I_{OH} = -10 \mu A$ | $V_{CC}-0.7$ | — | — | V |
| Output "Low" voltage | All output | $V_{OL}$ | $I_{OL} = 1.6$ mA | — | — | 0.55 | V |
| Current*** dissipation | Operating | $I_{CC}$ | $f = 1$MHz* | — | 5 | 10 | mA |
| | Wait | | | — | 2 | 5 | mA |
| | Stop | | | — | 2 | 10 | $\mu A$ |
| | Standby | | | — | 2 | 10 | $\mu A$ |
| Input leakage current | TIMER, $\overline{INT}$, $D_1 \sim D_7$, $\overline{STBY}$ | $|I_{IL}|$ | $V_{IN} = 0.5 \sim V_{CC}-0.5V$ | — | — | 1 | $\mu A$ |
| Three-state current | $A_0 \sim A_7$, $B_0 \sim B_7$, $C_0 \sim C_7$, $ADR_0 \sim ADR_{13}$,** $DATA_0 \sim DATA_7$, E**, R/$\overline{W}$** | $|I_{TSI}|$ | | — | — | 1 | $\mu A$ |
| Input Capacity | All terminals | $C_{in}$ | $f = 1$MHz, $V_{IN} = 0V$ | — | — | 15 | pF |

* The value at f = xMHz can be calculated by the following equation:    $I_{CC}$ (f= xMHz) = $I_{CC}$ (f= 1MHz) multiplied by x
** At standby mode
*** All output and $\overline{RES}$ terminals are open ($V_{IH}$ min = $V_{CC}-1.0V$, $V_{IL}$ max = 0.8V), and $I_{CC}$ of EPROM is not included.

# HD63P05Y1, HD63PA05Y1, HD63PB05Y1

● **AC Characteristics** ($V_{CC}$ = 5.0V ± 10%, $V_{SS}$ = GND, $T_a$ = 0 ～ +70°C, unless otherwise specified.)

| Item | Symbol | Test Condition | HD63P05Y1 | | | HD63PA05Y1 | | | HD63PB05Y1 | | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | min | typ | max | min | typ | max | min | typ | max | |
| Cycle Time | $t_{cyc}$ | | 1 | — | 10 | 0.666 | — | 10 | 0.5 | — | 10 | $\mu$s |
| Enable Rise Time | $t_{Er}$ | | — | — | 20 | — | — | 20 | — | — | 20 | ns |
| Enable Fall Time | $t_{Ef}$ | | — | — | 20 | — | — | 20 | — | — | 20 | ns |
| Enable Pulse Width ("High" Level) | $PW_{EH}$ | | 450 | — | — | 300 | — | — | 220 | — | — | ns |
| Enable Pulse Width ("Low" Level) | $PW_{EL}$ | Fig. 1 | 450 | — | — | 300 | — | — | 220 | — | — | ns |
| Address Delay Time | $t_{AD}$ | | — | — | 250 | — | — | 190 | — | — | 180 | ns |
| Address Hold Time | $t_{AH}$ | | 40 | — | — | 30 | — | — | 20 | — | — | ns |
| Data Delay Time | $t_{DW}$ | | — | — | 250 | — | — | 160 | — | — | 120 | ns |
| Data Hold Time (Write) | $t_{HW}$ | | 40 | — | — | 30 | — | — | 20 | — | — | ns |
| Data Set-up Time (Read) | $t_{DSR}$ | | 80 | — | — | 60 | — | — | 50 | — | — | ns |
| Data Hold Time (Read) | $t_{HR}$ | | 0 | — | — | 0 | — | — | 0 | — | — | ns |

● **Port Electrical Characteristics** ($V_{CC}$ = 5.0V ± 10%, $V_{SS}$ = GND, $T_a$ = 0 ～ +70°C, unless otherwise specified.)

| Item | | Symbol | Test Condition | min | typ | max | Unit |
|---|---|---|---|---|---|---|---|
| Output voltage "High" | Ports A, B, C | $V_{OH}$ | $I_{OH}$ = −200$\mu$A | 2.4 | — | — | V |
| | | | $I_{OH}$ = −10$\mu$A | $V_{CC}$ − 0.7 | — | — | V |
| Output voltage "Low" | | $V_{OL}$ | $I_{OL}$ = 1.6mA | — | — | 0.55 | V |
| Input voltage "High" | Ports A, B, C, D | $V_{IH}$ | | 2.0 | — | $V_{CC}$ + 0.3 | V |
| Input voltage "Low" | | $V_{IL}$ | | − 0.3 | — | 0.8 | V |
| Input leakage current | | $|I_{IL}|$ | $V_{in}$ = 0.5 ～ $V_{CC}$ − 0.5V | — | — | 1 | $\mu$A |

● **Port Timing** ($V_{CC}$ = 5.0V ± 10%, $V_{SS}$ = GND, $T_a$ = 0 ～ +70°C, unless otherwise noted.)

| Item | Symbol | Test Condition | HD63P05Y1 | | | HD63PA05Y1 | | | HD63PB05Y1 | | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | min | typ | max | min | typ | max | min | typ | max | |
| Port Data Set-up Time (Port A, B, C, D) | $t_{PDS}$ | Fig. 2 | 200 | — | — | 200 | — | — | 200 | — | — | ns |
| Port Data Hold Time (Port A, B, C, D) | $t_{PDH}$ | | 200 | — | — | 200 | — | — | 200 | — | — | ns |
| Port Data Delay Time (Port A, B, C) | $t_{PDW}$ | Fig. 3 | — | — | 300 | — | — | 300 | — | — | 300 | ns |

● Control Signal Timing ($V_{CC}$ = 5.0V ±10%, $V_{SS}$ = GND, Ta = 0 ~ +70°C, unless otherwise noted.)

| Item | Symbol | Test Condition | HD63P05Y1 | | | HD63PA05Y1 | | | HD63PB05Y1 | | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | min | typ | max | min | typ | max | min | typ | max | |
| $\overline{INT}$ Pulse Width | $t_{IWL}$ | | $t_{cyc}$+250 | — | — | $t_{cyc}$+200 | — | — | $t_{cyc}$+200 | — | — | ns |
| $\overline{INT_2}$ Pulse Width | $t_{IWL2}$ | | $t_{cyc}$+250 | — | — | $t_{cyc}$+200 | — | — | $t_{cyc}$+200 | — | — | ns |
| $\overline{RES}$ Pulse Width | $t_{RWL}$ | | 5 | — | — | 5 | — | — | 5 | — | — | $t_{cyc}$ |
| Control Set-up Time | $t_{CS}$ | Fig. 5 | 250 | — | — | 250 | — | — | 250 | — | — | ns |
| Timer Pulse Width | $t_{TWL}$ | | $t_{cyc}$+250 | — | — | $t_{cyc}$+200 | — | — | $t_{cyc}$+200 | — | — | ns |
| Oscillation Start Time (Crystal) | $t_{OSC}$ | Fig.5,Fig.20* | — | — | 20 | — | — | 20 | — | — | 20 | ms |
| Reset Delay Time | $t_{RHL}$ | Fig. 19 | 80 | — | — | 80 | — | — | 80 | — | — | ms |

* $C_L$ = 22pF ±20%, $R_s$ = 60Ω max.

● SCI Timing ($V_{CC}$ = 5.0V ±10%, $V_{SS}$ = GND, $T_a$ = 0 ~ +70°C, unless otherwise specified.)

| Item | Symbol | Test Condition | HD63P05Y1 | | | HD63PA05Y1 | | | HD63PB05Y1 | | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | min | typ | max | min | typ | max | min | typ | max | |
| Clock cycle | $t_{Scyc}$ | | 1 | — | 32768 | 0.67 | — | 21845 | 0.5 | — | 16384 | μs |
| Data output delay time | $t_{TXD}$ | Fig. 6, Fig. 7 | — | — | 250 | — | — | 250 | — | — | 250 | ns |
| Data set-up time | $t_{SRX}$ | | 200 | — | — | 200 | — | — | 200 | — | — | ns |
| Data hold time | $t_{HRX}$ | | 100 | — | — | 100 | — | — | 100 | — | — | ns |
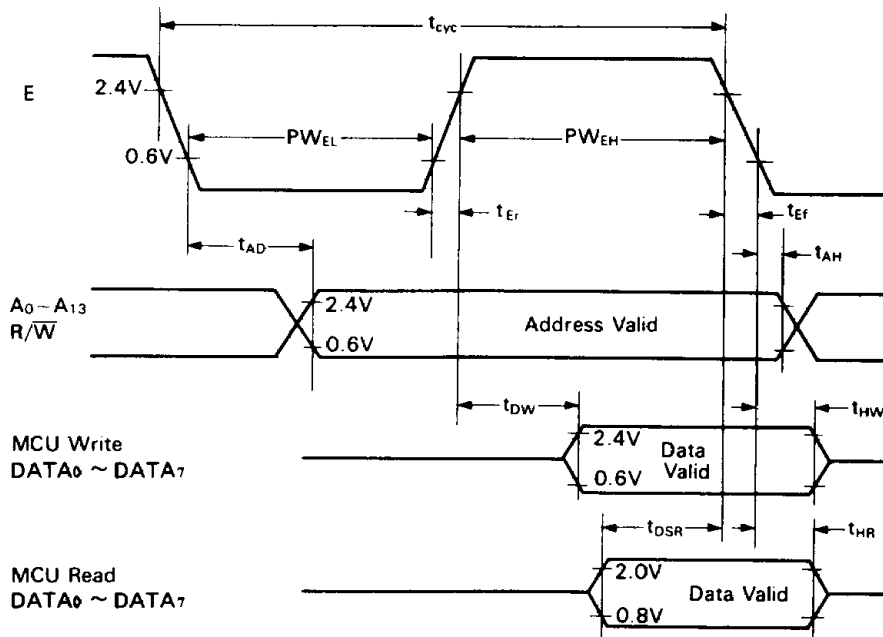
Figure 1  Bus Timing
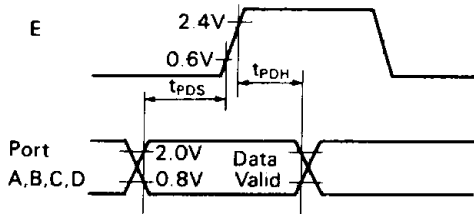


Figure 2  Port Data Set-up and Hold Times (MCU Read)
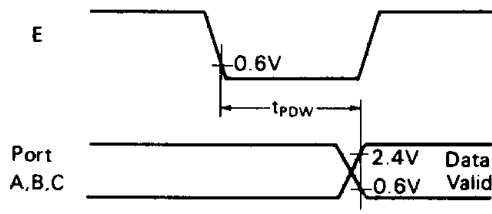


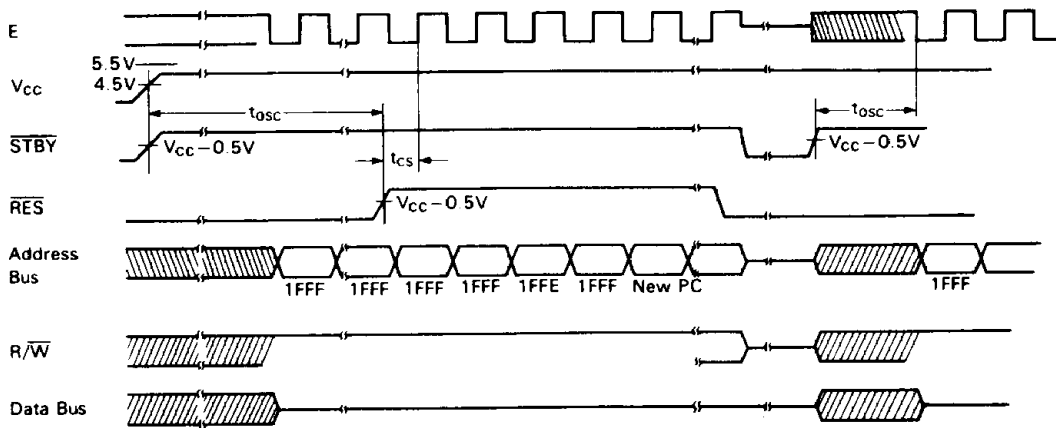Figure 3  Port Data Delay Time (MCU Write)
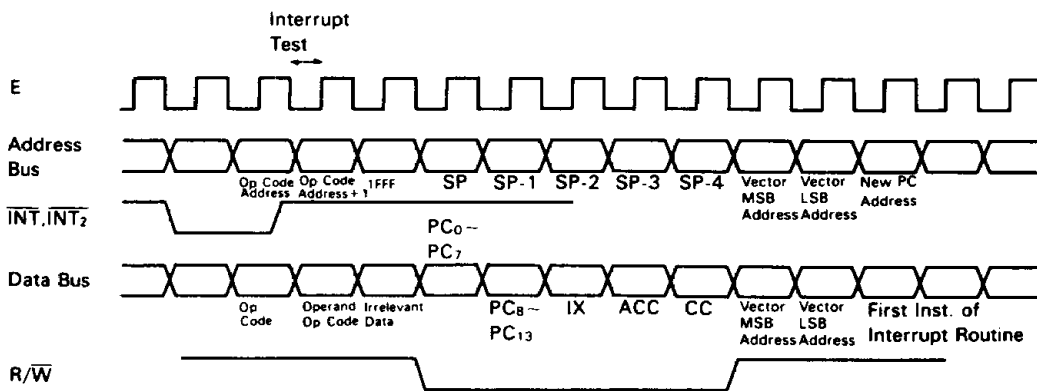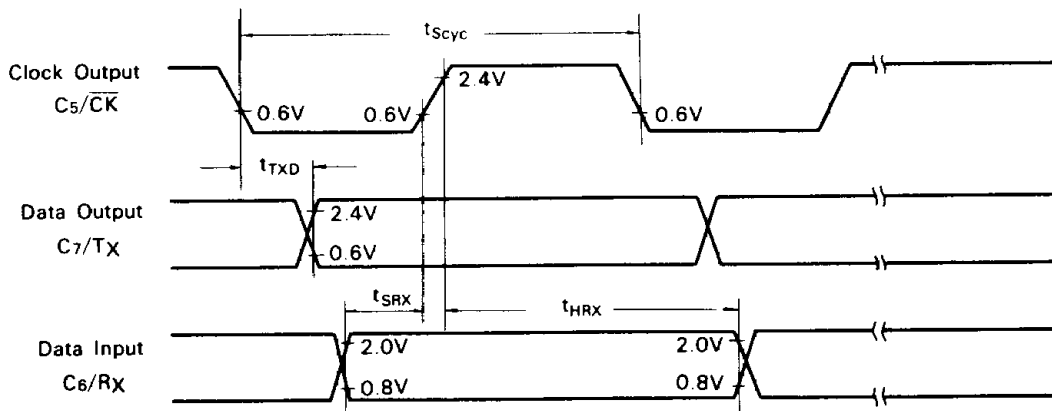


Figure 4  Interrupt Sequence
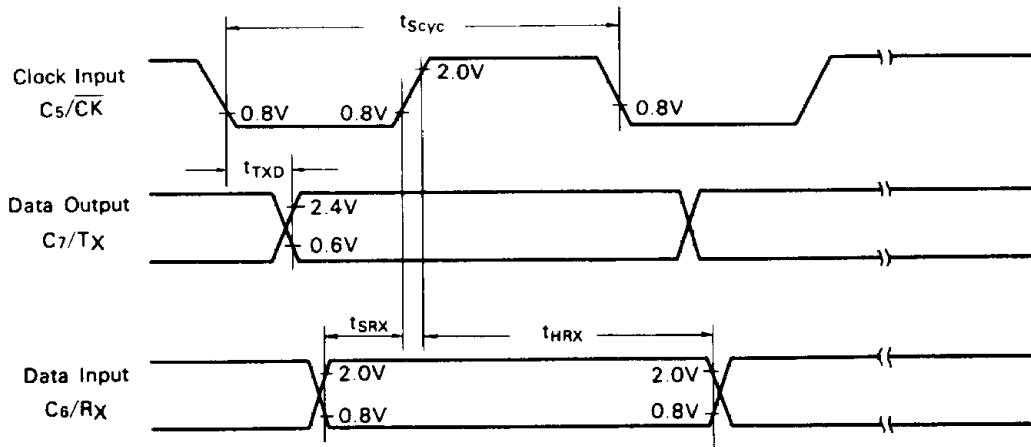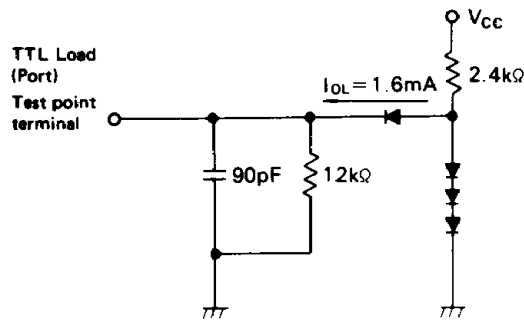
Figure5 Reset Timing

Figure6 SCI Timing (Internal Clock)

Figure7 SCI Timing(External Clock)

TTL Load
(Port)
Test point terminal

$I_{OL} = 1.6mA$    2.4kΩ

$V_{CC}$

90pF    12kΩ

[NOTES]  1. The load capacitance includes stray capacitance caused by the probe, etc.
2. All diodes are 1S2074 (H)

Figure 8   Test Load

## ■DESCRIPTION ON PIN FUNCTIONS

Here is the description of HD63P05Y1 MCU input and output signals.

### ●$V_{CC}$, $V_{SS}$

Power is supplied to the MCU using these two pins. When the operating voltage of the EPROM is 5.0V ± 5%, change $V_{CC}$ according to that of EPROM.

### ●INT, $\overline{INT_2}$

Used for requesting an external interrupt to the MCU. For details, see "INTERRUPT". The $\overline{INT_2}$ is used as the port $D_6$ pin.

### ●XTAL, EXTAL

Are input pins to the internal clock circuit. A crystal oscillator (AT cut, 2.0 to 8.0 MHz) or ceramic oscillator is connected to these pins. For instance, in order to obtain the system clock 1 MHz, a 4 MHz resonant fundamental crystal is useful because the divide-by-4 circuitry is included. EXTAL accepts an external clock input of duty 50% (±10%) to drive, then the internal clock is a quarter the frequency of the external clock. External drive frequency will be 4 or less times the maximum internal clock. For external driving, no XTAL should be connected. Refer to "INTERNAL OSCILLATOR" for using these input pins.

### ●TIMER

Is an external input pin to control the internal Timer. For details, see "TIMER".

### ●$\overline{RES}$

Is used for resetting MCU. For details, see "RESET".

### ●NUM

Is not for user application. It must be connected to $V_{CC}$ through 1kΩ resistance.

### ●Input/Output Pins ($A_0 \sim A_7$, $B_0 \sim B_7$, $C_0 \sim C_7$)

24 pins consist of three 8-bit I/O ports (A, B, C). Each of them is used as input or output pin, through program control

of the data direction register. For details, see "I/O PORTS".

### ●Input Pins ($D_1 \sim D_7$)

Are 7 input-only pins compatible with the TTL and CMOS. $D_6$ is used as $\overline{INT_2}$. When the $D_6$ is used as the port, set the $\overline{INT_2}$ interrupt mask bit of the miscellaneous register to "1" to prevent an $\overline{INT_2}$ from accidental interruption.

### ● Enable (E)

Supplies E clock. Output is a single-phase, TTL compatible and 1/4 crystal oscillation frequency or 1/4 external clock frequency. It can drive one TTL load and a 90pF condenser.

### ● Read/Write (R/$\overline{W}$)

Is an output pin compatible with the TTL. This indicates to peripheral and memory devices whether MCU is in Read ("High"), or in Write ("Low"). The normal standby state is Read ("High"). Its output can drive one TTL load and a 90pF condenser.

### ●Data Bus ($DATA_0 \sim DATA_7$)

Are three-state buffers compatible with the TTL. Each of them can drive one TTL load and 90pF.

### ●Address Bus ($ADR_0 \sim ADR_{13}$)

Are compatible with the TTL and can drive one TTL load and 90pF.

### ●$\overline{STBY}$

Used for bringing the MCU into the standby mode. With $\overline{STBY}$ at "Low" level, the oscillation stops and internal situation is reset. For details, see "STANDBY MODE". The following are I/O pins for serial communication interface (SCI), and used as ports $C_5$, $C_6$, and $C_7$. For details, see "SERIAL COMMUNICATION INTERFACE".

### ●$\overline{CK}$ ($C_5$)

Used to input or output clocks when receiving or transmitting serial data.

### ●Rx ($C_6$)

Used to receive serial data.

### ●Tx ($C_7$)

Used to transmit serial data.

## ■MEMORY MAP

The memory map of the HD63P05Y1 MCU is shown in Fig. 9. During interrupt, the contents of the registers are saved in the stack as shown in Fig. 10. The saving begins with the lower byte (PCL) of the program counter. Then the stack pointer value is decremented, and the higher byte (PCH) of the program counter, index register (X), accumulator (A) and condition code register (CC) are stacked in this order. In subroutine calls, only the contents of the program counter (PCH and PCL) are stacked.
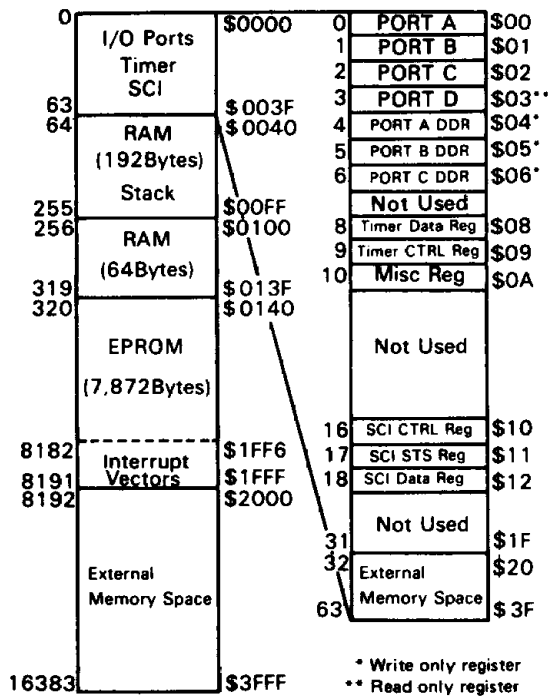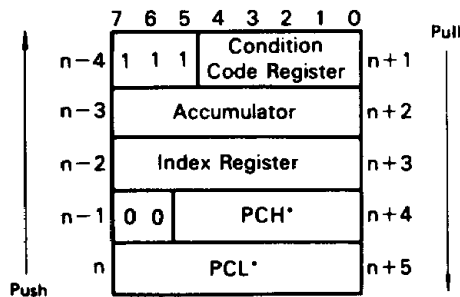
Figure 9   Memory Map of HD63P05Y1 MCU



* In a subroutine call, only PCL and PCH are stacked.

Figure 10   Sequence of Interrupt Stacking

# ■REGISTERS

There are five registers which the programmers can handle.

## ●Accumulator (A)

The accumulator is a general purpose 8-bit register which holds operands, the results of arithmetic operations or data processing.

## ●Index Register (X)

The index register is an 8-bit register used for the index addressing mode. It contains an 8-bit value to be added to an instruction value to create an effective address. The index register can also be used for data manipulations using the read-modify-write instruction. The index register may also be used as a temporary storage area.

## ●Program Counter (PC)
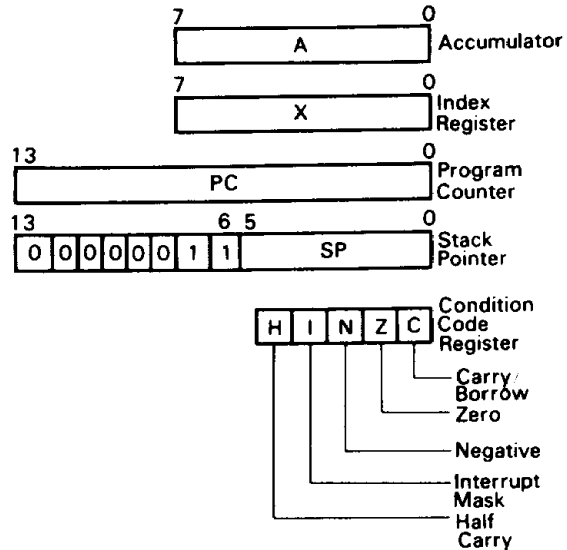


Figure 11   Programming Model

The program counter is a 14-bit register which contains the address of the next instruction to be executed.

## ●Stack Pointer (SP)

The stack pointer is a 14-bit register which indicates the address of the next free location in the stack. Initially, the stack pointer is set to $00FF. It is decremented as data is pushed in, and incremented as it is pulled out. The upper 8 bits of the stack pointer are fixed to 00000011.

During an MCU reset or when the reset stack pointer (RSP) instruction is executed, the pointer is set to the location $00FF. A subroutine or interrupt may be nested down to location $00C1 which allows programmers to use up to 31 levels of subroutine call or 12 levels of interrupt response.

## ●Condition Code Register (CC)

The condition code register is a 5-bit register. Each bit indicates the result of the executed instruction. These bits can be individually tested by conditional branch instructions. The CC bits are as follows.

Half Carry (H): Used to indicate a carry occurring between bits 3 and 4 during an arithmetic operation (ADD, ADC).

Interrupt (I): Setting this bit causes all interrupts to be masked except for software ones. If an interrupt occurs while the bit I is set, the interrupt is latched, and processed as soon as the interrupt mask bit (I) is reset. (Exactly, the interrupt enters the processing routine after the instruction next to the CLI is executed.)

Negative (N): Used to indicate that the result of the latest arithmetic operation, logical operation or data processing is negative (Bit 7 is logical "1").

Zero (Z): Used to indicate that the result of the latest arithmetic operation, logical operation or data processing is zero.

Carry/Borrow (C): Shows a carry or borrow occurring in the latest arithmetic operation. This bit is also affected by the Bit Test and Branch, Shift and

Rotate instructions.

## ■INTERRUPT

There are six different types of interrupt: external interrupt ($\overline{\text{INT}}$, $\overline{\text{INT}_2}$), internal timer interrupts (TIMER, TIMER 2), serial interrupt (SCI) and interrupt by an instruction (SWI).

Of these six interrupts, the $\overline{\text{INT}_2}$ and TIMER, and SCI and TIMER 2 respectively generate the same vector address. When an interrupt occurs, the program in execution stops and CPU state at the interrupt is saved onto the stack. In addition, the interrupt causes the interrupt mask bit (I) in the condition code register to be set and obtains the start address of the interrupt routine from an assigned interrupt vector address before the interrupt routine starts from the state address. The system exits from the interrupt routine by RTI instruction. When the RTI instruction is executed, the CPU state before the interrupt (saved in the stack) is pulled and the CPU starts the program again from the next step to the interrupted one. Table 1. lists the priority of interrupts and their vector addresses.

**Table 1 Priority of Interrupts**

| Interrupt | Priority | Vector Address | |
|---|---|---|---|
| RES | 1 | $1FFE, | $1FFF |
| SWI | 2 | $1FFC, | $1FFD |
| INT | 3 | $1FFA, | $1FFB |
| TIMER/$\overline{\text{INT}_2}$ | 4 | $1FF8, | $1FF9 |
| SCI/TIMER$_2$ | 5 | $1FF6, | $1FF7 |

A flow chart of the interrupt is shown in Fig. 12. Also a block diagram of the interrupt request source is shown in Fig. 13. In the block diagram, both the external interrupts $\overline{\text{INT}}$ and $\overline{\text{INT}_2}$ are edge trigger inputs. At the falling edge of the input, an interrupt request is generated and latched.

The $\overline{\text{INT}}$ interrupt request is automatically cleared if a program jumps to the $\overline{\text{INT}}$ routine. In the case of $\overline{\text{INT}_2}$, the
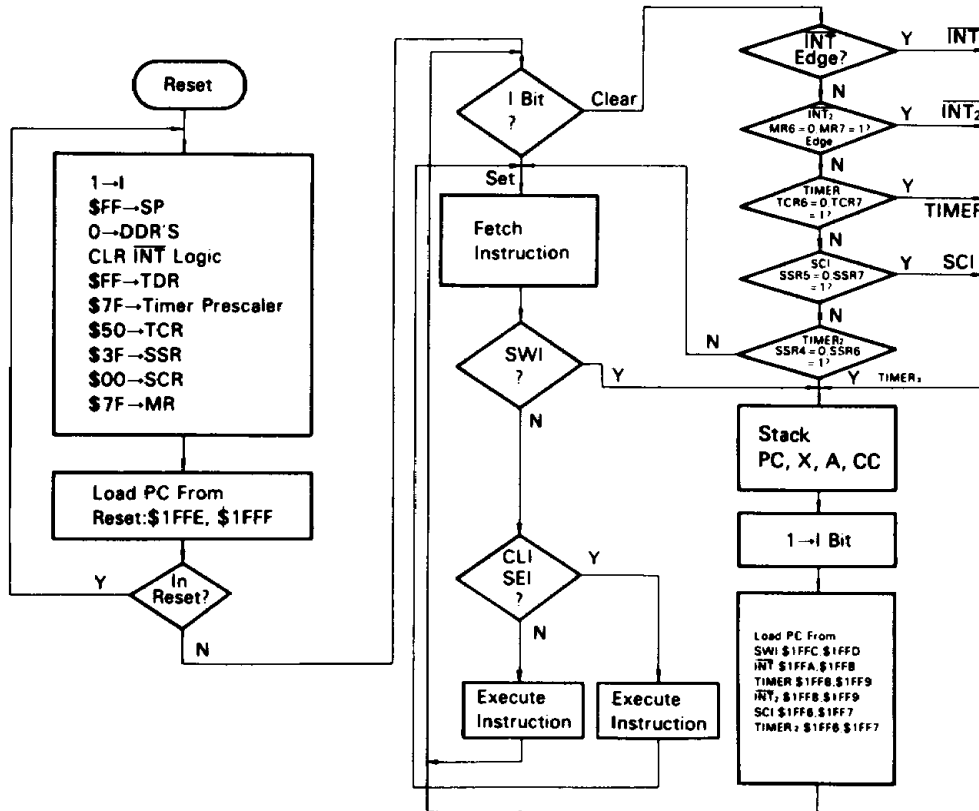


Figure 12 Interrupt Flowchart

interrupt request is cleared when "0" is written in bit 7 of the miscellaneous register. For external interrupts($\overline{\text{INT}}$,$\overline{\text{INT}_2}$), internal timer interrupts (TIMER, TIMER2) and serial interrupt (SCI), these interrupt requests are held, but not operated, while bit I of the condition code register is set. Immediately after the bit I is cleared, the corresponding interrupt is activated.

The $\overline{\text{INT}_2}$ interrupt can be masked by setting bit 6 of the

miscellaneous register; the TIMER interrupt by bit 6 of the timer control register, the SCI interrupt by bit 5 of the serial status register and the TIMER2 interrupt by bit 4 of the serial status register.

The state of the $\overline{\text{INT}}$ pin is tested by BIL or BIH instructions. The $\overline{\text{INT}}$ falling edge detector circuit and its latch circuit are independent of tests by these instructions. The state of $\overline{\text{INT}_2}$ pin is also independent.
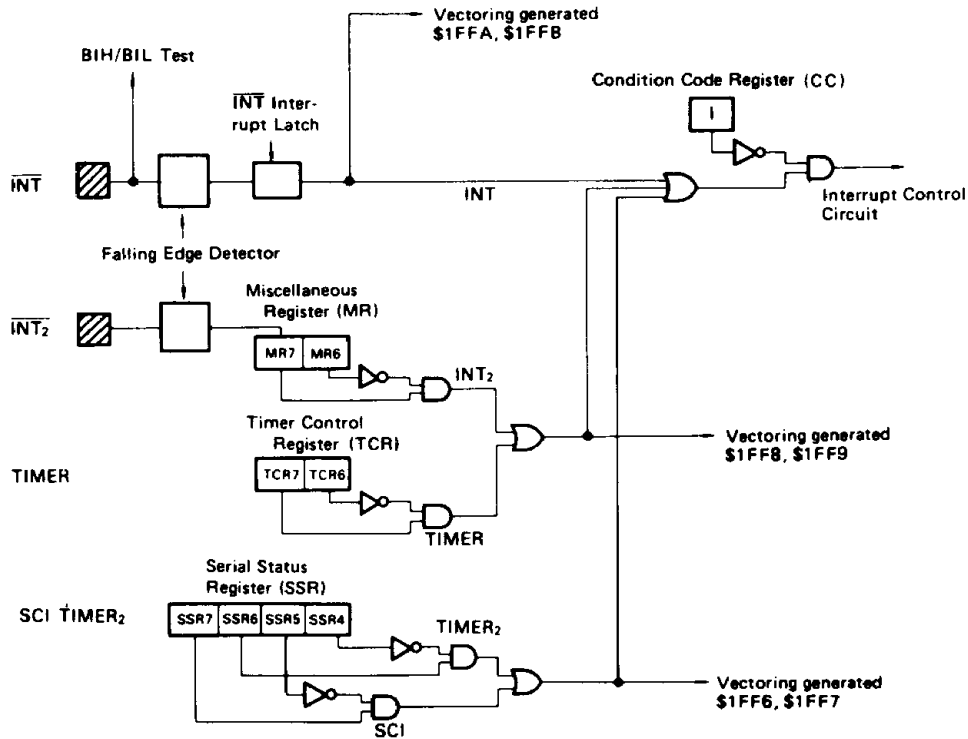
Figure 13   Interrupt Request Generation Circuitry

● **Miscellaneous Register (MR: $000A)**

The interrupt vector address for external interrupt $\overline{INT_2}$ is the same as that for the TIMER interrupt, as shown in Table 1. For this reason, a special register called a miscellaneous register (MR: $000A) is available for $\overline{INT_2}$ interrupt control. Bit 7 of the miscellaneous register is of $\overline{INT_2}$ interrupt request flag. When the falling edge is detected at the $\overline{INT_2}$ pin, "1" is set in bit 7. The software in the interrupt routine (vector address: $1FF8, $1FF9) checks to see if it is $\overline{INT_2}$ interrupt. Bit 7 is reset by software. Bit 6 is the $\overline{INT_2}$ interrupt mask bit. If the bit is set to "1", the $\overline{INT_2}$ interrupt is disabled.

**Miscellaneous Register (MR:$000A)**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| MR7 | MR6 | / | / | / | / | / | / |

└─────────── $\overline{INT_2}$ Interrupt Mask
└─────────────────── $\overline{INT_2}$ Interrupt Request Flag
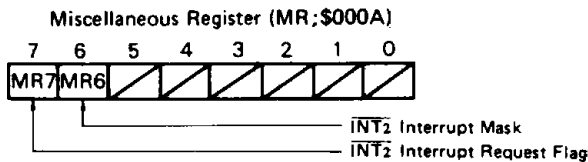
Both "READ" and "WRITE" are possible with bit 7, but "1" can not be written to in this bit by software. Therefore, interrupt requests by software are not possible. By resetting, bit 7 is cleared and bit 6 is entered "1".

■ **TIMER**

The MCU timer block diagram is shown in Fig. 14. The 8-bit counter is loaded under program control and is decremented by the clock input. When the timer data register (TDR) reaches 0, the timer interrupt request bit (bit 7) in the timer control register is set. The MCU responds to this interrupt by saving the present CPU state in the stack, fetching the

timer interrupt routine address from address $1FF8 and $1FF9. The timer interrupt can be masked by setting the timer interrupt mask bit (bit 6) in the timer control register. The mask bit (I) in the condition code register can also disable the timer interrupt. The source clock for the timer can be either an external signal from the timer input pin or the internal E signal (oscillator clock divided by 4). If the E signal is selected as the source, the clock input can be gated by the input to the timer input pin.

When the timer counter reaches "0", it starts counting down from $FF. The count can be monitored at any time by reading the timer data register. This function allows knowledge of the length of time after a timer interrupt with a program, without destroying the contents of the counter.
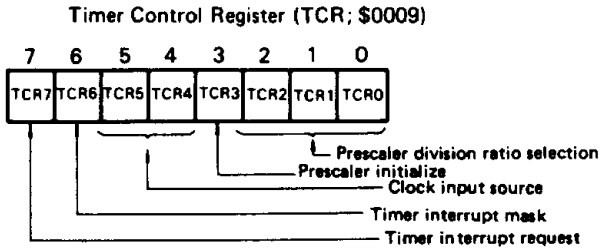
When the MCU is reset, both the prescaler and counter return to the initial state of logical "1". At the same time, the timer interrupt request bit (bit 7) is cleared and the timer interrupt mask bit (bit 6) is set. Write "0" in the timer interrupt request bit (bit 7) to clear it.

| TCR7 | Timer interrupt request |
|------|------------------------|
| 0 | Absent |
| 1 | Present |

| TCR6 | Timer interrupt mask |
|------|---------------------|
| 0 | Enabled |
| 1 | Disabled |

● **Timer Control Register (TCR; $0009)**

Selection of a clock source, selection of a prescaler frequency division ratio, and a timer interrupt can be controlled by the timer control register (TCR; $0009).

For the selection of a clock source, any one of the four modes (see Table 2) can be selected by bits 5 and 4 of the timer control register (TCR).

Timer Control Register (TCR; $0009)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TCR7 | TCR6 | TCR5 | TCR4 | TCR3 | TCR2 | TCR1 | TCR0 |

- Prescaler division ratio selection
- Prescaler initialize
- Clock input source
- Timer interrupt mask
- Timer interrupt request

After resetting, the TCR is initialized to "E under timer terminal control" (bit 5 = 0, bit 4 = 1). If the timer terminal is "1", the counter starts counting down with "$FF" immediately after the reset.

Table 2  Clock Source Selection

| TCR | | Clock input source |
|---|---|---|
| Bit 5 | Bit 4 | |
| 0 | 0 | Internal clock E |
| 0 | 1 | E under timer terminal control |
| 1 | 0 | No clock input (counting stopped) |
| 1 | 1 | Event input from timer terminal |
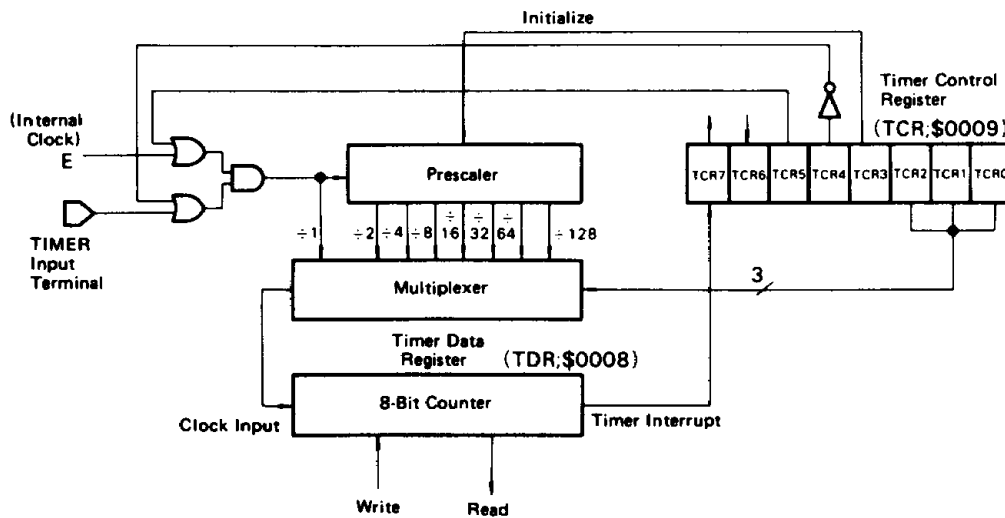


Figure 14  Timer Block Diagram

Table 3  Prescaler Division Ratio Selection

| TCR | | | Prescaler division ratio |
|---|---|---|---|
| Bit 2 | Bit 1 | Bit 0 | |
| 0 | 0 | 0 | ÷1 |
| 0 | 0 | 1 | ÷2 |
| 0 | 1 | 0 | ÷4 |
| 0 | 1 | 1 | ÷8 |
| 1 | 0 | 0 | ÷16 |
| 1 | 0 | 1 | ÷32 |
| 1 | 1 | 0 | ÷64 |
| 1 | 1 | 1 | ÷128 |

The prescaler is initialized by writing "1" in bit 3. The bit is always "0", when "READ". A prescaler division ratio is selected by a combination of the three bits (bits 0, 1 and 2) of the timer control register (See Table 3). There are eight division ratios; ÷1, ÷2, ÷4, ÷8, ÷16, ÷32, ÷64 and ÷128.

After resetting, the TCR returns to the ÷1 mode. The timer interrupt is enabled when the timer interrupt mask bit is "0", and disabled when the bit is "1". When a timer interrupt occurs, "1" is set in the timer interrupt request bit. The bit is cleared by writing "0" into it.

■**SERIAL COMMUNICATION INTERFACE (SCI)**

Used for 8-bit data communication. Transfer rate ranges from 1μs to about 32 ms (when oscillated at 4 MHz), and there are sixteen selections.

The SCI consists of three registers, one octal counter and one prescaler. (See Fig. 15) The SCI communicates with the CPU through the data bus, and with peripherals through bits 5, 6 and 7 of port C. Operations of the registers and data transfer are described below.
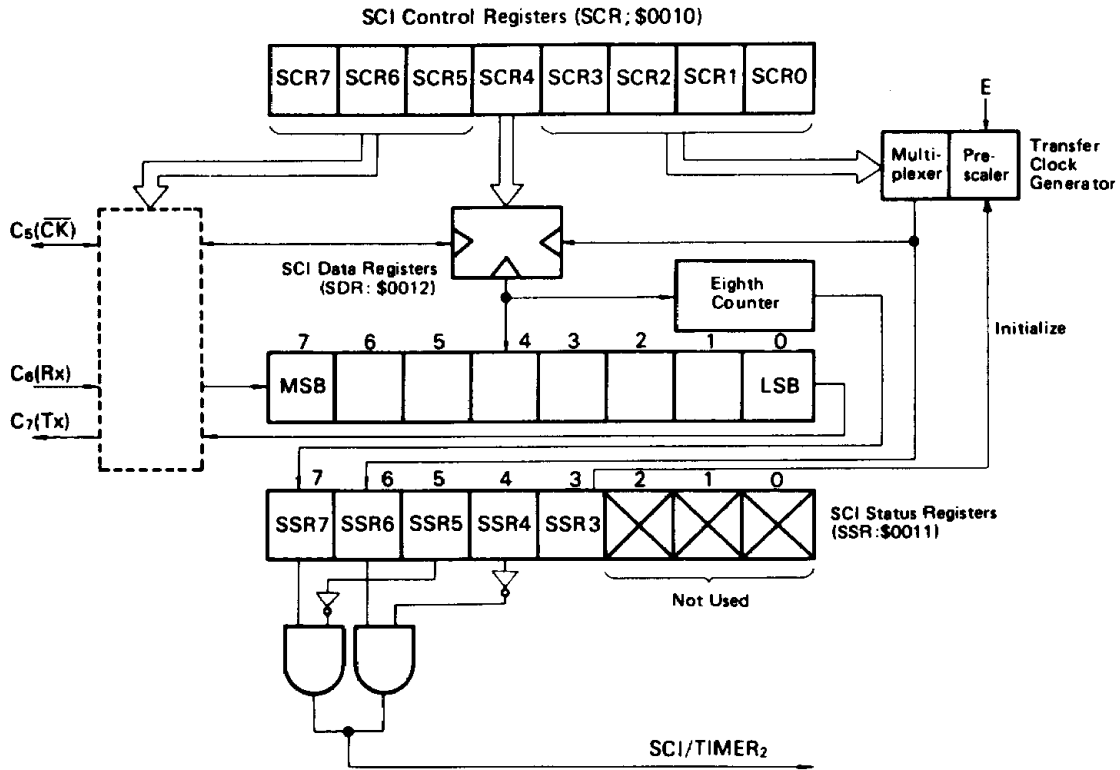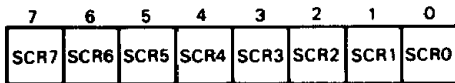
Figure 15  SCI Block Diagram

● **SCI Control Register (SCR; $0010)**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SCR7 | SCR6 | SCR5 | SCR4 | SCR3 | SCR2 | SCR1 | SCR0 |

| SCR7 | C₇ terminal |
|---|---|
| 0 | Used as I/O terminal (by DDR). |
| 1 | Serial data output (DDR output) |

| SCR6 | C₆ terminal |
|---|---|
| 0 | Used as I/O terminal (by DDR). |
| 1 | Serial data input (DDR input) |

| SCR5 | SCR4 | Clock source | C₅ terminal |
|---|---|---|---|
| 0 | 0 | — | Used as I/O terminal (by DDR). |
| 0 | 1 | — | |
| 1 | 0 | Internal | Clock output (DDR output) |
| 1 | 1 | External | Clock input (DDR input) |

**Bit 7 (SCR7)**

When this bit is set, the DDR corresponding to the C₇ becomes "1" and this terminal serves for output of SCI data. After resetting the bit is cleared to "0".

**Bit 6 (SCR6)**

When this bit is set, the DDR corresponding to the C₆ becomes "0" and this terminal serves for input of SCI data. After resetting the bit is cleared to "0".

**Bits 5 and 4 (SCR5, SCR4)**

These bits are used to select a clock source. After resetting the bits are cleared to "0".

**Bits 3 ~ 0 (SCR3 ~ SCR0)**

These bits are used to select a transfer clock rate. After resetting the bits are cleared to "0".

| SCR3 | SCR2 | SCR1 | SCR0 | Transfer clock rate | |
|---|---|---|---|---|---|
| | | | | 4.00 MHz | 4.194 MHz |
| 0 | 0 | 0 | 0 | 1 μs | 0.95 μs |
| 0 | 0 | 0 | 1 | 2 μs | 1.91 μs |
| 0 | 0 | 1 | 0 | 4 μs | 3.82 μs |
| 0 | 0 | 1 | 1 | 8 μs | 7.64 μs |
| ⟨ | ⟨ | ⟨ | ⟨ | ⟨ | ⟨ |
| 1 | 1 | 1 | 1 | 32768 μs | 1/32 s |

**●SCI Data Register (SDR; $0012)**

A serial-parallel conversion register that is used for transfer of data.

**●SCI Status Register (SSR; $0011)**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SSR7 | SSR6 | SSR5 | SSR4 | SSR3 | ✕ | ✕ | ✕ |

**Bit 7 (SSR7)**

Bit 7 is the SCI interrupt request bit which is set on completion of transmitting or receiving 8-bit data. It is cleared when reset or data is written to or read from the SCI data register with the SCR5="1". The bit can be cleared by writing "0" into it.

**Bit 6 (SSR6)**

Bit 6 is the TIMER2 interrupt request bit. TIMER2 is commonly used with the serial clock generator, and SSR6 is set each time the internal transfer clock falls. When resetting, the bit is cleared. It can also be cleared by writing "0" into it. (For details, see TIMER2).

**Bit 5 (SSR5)**

Bit 5 is the SCI interrupt mask bit which can be set or cleared by software. When it is "1", the SCI interrupt (SSR7) is masked. When resetting, it is set to "1".

**Bit 4 (SSR4)**

Bit 4 is the TIMER2 interrupt mask bit which can be set or cleared by software. When the bit is "1", the TIMER2 interrupt (SSR6) is masked. When resetting, it is set to "1".

**Bit 3 (SSR3)**

When "1" is written into this bit, the prescaler of the transfer clock generator is initialized. When "READ", the bit is always "0".

**Bits 2 ~ 0**

Not used.

| SSR7 | SCI interrupt request |
|---|---|
| 0 | Absent |
| 1 | Present |

| SSR6 | TIMER2 interrupt request |
|---|---|
| 0 | Absent |
| 1 | Present |

| SSR5 | SCI interrupt mask |
|---|---|
| 0 | Enabled |
| 1 | Disabled |

| SSR4 | TIMER2 interrupt mask |
|---|---|
| 0 | Enabled |
| 1 | Disabled |

**● Data Transmission**

By writing the desired control bits into the SCI control registers, a transfer rate and a transfer clock source are determined and bits 7 and 5 of port C are set at the serial data output terminal and the serial clock terminal, respectively. The transmit data should be stored from the accumulator or index register into the SCI data register. The data written in the SCI data register is output from the C7/Tx terminal, starting with the LSB, synchronously with the falling edge of the serial clock (See Fig. 16). When 8 bits of data have been transmitted, the interrupt request bit is set in bit 7 of the SCI status register with the rising edge of the last serial clock. This request can be masked by setting bit 5 of the SCI status register. Once the data has been sent, the 8th bit data (MSB) stays at the C7/Tx terminal. If an external clock source has been selected, the transfer rate determined by bits 0 to 3 of the SCI control register is ignored, and the Cs/CK terminal is set as input. If the internal clock has been selected, the Cs/CK terminal is set as output and clocks are output at the transfer rate selected by bits 0 to 3 of the SCI control register.
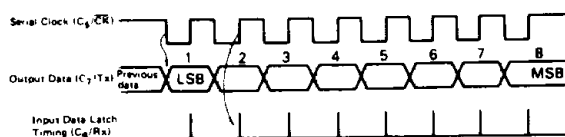


Figure 16  SCI Timing Chart

**● Data Reception**

By writing the desired control bits into the SCI control register, a transfer rate and a transfer clock source are determined and bit 6 and 5 of port C are set at the serial data input terminal and the serial clock terminal, respectively. Then dummy-writing or -reading the SCI data register, the system is ready for receiving data. (This procedure is not needed after reading subsequent received data. It must be taken after reset and after not reading subsequent received data.)

The data from the C6/Rx terminal is input to the SCI data register synchronously with the rising edge of the serial clock (see Fig. 16). When 8 bits of data have been received, the interrupt request bit is set in bit 7 of the SCI status register. This request can be masked by setting bit 5 of the SCI status register. If an external clock source have been selected, the transfer rate determined by bits 0 ~ 3 of the SCI control register is ignored, and the data is received synchronously with the clock from the Cs/CK terminal. If the internal clock has been selected, the Cs/CK terminal is set as output and clocks are output at the transfer rate selected by bits 0 ~ 3 of the SCI control register.

**● TIMER2**

The SCI transfer clock generator can be used as a timer. The clock selected by bits 3 to 0 of the SCI control register (4 μs to approx. 32 ms (when oscillated at 4 MHz)) is input to bit 6 of the SCI status register and the TIMER2 interrupt request bit is set at each falling edge of the clock. Since interrupt requests occur periodically, TIMER2 can be used as a reload counter or clock.
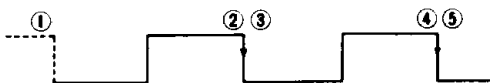
①    : Transfer clock generator is reset and mask bit (bit 4 of SCI, status register) is cleared.

②,④ : TIMER₂ interrupt request

③,⑤ : TIMER₂ interrupt request bit cleared

TIMER₂ is commonly used with the SCI transfer clock generator. If wanting to use TIMER₂ independently of the SCI, specify "External" (SCR5 = 1, SCR4 = 1) as the SCI clock source.

If "Internal" is selected as the clock source, reading or writing the SDR causes the prescaler of the transfer clock generator to be initialized.

## ■I/O PORTS

There are 24 input/output terminals (ports A, B, C). Each I/O terminal can be selected for either input or output by the data direction register. Specifically, an I/O port will be input if "0" is written in the data direction register, and output if "1" is written in the data direction register. Port A, B or C reads latched data if it has been programmed as output, even with the output load, the output level fluctuating. (See Fig. 17).

When resetting the data direction register and data register go to "0" and all input/output terminals are used as input.

Seven input-only terminals are available (port D). Writing to these ones is invalid.

All input/output terminals and input terminals are TTL compatible and CMOS compatible in respect of both input and output.

If I/O ports or input ports are not used, they should be connected to $V_{SS}$ via resistors. With none connected to these terminals, there is the possibility of power being consumed despite their not being used.
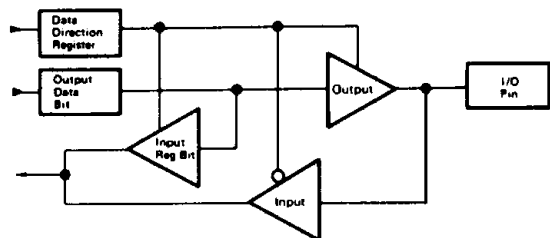


| Bit of data direction register | Bit of output data | Status of output | Input to MCU |
|---|---|---|---|
| 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 |
| 0 | X | 3-state | Pin |

Figure 17   Input/Output Port Diagram

## ■RESET

The MCU can be reset either by external reset input ($\overline{RES}$) or power-on reset. (See Fig. 18). On power up, the reset input must be held "Low" for at least $t_{osc}$ to assure that the internal oscillator is stabilized. A sufficient delay time can be obtained by connecting a capacitance to the $\overline{RES}$ input as shown in Fig. 19.
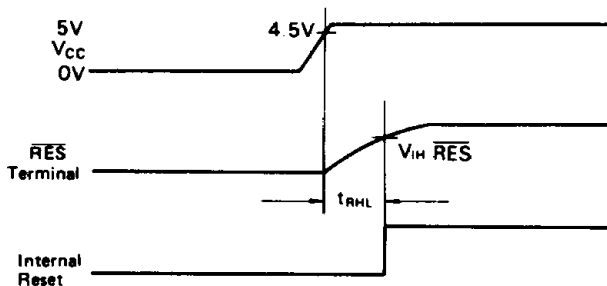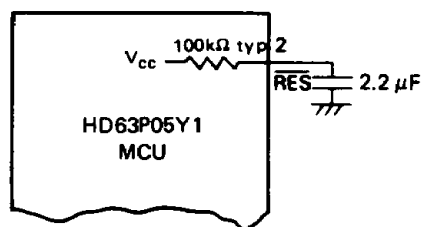


Figure 18   Power On and Reset Timing



Figure 19   Input Reset Delay Circuit

## ■INTERNAL OSCILLATOR

The internal oscillator circuit is designed to meet the



Crystal Oscillator

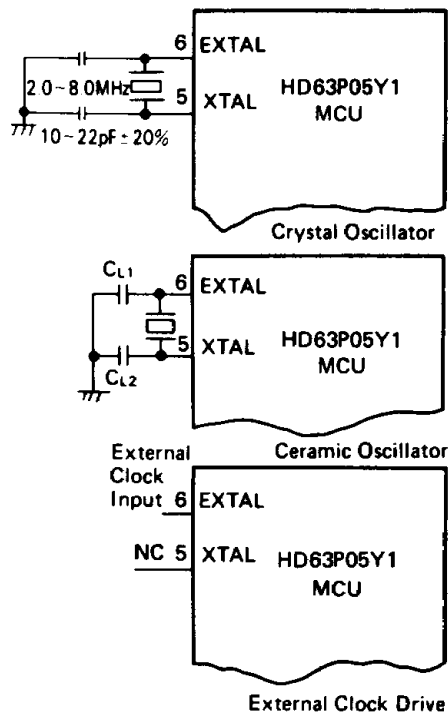Ceramic Oscillator

External Clock Drive

Figure 20   Internal Oscillator Circuit

requirement for minimum external configurations. It can be driven by connecting a crystal (AT cut 2.0 ~ 8.0MHz) or ceramic oscillator between pins 5 and 6 depending on the required oscillation frequency stability.

Three different terminal connections are shown in Fig. 20. Figs. 21 and 22 illustrate the specifications and typical arrangement of the crystal.
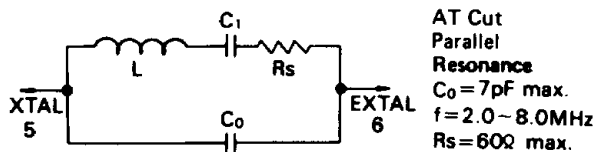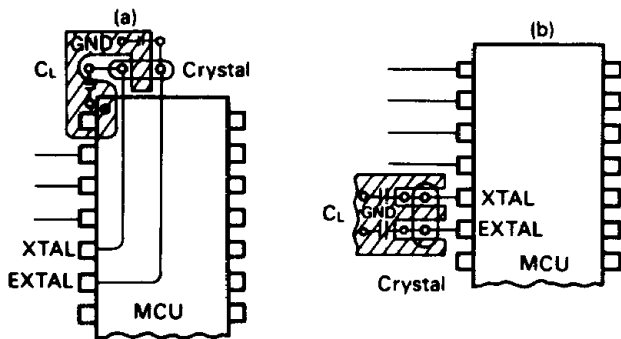


Figure 21  Parameters of Crystal



[NOTE]  Use as short wirings as possible for connection of the crystal with the EXTAL and XTAL terminals. Do not allow these wirings to cross others.

Figure 22  Typical Crystal Arrangement

## ■LOW POWER DISSIPATION MODE

The HD63P05Y1 has three low power dissipation modes: wait, stop and standby.

### ● Wait Mode

When a WAIT instruction being executed, the MCU enters into the wait mode. In this mode, the oscillator stays active but the internal clock stops. The CPU stops but the peripheral functions – the timer and the serial communication interface – stay active. (NOTE: Once the system has entered the wait mode, the serial communication interface can no longer be retriggered.) In the wait mode, the registers, RAM and I/O terminals hold the condition just before entering the wait mode. Both address (Ao ~ A12) and chip enable ($\overline{CE}$) for the EPROM are in "1" state.

Release from this mode can be done by interrupt ($\overline{INT}$, TIMER/$\overline{INT_2}$ or SCI/TIMER2), $\overline{RES}$ or $\overline{STBY}$. The $\overline{RES}$ resets the MCU and the $\overline{STBY}$ brings it into the standby mode. (This will be mentioned later.)

When interrupt is requested to the CPU and accepted, the wait mode is released and the CPU is brought to the operation mode and vectors to the interrupt routine. If the interrupt is masked by the I bit of the condition code register, after release from the wait mode the MCU executes the instruction following WAIT. If an interrupt other than the $\overline{INT}$ (i.e., TIMER/$\overline{INT_2}$ or SCI/TIMER2) is masked by the timer control re-

gister, miscellaneous register or serial status register, there is no interrupt request to the CPU, so the wait mode cannot be released.

Fig. 23 shows a flowchart of the wait function.

### ● Stop Mode

When STOP instruction is being executed, the MCU enters the stop mode. In this mode, the oscillator stops and the CPU and peripheral functions become inactive but the RAM, register and I/O terminals hold the condition they had just before entering the stop mode. Both address (Ao ~ A12) and chip enable ($\overline{CE}$) for the EPROM are in "1" state.

Release from this mode can be done by an external interrupt ($\overline{INT}$ or $\overline{INT_2}$), $\overline{RES}$ or $\overline{STBY}$. The $\overline{RES}$ resets the MCU and the $\overline{STBY}$ brings it into the standby mode.

When an interrupt is requested and accepted by the CPU, the stop mode is released and the CPU is brought in the operation mode and vectors to the interrupt routine. If the interrupt is masked by the I bit of the condition code register, after release from the stop mode, the MCU executes the instruction following STOP. If the $\overline{INT_2}$ interrupt is masked by the miscellaneous register, there is no interrupt request to the MCU, so the stop mode cannot be released.

Fig. 24 shows the flowchart of the stop function. Fig. 25 shows a timing chart of the return to the operation mode from the stop mode.

For releasing from the stop mode by an interrupt, oscillation starts upon input of the interrupt and, after the internal delay time for stabilized oscillation, the CPU becomes active. For restarting by $\overline{RES}$, oscillation starts when the $\overline{RES}$ goes "0" and the CPU restarts when the $\overline{RES}$ goes "1".. The duration of $\overline{RES}$="0" must exceed $t_{osc}$ to assure stabilized oscillation.

### ● Standby Mode

The MCU enters the standby mode when the $\overline{STBY}$ terminal goes "Low". In this mode, all operations stop and the internal condition is reset but the contents of the RAM are held. The I/O terminals turn to high-impedance state. Both address (Ao ~ A12) and chip enable ($\overline{CE}$) for the EPROM are in "1" state. The standby mode should be released by bringing $\overline{STBY}$ "High". The CPU must be restarted by resetting. The timing of input signals at the $\overline{RES}$ and $\overline{STBY}$ terminals is shown in Fig. 26.

Table 4 lists the status of each parts of the MCU in each low power dissipation modes. Transitions between each mode are shown in Fig. 27.
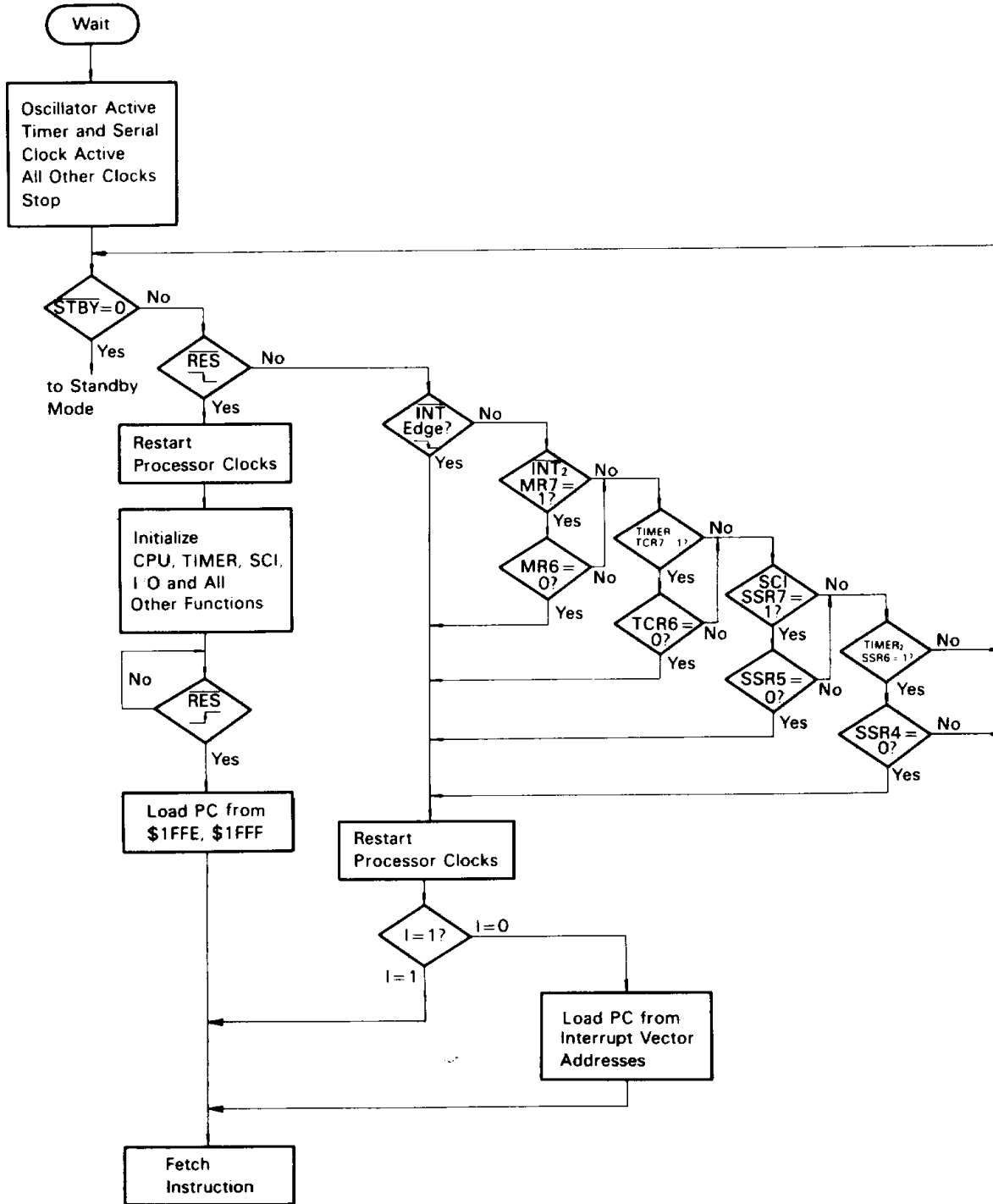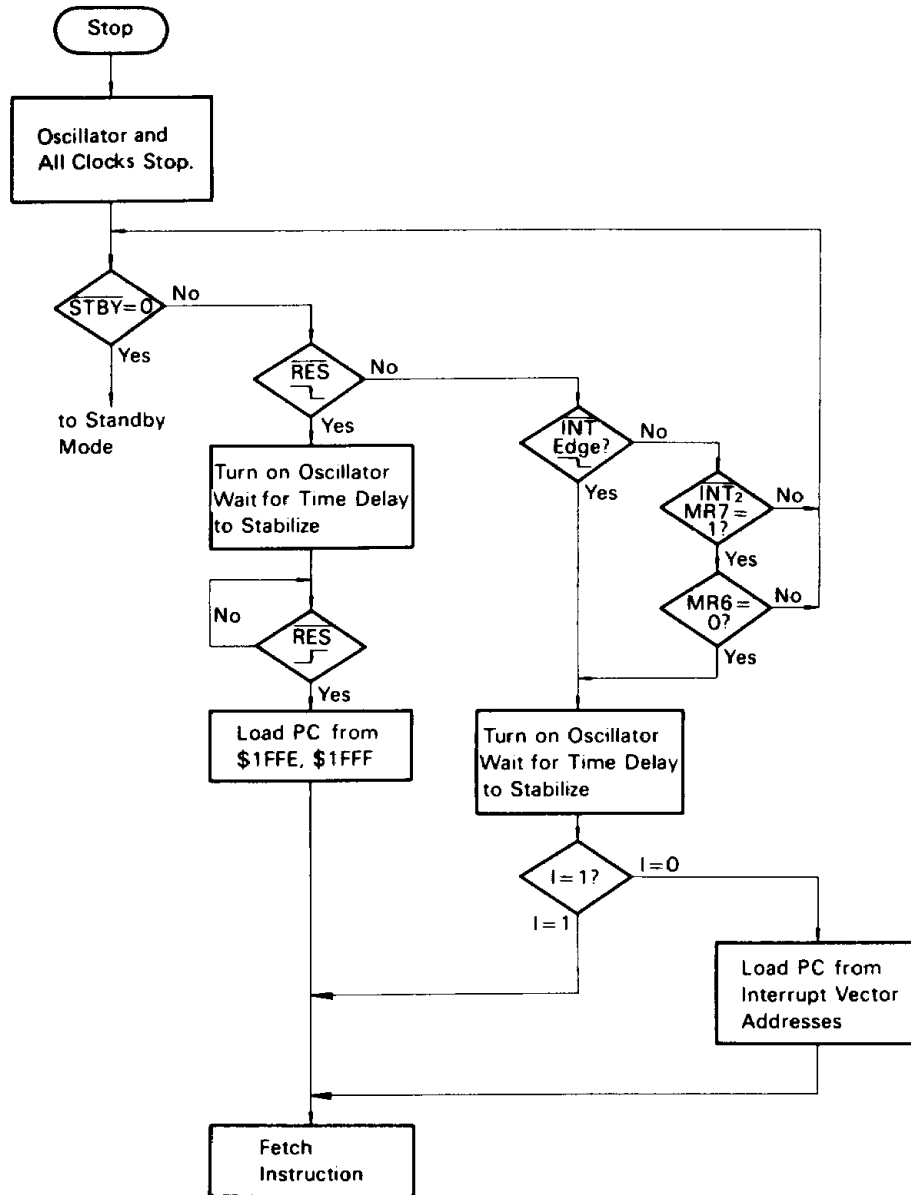
Figure 23  Wait Mode Flow Chart

Figure 24   Stop Mode Flow Chart

(a) Restart by Interrupt
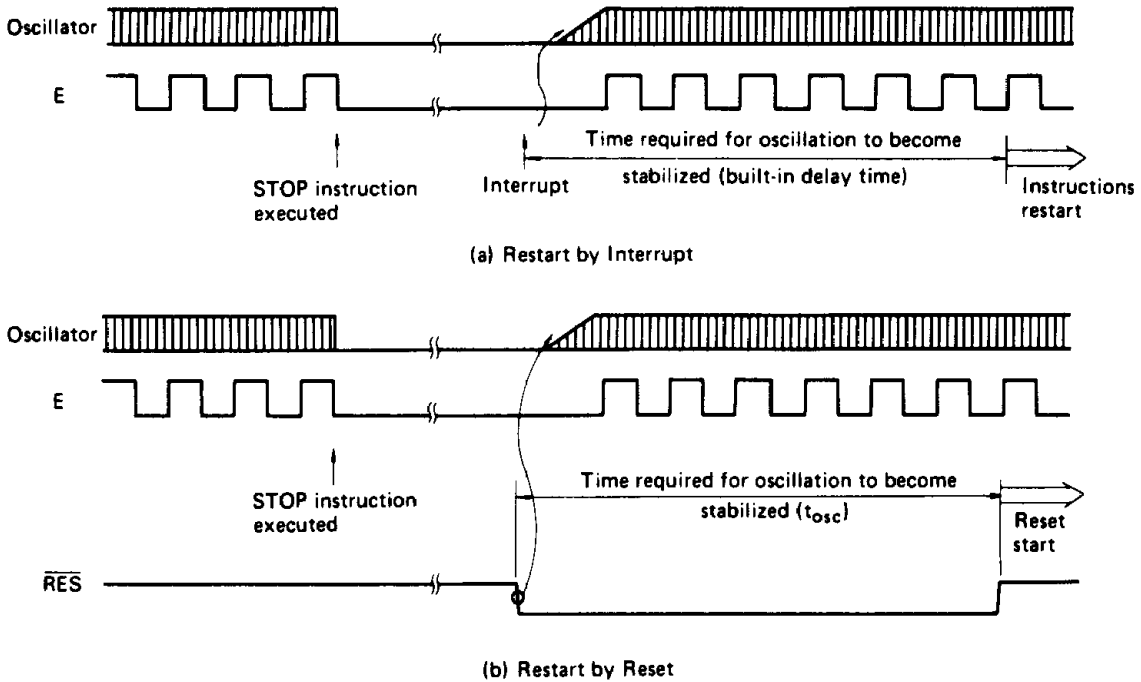


(b) Restart by Reset

Figure 25   Timing Chart of Releasing from Stop Mode
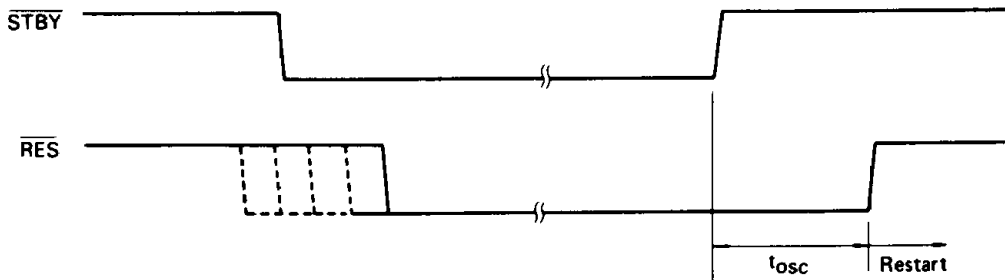


Figure 26   Timing Chart of Releasing from Standby Mode

Table 4   Status of Each Part of MCU in Low Power Dissipation Modes

| Mode | Start | | Condition | | | | | | Escape |
|------|-------|------|-----------------|-----|------|--------|-----|----------------|--------|
| | | | Oscillator | CPU | Timer, Serial | Register | RAM | I/O terminal | |
| WAIT | Software | WAIT instruction | Active | Stop | Active | Hold | Hold | Hold | $\overline{STBY}$, $\overline{RES}$, $\overline{INT}$, $\overline{INT_2}$, each interrupt request of TIMER, TIMER$_2$, SCI |
| STOP | | STOP instruction | Stop | Stop | Stop | Hold | Hold | Hold | $\overline{STBY}$, $\overline{RES}$, $\overline{INT}$, $\overline{INT_2}$ |
| Standby | Hardware | $\overline{STBY}$="Low" | Stop | Stop | Stop | Reset | Hold | High impedance | $\overline{STBY}$="High" |

Figure 27   Transitions among Active Mode, Wait Mode,
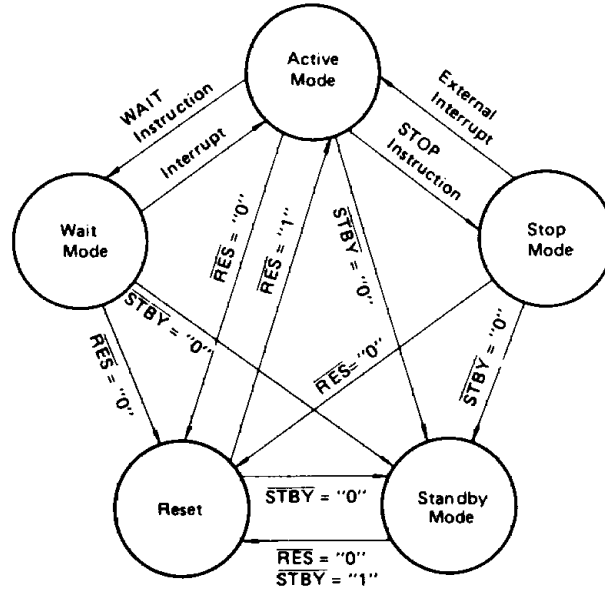Stop Mode, Standby Mode and Reset

## ■ PRECAUTION TO THE BOARD DESIGN OF OSCILLATION CIRCUIT

As shown in Fig.28, the cross talk may disturb normal oscillation if signal lines are set near the oscillation circuit. When designing a board, be careful of this. Crystal and $C_L$ must be put near XTAL and EXTAL pins as possible.
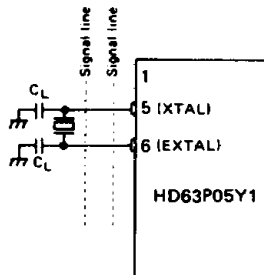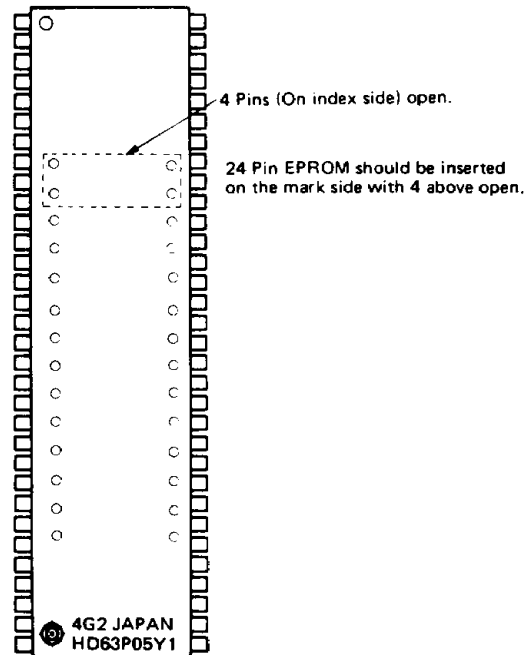


Figure 28   Precaution to the board design of oscillation circuit

## ■ PRECAUTION TO USE THE EPROM ON-PACKAGE 8-BIT SINGLE-CHIP MICROCOMPUTER

Please be careful of the following, since this MCU has a special structure with pin socket on the package.

(1) Don't apply high static voltage or surge voltage over MAXIMUM RATINGS to the socket pins as well as the LSI pins. If so, that may cause permanent damage to the device.

(2) When using 32k EPROM (24-pin), insert it leaving the four pins above open.

(3) When inserting this into system products like mask ROM type single chip microcomputer, be careful of the following to give effective contact between the EPROM pins and socket pins.



4 Pins (On index side) open.

24 Pin EPROM should be inserted on the mark side with 4 above open.

4G2 JAPAN HD63P05Y1

(a) When soldering the LSI onto a printed circuit board, the recommended condition is
    Temperature:   lower than 250°C
    Time:          within 10 sec.

(b) Be careful that detergent or coating does not get into the socket during flux washing or board coating after soldering, because that may cause bad effect on socket contact.

(c) Avoid permanent application of this under conditions

of continuous vibration.

(d) The socket, repeatedly inserted and removed, loses its contactability. It is recommended to use new one when used in production.

## ■ BIT MANIPULATION

The HD63P05Y1 MCU can use a single instruction (BSET or BCLR) to set or clear one bit of the RAM within page 0 or an I/O port (except the write-only registers such as the data direction register). Every bit of memory or I/O within page 0 ($00 ~ $FF) can be tested by the BRSET or BRCLR instruction; depending on the result of the test, the program can branch to required destinations. Since bits in the RAM on page 0, or I/O can be manipulated, the user may use a bit within the RAM on page 0 as a flag or handle a single I/O bit as an independent I/O terminal. Fig. 29 shows an example of bit manipulation and the validity of test instructions. In the example, the program is configured assuming that bit 0 of port A is connected to a zero cross detector circuit and bit 1 of the same port to the trigger of a triac.

The program shown can activate the triac within a time of 10μs from zero-crossing through the use of only 7 bytes on the memory. The on-chip timer provides a required time of delay and pulse width modulation of power is also possible.

```
SELF 1.    BRCLR 0, PORT A, SELF 1
           BSET 1, PORT A
           BCLR 1, PORT A
```

Figure 29    Example of Bit Manipulation

## ■ ADDRESSING MODES

Ten different addressing modes are available to the HD63P05Y1 MCU.

### ● Immediate

See Fig. 30. The immediate addressing mode provides access to a constant which does not vary during execution of the program.

This access requires an instruction length of 2 bytes. The effective address (EA) is PC and the operand is fetched from the byte that follows the operation code.

### ● Direct

See Fig. 31. In the direct addressing mode, the address of the operand is contained in the 2nd byte of the instruction. The user can gain direct access to memory up to the lower 255th address. 192 byte RAM and I/O registers are on page 0 of address space so that the direct addressing mode may be utilized.

### ● Extended

See Fig. 32. The extended addressing is used for referencing to all addresses of memory. The EA is the contents of the 2 bytes that follow the operation code. An extended addressing instruction requires a length of 3 bytes.

### ● Relative

See Fig. 33. The relative addressing mode is used with branch instructions only. When a branch occurs, the program counter is loaded with the contents of the byte following the operation code. EA = (PC) + 2 + Rel., where Rel. indicates a signed 8-bit data following the operation code. If no branch occurs, Rel. = 0. When a branch occurs, the program jumps to any byte in the range +129 to -127. A branch instruction requires a length of 2 bytes.

### ● Indexed (No Offset)

See Fig. 34. The indexed addressing mode allows access up to the lower 255th address of memory. In this mode, an instruction requires a length of one byte. The EA is the contents of the index register.

### ● Indexed (8-bit Offset)

See Fig. 35. The EA is the contents of the byte following the operation code, plus the contents of the index register. This mode allows access up to the lower 511th address of memory. Each instruction when used in the index addressing mode (8-bit offset) requires a length of 2 bytes.

### ● Indexed (16-bit Offset)

See Fig. 36. The contents of the 2 bytes following the operation code are added to content of the index register to compute the value of EA. In this mode, the complete memory can be accessed. When used in the indexed addressing mode (16-bit offset), an instruction must be 3 bytes long.

### ● Bit Set/Clear

See Fig. 37. This addressing mode is applied to the BSET and BCLR instructions that can set or clear any bit on page 0. The lower 3 bits of the operation code specify the bit to be set or cleared. The byte that follows the operation code indicates an address within page 0.

### ● Bit Test and Branch

See Fig. 38. This addressing mode is applied to the BRSET and BRCLR instructions that can test any bit within page 0 and can be branched in the relative addressing mode. The byte to be tested is addressed depending on the contents of the byte following the operation code. Individual bits within the byte to be tested are specified by the lower 3 bits of the operation code. The 3rd byte represents a relative value which will be added to the program counter when a branch condition is established. Each of these instructions should be 3 bytes long. The value of the test bit is written in the carry bit of the condition code register.

### ● Implied

See Fig. 39. This mode involves no EA. All information needed for execution of an instruction is contained in the operation code. Direct manipulation on the accumulator and index register is included in the implied addressing mode. Other instructions such as SWI and RTI are also used in this mode. All instructions used in the implied addressing mode should have a length of one byte.
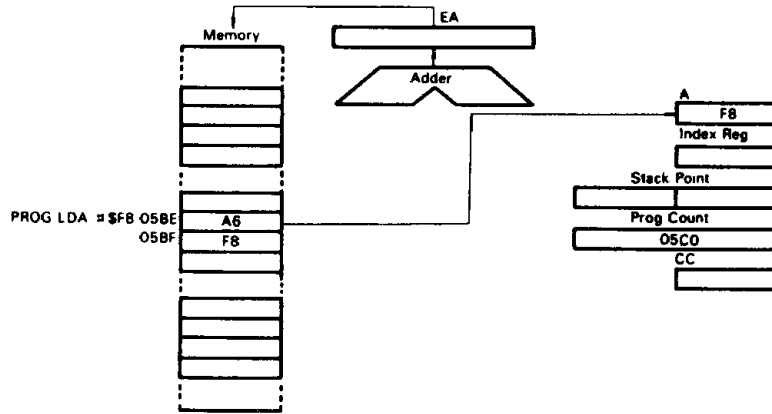
Figure 30   Example of Immediate Addressing



Figure 31   Example of Direct Addressing
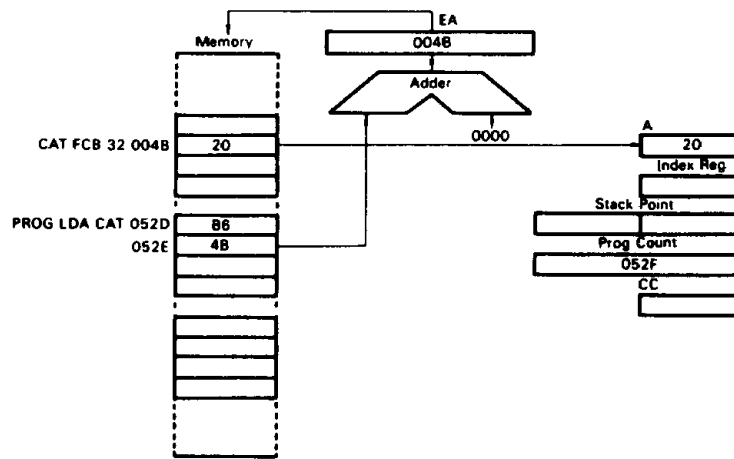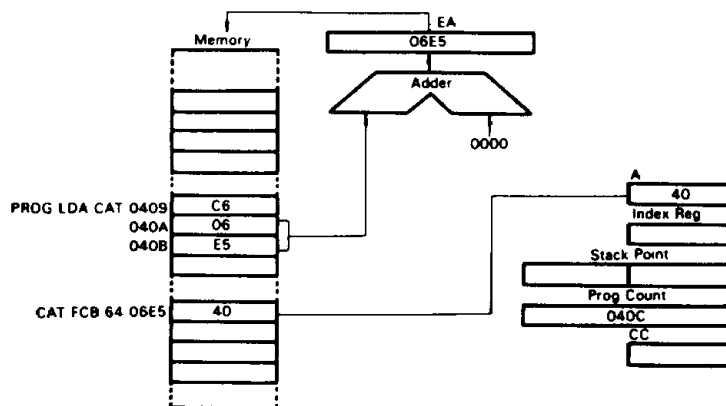


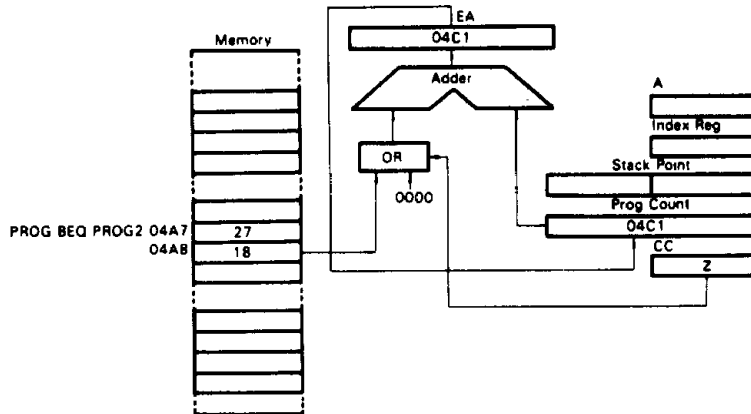Figure 32   Example of Extended Addressing

Figure 33   Example of Relative Addressing
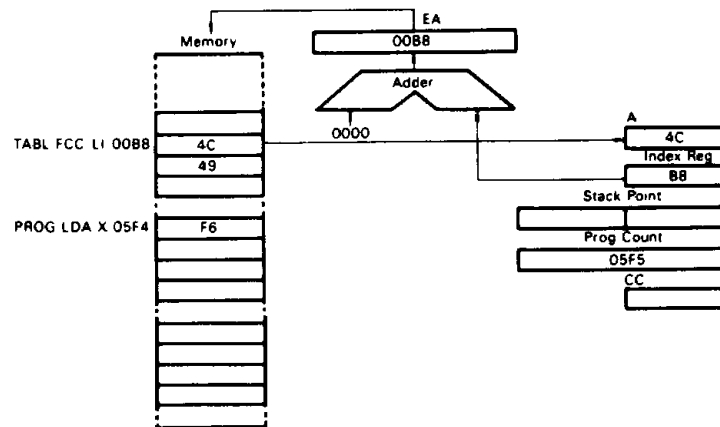


Figure 34   Example of Indexed (No Offset) Addressing
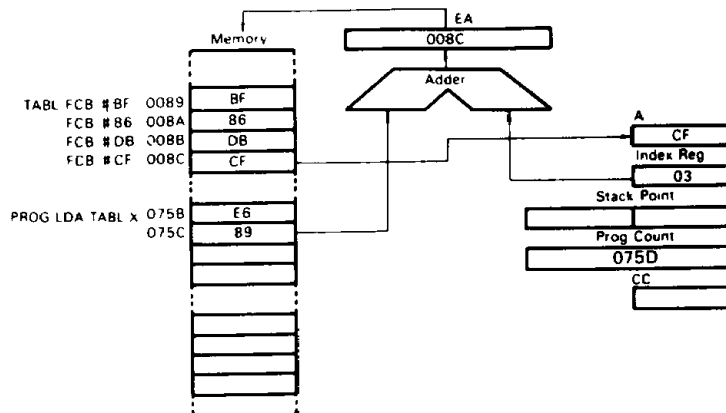


Figure 35   Example of Indexed (8-bit Offset) Addressing
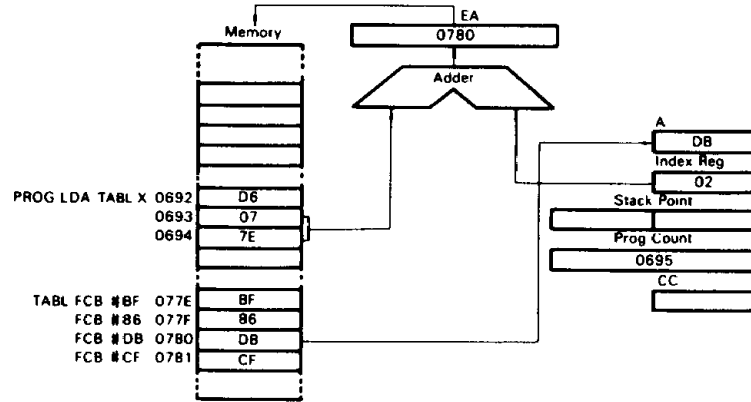
Figure 36   Example of Indexed (16-bit Offset) Addressing
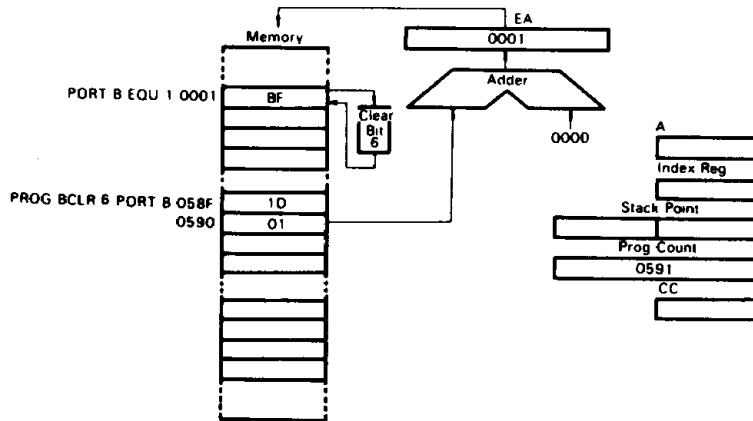


Figure 37   Example of Bit Set/Clear Addressing
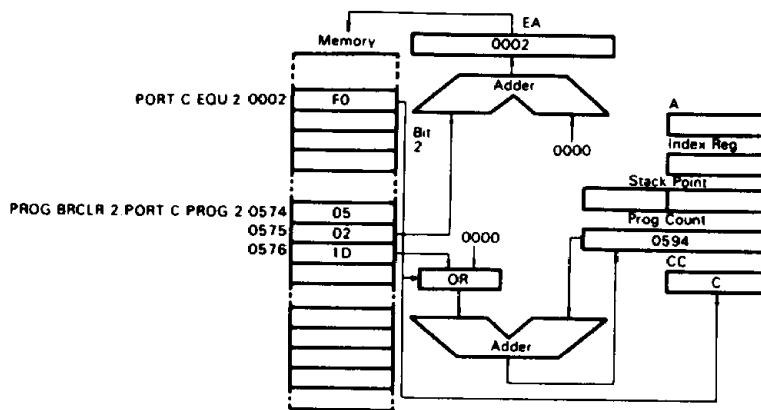


Figure 38   Example of Bit Test and Branch Addressing
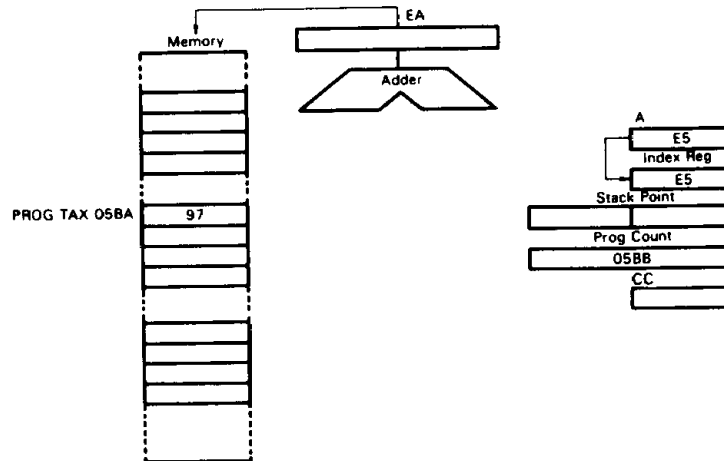
Figure 39   Example of Implied Addressing

# ■ INSTRUCTION SET

There are 62 basic instructions available to the HD63P05Y1 MCU. They can be classified into five categories: register/memory, read/modify/write, branch, bit manipulation, and control. The details of each instruction are described in Tables 5 through 11.

## ● Register/Memory Instructions

Most of these instructions use two operands. One operand is either an accumulator or index register. The other is derived from memory using one of the addressing modes used on the MCU. There is no register operand in the unconditional jump instruction (JMP) and the subroutine jump instruction (JSR). See Table 5.

## ● Read/Modify/Write Instructions

These instructions read a memory or register, then modify or test its contents, and write the modified value into the memory or register. Zero test instruction (TST) does not write data, and is handled as an exception in the read/modify/write group. See Table 6.

## ● Branch Instructions

A branch instruction branches from the program sequence in progress if a particular condition is established. See Table 7.

## ● Bit Manipulation Instructions

These instructions can be used with any bit located up to the lower 255th address of memory. Two groups are available; one for setting or clearing and the other for bit testing and branching. See Table 8.

## ● Control Instructions

The control instructions control the operation of the MCU which is executing a program. See Table 9.

## ● List of Instructions in Alphabetical Order

Table 10 lists all the instructions used on the MCU in the alphabetical order.

## ● Operation Code Map

Table 11 shows the operation code map for the instructions used on the MCU.
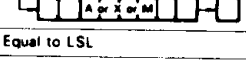
Table 5  Register/Memory Instructions

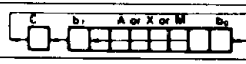| Operations | Mnemonic | Immediate OP | # | ~ | Direct OP | # | ~ | Extended OP | # | ~ | Indexed (No Offset) OP | # | ~ | Indexed (8-Bit Offset) OP | # | ~ | Indexed (16-Bit Offset) OP | # | ~ | Boolean/Arithmetic Operation | H | I | N | Z | C |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Load A from Memory | LDA | A6 | 2 | 2 | B6 | 2 | 3 | C6 | 3 | 4 | F6 | 1 | 3 | E6 | 2 | 4 | D6 | 3 | 5 | M→A | • | • | Λ | Λ | • |
| Load X from Memory | LDX | AE | 2 | 2 | BE | 2 | 3 | CE | 3 | 4 | FE | 1 | 3 | EE | 2 | 4 | DE | 3 | 5 | M→X | • | • | Λ | Λ | • |
| Store A in Memory | STA | | | | B7 | 2 | 3 | C7 | 3 | 4 | F7 | 1 | 4 | E7 | 2 | 4 | D7 | 3 | 5 | A→M | • | • | Λ | Λ | • |
| Store X in Memory | STX | | | | BF | 2 | 3 | CF | 3 | 4 | FF | 1 | 4 | EF | 2 | 4 | DF | 3 | 5 | X→M | • | • | Λ | Λ | • |
| Add Memory to A | ADD | AB | 2 | 2 | B8 | 2 | 3 | CB | 3 | 4 | FB | 1 | 3 | EB | 2 | 4 | DB | 3 | 5 | A+M→A | Λ | • | Λ | Λ | Λ |
| Add Memory and Carry to A | ADC | A9 | 2 | 2 | B9 | 2 | 3 | C9 | 3 | 4 | F9 | 1 | 3 | E9 | 2 | 4 | D9 | 3 | 5 | A+M+C→A | Λ | • | Λ | Λ | Λ |
| Subtract Memory | SUB | A0 | 2 | 2 | B0 | 2 | 3 | C0 | 3 | 4 | F0 | 1 | 3 | E0 | 2 | 4 | D0 | 3 | 5 | A-M→A | • | • | Λ | Λ | Λ |
| Subtract Memory from A with Borrow | SBC | A2 | 2 | 2 | B2 | 2 | 3 | C2 | 3 | 4 | F2 | 1 | 3 | E2 | 2 | 4 | D2 | 3 | 5 | A-M-C→A | • | • | Λ | Λ | Λ |
| AND Memory to A | AND | A4 | 2 | 2 | B4 | 2 | 3 | C4 | 3 | 4 | F4 | 1 | 3 | E4 | 2 | 4 | D4 | 3 | 5 | A·M→A | • | • | Λ | Λ | • |
| OR Memory with A | ORA | AA | 2 | 2 | BA | 2 | 3 | CA | 3 | 4 | FA | 1 | 3 | EA | 2 | 4 | DA | 3 | 5 | A+M→A | • | • | Λ | Λ | • |
| Exclusive OR Memory with A | EOR | A8 | 2 | 2 | B8 | 2 | 3 | C8 | 3 | 4 | F8 | 1 | 3 | E8 | 2 | 4 | D8 | 3 | 5 | A⊕M→A | • | • | Λ | Λ | • |
| Arithmetic Compare A with Memory | CMP | A1 | 2 | 2 | B1 | 2 | 3 | C1 | 3 | 4 | F1 | 1 | 3 | E1 | 2 | 4 | D1 | 3 | 5 | A-M | • | • | Λ | Λ | Λ |
| Arithmetic Compare X with Memory | CPX | A3 | 2 | 2 | B3 | 2 | 3 | C3 | 3 | 4 | F3 | 1 | 3 | E3 | 2 | 4 | D3 | 3 | 5 | X-M | • | • | Λ | Λ | Λ |
| Bit Test Memory with A (Logical Compare) | BIT | A5 | 2 | 2 | B5 | 2 | 3 | C5 | 3 | 4 | F5 | 1 | 3 | E5 | 2 | 4 | D5 | 3 | 5 | A·M | • | • | Λ | Λ | • |
| Jump Unconditional | JMP | | | | BC | 2 | 2 | CC | 3 | 3 | FC | 1 | 2 | EC | 2 | 3 | DC | 3 | 4 | | • | • | • | • | • |
| Jump to Subroutine | JSR | | | | BD | 2 | 5 | CD | 3 | 6 | FD | 1 | 5 | ED | 2 | 5 | DD | 3 | 6 | | • | • | • | • | • |

Symbols: Op = Operation
\# = Number of bytes
~ = Number of cycles

Table 6  Read/Modify/Write Instructions

| Operations | Mnemonic | Implied(A) OP | # | ~ | Implied(X) OP | # | ~ | Direct OP | # | ~ | Indexed (No Offset) OP | # | ~ | Indexed (8-Bit Offset) OP | # | ~ | Boolean/Arithmetic Operation | H | I | N | Z | C |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Increment | INC | 4C | 1 | 2 | 5C | 1 | 2 | 3C | 2 | 5 | 7C | 1 | 5 | 6C | 2 | 6 | A+1→A or X+1→X or M+1→M | • | • | Λ | Λ | • |
| Decrement | DEC | 4A | 1 | 2 | 5A | 1 | 2 | 3A | 2 | 5 | 7A | 1 | 5 | 6A | 2 | 6 | A-1→A or X-1→X or M-1→M | • | • | Λ | Λ | • |
| Clear | CLR | 4F | 1 | 2 | 5F | 1 | 2 | 3F | 2 | 5 | 7F | 1 | 5 | 6F | 2 | 6 | 00→A or 00→X or 00→M | • | • | 0 | 1 | • |
| Complement | COM | 43 | 1 | 2 | 53 | 1 | 2 | 33 | 2 | 5 | 73 | 1 | 5 | 63 | 2 | 6 | Ā→A or X̄→X or M̄→M | • | • | Λ | Λ | 1 |
| Negate (2's Complement) | NEG | 40 | 1 | 2 | 50 | 1 | 2 | 30 | 2 | 5 | 70 | 1 | 5 | 60 | 2 | 6 | 00-A→A or 00-X→X or 00-M→M | • | • | Λ | Λ | Λ |
| Rotate Left Thru Carry | ROL | 49 | 1 | 2 | 59 | 1 | 2 | 39 | 2 | 5 | 79 | 1 | 5 | 69 | 2 | 6 | (C ← A or X or M) | • | • | Λ | Λ | Λ |
| Rotate Right Thru Carry | ROR | 46 | 1 | 2 | 56 | 1 | 2 | 36 | 2 | 5 | 76 | 1 | 5 | 66 | 2 | 6 | (C → A or X or M) | • | • | Λ | Λ | Λ |
| Logical Shift Left | LSL | 48 | 1 | 2 | 58 | 1 | 2 | 38 | 2 | 5 | 78 | 1 | 5 | 68 | 2 | 6 | (C ← A or X or M ← 0) | • | • | Λ | Λ | Λ |
| Logical Shift Right | LSR | 44 | 1 | 2 | 54 | 1 | 2 | 34 | 2 | 5 | 74 | 1 | 5 | 64 | 2 | 6 | (0 → A or X or M → C) | • | • | 0 | Λ | Λ |
| Arithmetic Shift Right | ASR | 47 | 1 | 2 | 57 | 1 | 2 | 37 | 2 | 5 | 77 | 1 | 5 | 67 | 2 | 6 | (A or X or M → C) | • | • | Λ | Λ | Λ |
| Arithmetic Shift Left | ASL | 48 | 1 | 2 | 58 | 1 | 2 | 38 | 2 | 5 | 78 | 1 | 5 | 68 | 2 | 6 | Equal to LSL | • | • | Λ | Λ | Λ |
| Test for Negative or Zero | TST | 4D | 1 | 2 | 5D | 1 | 2 | 3D | 2 | 4 | 7D | 1 | 4 | 6D | 2 | 5 | A-00 or X-00 or M-00 | • | • | Λ | Λ | • |

Symbols: Op = Operation
\# = Number of bytes
~ = Number of cycles

Table 7  Branch Instructions

| Operations | Mnemonic | Addressing Modes Relative | | | Branch Test | Condition Code | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | OP | # | ~ | | H | I | N | Z | C |
| Branch Always | BRA | 20 | 2 | 3 | None | • | • | • | • | • |
| Branch Never | BRN | 21 | 2 | 3 | None | • | • | • | • | • |
| Branch IF Higher | BHI | 22 | 2 | 3 | $C+Z=0$ | • | • | • | • | • |
| Branch IF Lower or Same | BLS | 23 | 2 | 3 | $C+Z=1$ | • | • | • | • | • |
| Branch IF Carry Clear | BCC | 24 | 2 | 3 | $C=0$ | • | • | • | • | • |
| (Branch IF Higher or Same) | (BHS) | 24 | 2 | 3 | $C=0$ | • | • | • | • | • |
| Branch IF Carry Set | BCS | 25 | 2 | 3 | $C=1$ | • | • | • | • | • |
| (Branch IF Lower) | (BLO) | 25 | 2 | 3 | $C=1$ | • | • | • | • | • |
| Branch IF Not Equal | BNE | 26 | 2 | 3 | $Z=0$ | • | • | • | • | • |
| Branch IF Equal | BEQ | 27 | 2 | 3 | $Z=1$ | • | • | • | • | • |
| Branch IF Half Carry Clear | BHCC | 28 | 2 | 3 | $H=0$ | • | • | • | • | • |
| Branch IF Half Carry Set | BHCS | 29 | 2 | 3 | $H=1$ | • | • | • | • | • |
| Branch IF Plus | BPL | 2A | 2 | 3 | $N=0$ | • | • | • | • | • |
| Branch IF Minus | BMI | 2B | 2 | 3 | $N=1$ | • | • | • | • | • |
| Branch IF Interrupt Mask Bit is Clear | BMC | 2C | 2 | 3 | $I=0$ | • | • | • | • | • |
| Branch IF Interrupt Mask Bit is Set | BMS | 2D | 2 | 3 | $I=1$ | • | • | • | • | • |
| Branch IF Interrupt Line is Low | BIL | 2E | 2 | 3 | $INT=0$ | • | • | • | • | • |
| Branch IF Interrupt Line is High | BIH | 2F | 2 | 3 | $INT=1$ | • | • | • | • | • |
| Branch to Subroutine | BSR | AD | 2 | 5 | —— | • | • | • | • | • |

Symbols: Op = Operation
 # = Number of bytes
 ~ = Number of cycles

Table 8  Bit Manipulation Instructions

| Operations | Mnemonic | Addressing Modes Bit Set/Clear | | | Bit Test and Branch | | | Boolean/ Arithmetic Operation | Branch Test | Condition Code | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | OP | # | ~ | OP | # | ~ | | | H | I | N | Z | C |
| Branch IF Bit n is set | BRSET n(n=0···7) | — | — | — | 2·n | 3 | 5 | —— | $Mn=1$ | • | • | • | • | ∧ |
| Branch IF Bit n is clear | BRCLR n(n=0···7) | — | — | — | 01+2·n | 3 | 5 | —— | $Mn=0$ | • | • | • | • | ∧ |
| Set Bit n | BSET n(n=0···7) | 10+2·n | 2 | 5 | — | — | — | $1→Mn$ | —— | • | • | • | • | • |
| Clear Bit n | BCLR n(n=0···7) | 11+2·n | 2 | 5 | — | — | — | $0→Mn$ | —— | • | • | • | • | • |

Symbols: Op = Operation
 # = Number of bytes
 ~ = Number of cycles

Table 9  Control Instructions

| Operations | Mnemonic | Addressing Modes Implied OP | # | ~ | Boolean Operation | H | I | N | Z | C |
|---|---|---|---|---|---|---|---|---|---|---|
| Transfer A to X | TAX | 97 | 1 | 2 | A→X | ● | ● | ● | ● | ● |
| Transfer X to A | TXA | 9F | 1 | 2 | X→A | ● | ● | ● | ● | ● |
| Set Carry Bit | SEC | 99 | 1 | 1 | 1→C | ● | ● | ● | ● | 1 |
| Clear Carry Bit | CLC | 98 | 1 | 1 | 0→C | ● | ● | ● | ● | 0 |
| Set Interrupt Mask Bit | SEI | 9B | 1 | 2 | 1→I | ● | 1 | ● | ● | ● |
| Clear Interrupt Mask Bit | CLI | 9A | 1 | 2 | 0→I | ● | 0 | ● | ● | ● |
| Software Interrupt | SWI | 83 | 1 | 10 | | ● | 1 | ● | ● | ● |
| Return from Subroutine | RTS | 81 | 1 | 5 | | ● | ● | ● | ● | ● |
| Return from Interrupt | RTI | 80 | 1 | 8 | | ? | ? | ? | ? | ? |
| Reset Stack Pointer | RSP | 9C | 1 | 2 | $FF→SP | ● | ● | ● | ● | ● |
| No-Operation | NOP | 9D | 1 | 1 | Advance Prog. Cntr. Only | ● | ● | ● | ● | ● |
| Decimal Adjust A | DAA | 8D | 1 | 2 | Converts binary add of BCD charcters into BCD format | ● | ● | ∧ | ∧ | ∧* |
| Stop | STOP | 8E | 1 | 4 | | ● | ● | ● | ● | ● |
| Wait | WAIT | 8F | 1 | 4 | | ● | ● | ● | ● | ● |

Symbols: Op = Operation
      # = Number of bytes
      ~ = Number of cycles

* Are BCD characters of upper byte 10 or more? (They are not cleared if set in advance.)

Table 10  Instruction Set (in Alphabetical Order)

| Mnemonic | Implied | Immediate | Direct | Extended | Relative | Indexed (No Offset) | Indexed (8-Bit) | Indexed (16-Bit) | Bit Set/ Clear | Bit Test & Branch | H | I | N | Z | C |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADC | | × | × | × | | × | × | × | | | ∧ | ● | ∧ | ∧ | ∧ |
| ADD | | × | × | × | | × | × | × | | | ∧ | ● | ∧ | ∧ | ∧ |
| AND | | × | × | × | | × | × | × | | | ● | ● | ∧ | ∧ | ● |
| ASL | × | | × | | | × | × | | | | ● | ● | ∧ | ∧ | ∧ |
| ASR | × | | × | | | × | × | | | | ● | ● | ∧ | ∧ | ∧ |
| BCC | | | | | × | | | | | | ● | ● | ● | ● | ● |
| BCLR | | | | | | | | | × | | ● | ● | ● | ● | ● |
| BCS | | | | | × | | | | | | ● | ● | ● | ● | ● |
| BEQ | | | | | × | | | | | | ● | ● | ● | ● | ● |
| BHCC | | | | | × | | | | | | ● | ● | ● | ● | ● |
| BHCS | | | | | × | | | | | | ● | ● | ● | ● | ● |
| BHI | | | | | × | | | | | | ● | ● | ● | ● | ● |
| (BHS) | | | | | × | | | | | | ● | ● | ● | ● | ● |
| BIH | | | | | × | | | | | | ● | ● | ● | ● | ● |
| BIL | | | | | × | | | | | | ● | ● | ● | ● | ● |
| BIT | | × | × | × | | × | × | × | | | ● | ● | ∧ | ∧ | ● |
| (BLO) | | | | | × | | | | | | ● | ● | ● | ● | ● |
| BLS | | | | | × | | | | | | ● | ● | ● | ● | ● |
| BMC | | | | | × | | | | | | ● | ● | ● | ● | ● |
| BMI | | | | | × | | | | | | ● | ● | ● | ● | ● |
| BMS | | | | | × | | | | | | ● | ● | ● | ● | ● |
| BNE | | | | | × | | | | | | ● | ● | ● | ● | ● |
| BPL | | | | | × | | | | | | ● | ● | ● | ● | ● |
| BRA | | | | | × | | | | | | ● | ● | ● | ● | ● |

(to be continued)

Condition Code Symbols:
- H    Half Carry (From Bit 3)
- I    Interrupt Mask
- N    Negative (Sign Bit)
- Z    Zero
- C    Carry/Borrow
- ∧    Test and Set if True, Cleared Otherwise
- ●    Not Affected
- ?    Load CC Register From Stack

Table 10  Instruction Set (in Alphabetical Order)

| Mnemonic | Implied | Immediate | Direct | Extended | Relative | Indexed (No Offset) | Indexed (8-Bit) | Indexed (16-Bit) | Bit Set/ Clear | Bit Test & Branch | H | I | N | Z | C |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BRN | | | | | X | | | | | | ● | ● | ● | ● | ● |
| BRCLR | | | | | | | | | | X | ● | ● | ● | ● | ∧ |
| BRSET | | | | | | | | | | X | ● | ● | ● | ● | ∧ |
| BSET | | | | | | | | | X | | ● | ● | ● | ● | ● |
| BSR | | | | | X | | | | | | ● | ● | ● | ● | ● |
| CLC | X | | | | | | | | | | ● | ● | ● | ● | 0 |
| CLI | X | | | | | | | | | | ● | 0 | ● | ● | ● |
| CLR | X | | X | | | X | X | | | | ● | ● | 0 | 1 | ● |
| CMP | | X | X | X | | X | X | X | | | ● | ● | ∧ | ∧ | ∧ |
| COM | X | | X | | | X | X | | | | ● | ● | ∧ | ∧ | 1 |
| CPX | | X | X | X | | X | X | X | | | ● | ● | ∧ | ∧ | ∧ |
| DAA | X | | | | | | | | | | ● | ● | ∧ | ∧ | ∧ |
| DEC | X | | X | | | X | X | | | | ● | ● | ∧ | ∧ | ● |
| EOR | | X | X | X | | X | X | X | | | ● | ● | ∧ | ∧ | ● |
| INC | X | | X | | | X | X | | | | ● | ● | ∧ | ∧ | ● |
| JMP | | | X | X | | X | X | X | | | ● | ● | ● | ● | ● |
| JSR | | | X | X | | X | X | X | | | ● | ● | ● | ● | ● |
| LDA | | X | X | X | | X | X | X | | | ● | ● | ∧ | ∧ | ● |
| LDX | | X | X | X | | X | X | X | | | ● | ● | ∧ | ∧ | ● |
| LSL | X | | X | | | X | X | | | | ● | ● | ∧ | ∧ | ∧ |
| LSR | X | | X | | | X | X | | | | ● | ● | 0 | ∧ | ∧ |
| NEG | X | | X | | | X | X | | | | ● | ● | ∧ | ∧ | ∧ |
| NOP | X | | | | | | | | | | ● | ● | ● | ● | ● |
| ORA | | X | X | X | | X | X | X | | | ● | ● | ∧ | ∧ | ● |
| ROL | X | | X | | | X | X | | | | ● | ● | ∧ | ∧ | ∧ |
| ROR | X | | X | | | X | X | | | | ● | ● | ∧ | ∧ | ∧ |
| RSP | X | | | | | | | | | | ● | ● | ● | ● | ● |
| RTI | X | | | | | | | | | | ? | ? | ? | ? | ? |
| RTS | X | | | | | | | | | | ● | ● | ● | ● | ● |
| SBC | | X | X | X | | X | X | X | | | ● | ● | ∧ | ∧ | ∧ |
| SEC | X | | | | | | | | | | ● | ● | ● | ● | 1 |
| SEI | X | | | | | | | | | | ● | 1 | ● | ● | ● |
| STA | | | X | X | | X | X | X | | | ● | ● | ∧ | ∧ | ● |
| STOP | X | | | | | | | | | | ● | ● | ● | ● | ● |
| STX | | | X | X | | X | X | X | | | ● | ● | ∧ | ∧ | ● |
| SUB | | X | X | X | | X | X | X | | | ● | ● | ∧ | ∧ | ∧ |
| SWI | X | | | | | | | | | | ● | 1 | ● | ● | ● |
| TAX | X | | | | | | | | | | ● | ● | ● | ● | ● |
| TST | X | | X | | | X | X | | | | ● | ● | ∧ | ∧ | ● |
| TXA | X | | | | | | | | | | ● | ● | ● | ● | ● |
| WAIT | X | | | | | | | | | | ● | ● | ● | ● | ● |

Condition Code Symbols:
H   Half Carry (From Bit 3)     C   Carry/Borrow
I   Interrupt Mask              ∧   Test and Set if True, Cleared Otherwise
N   Negative (Sign Bit)         ●   Not Affected
Z   Zero                        ?   Load CC Register From Stack

# HD63P05Y1, HD63PA05Y1, HD63PB05Y1

Table 11 Operation Code Map

| | Bit Manipulation — Test & Branch (0) | Set/ Clear (1) | Branch — Rel (2) | DIR (3) | A (4) | X (5) | ,X1 (6) | ,XO (7) | Control — IMP (8) | IMP (9) | IMM (A) | DIR (B) | EXT (C) | ,X2 (D) | ,X1 (E) | ,XO (F) | HIGH |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | BRSETO | BSETO | BRA | NEG | | | | | RTI* | | SUB | | | | | | 0 |
| 1 | BRCLRO | BCLRO | BRN | | | | | | RTS* | | CMP | | | | | | 1 |
| 2 | BRSET1 | BSET1 | BHI | | | | | | | | SBC | | | | | | 2 |
| 3 | BRCLR1 | BCLR1 | BLS | COM | | | | | SWI* | | CPX | | | | | | 3 |
| 4 | BRSET2 | BSET2 | BCC | LSR | | | | | | | AND | | | | | | 4 |
| 5 | BRCLR2 | BCLR2 | BCS | | | | | | | | BIT | | | | | | 5 |
| 6 | BRSET3 | BSET3 | BNE | ROR | | | | | | | LDA | | | | | | 6 |
| 7 | BRCLR3 | BCLR3 | BEQ | ASR | | | | | | TAX* | STA | | | | | STA(+1) | 7 |
| 8 | BRSET4 | BSET4 | BHCC | LSL/ASL | | | | | | CLC | EOR | | | | | | 8 |
| 9 | BRCLR4 | BCLR4 | BHCS | ROL | | | | | | SEC | ADC | | | | | | 9 |
| A | BRSET5 | BSET5 | BPL | DEC | | | | | | CLI* | ORA | | | | | | A |
| B | BRCLR5 | BCLR5 | BMI | | | | | | | SEI* | ADD | | | | | | B |
| C | BRSET6 | BSET6 | BMC | INC | | | | | | RSP* | JMP(−1) | | | | | | C |
| D | BRCLR6 | BCLR6 | BMS | TST(−1) | TST | | TST(−1) | | DAA* | NOP | BSR* | JSR(+2) | | JSR(+1) | | JSR(+2) | D |
| E | BRSET7 | BSET7 | BIL | | | | | | STOP* | − | LDX | | | | | | E |
| F | BRCLR7 | BCLR7 | BIH | CLR | | | | | WAIT* | TXA* | STX | | | | | STX(+1) | F |
| | 3/5 | 2/5 | 2/3 | 2/5 | 1/2 | 1/2 | 2/6 | 1/5 | 1/* | 1/1 | 2/2 | 2/3 | 3/4 | 3/5 | 2/4 | 1/3 | LOW |

(NOTES)
1. "−" is an undefined operation code.
2. The lowermost numbers in each column represent a byte count and the number of cycles required (byte count/number of cycles). The number of cycles for the mnemonics asterisked (*) is as follows:

| | | | |
|---|---|---|---|
| RTI | 8 | TAX | 2 |
| RTS | 5 | RSP | 2 |
| SWI | 10 | TXA | 2 |
| DAA | 2 | BSR | 5 |
| STOP | 4 | CLI | 2 |
| WAIT | 4 | SEI | 2 |

3. The parenthesized numbers must be added to the cycle count of the particular instruction.

● **Additional Instructions**

The following new instructions are used on the HD63P05Y1:

**DAA** Converts the contents of the accumulator into BCD code.

**WAIT** Causes the MCU to enter the wait mode. For this mode, see the topic, Wait Mode.

**STOP** Causes the MCU to enter the stop mode. For this mode, see the topic, Stop Mode.

■ **PRECAUTION 1—BOARD DESIGN OF OSCILLATION CIRCUIT**

When connecting crystal and ceramic resonator with the XTAL and EXTAL pins to oscillate, observe the followings in designing the board.

(1) Locate crystal, ceramic resonator, and load capacity $C_1$ and $C_2$ as near the LSI as possible. (Induction of noise from outside to the XTAL and EXTAL pins may cause trouble in oscillation.)

(2) Wire the signal lines to the neighboring XTAL and EXTAL pins as far apart as possible.

(3) Board design of situating signal lines or power supply lines near the oscillator circuit as shown as Figure 41, should not be used because of trouble in oscillation by induction. The resistor between the XTAL and EXTAL, and pins close to them should be 10MΩ or more. The circuit in Fig. 40 is an example of good board design.

■ **PRECAUTION 2—PROGRAM OF WRITE ONLY REGISTER**

Read/Modify/Write instructions are unavailable for changing the contents of Write Only Register (e.g. DDR; Data Direction Register of I/O port) of HD6305X, HD6305Y and HD63P05Y.

(1) Data cannot be read from Write Only Register. (e.g. DDR of I/O port

While Read/Modify/Write instructions are executed in the following sequence.

(i) Reads the contents from appointed address.
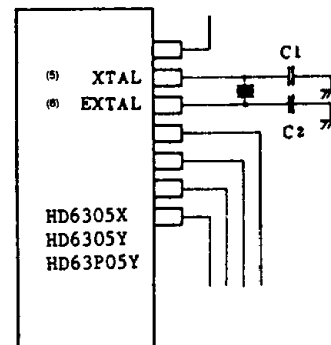(ii) Changes the data which has been read.



Figure 40 Design of Oscillation Circuit Board

(iii) Turn the data back to the original address.
Thus, Read/Modify/Write instructions cannot be applied to Write Only Register such as DOR.
(2) For the same reason, do not set DOR of I/O port using BSET and BCLR instructions.
(3) Stored instructions (e.g. STA and STX, etc.) are available for writing into the Write Only Register.

## ■ PRECAUTION 3—SENDING/RECEIVING PROGRAM OF SERIAL DATA

Be careful that malfunction may occur if SDR (SERIAL DATA REGISTER: $0012) is read or written during transmitting or receiving serial data.

## ■ PRECAUTION 4—WAIT/STOP INSTRUCTIONS PROGRAM

When I bit of condition code register is "1" and interrupt ($\overline{INT}$, TIMER/$\overline{INT}_2$, SCI/TIMER 2) is held, the MCU does not enter into WAIT mode by executing WAIT instruction.

In that case, after the 4 dummy cycles, the MCU executes the next instruction.

In the same way, when external interrupts ($\overline{INT}$, $\overline{INT}_2$) are held at the bit I set, the MCU does not enter into the STOP mode by executing STOP instruction. In that case the MCU executes the best instruction after the 4 dummy cycles.
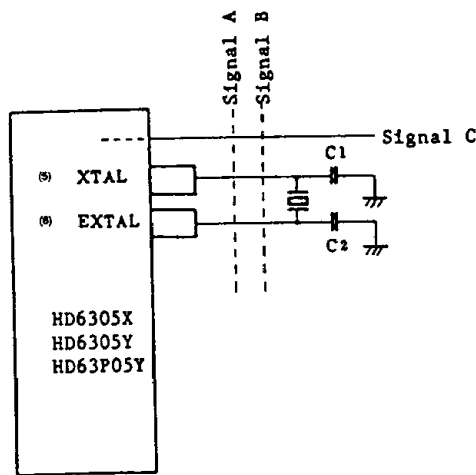


Figure 41  Example of Circuit Causing Trouble in Oscillation

## ■ PRECAUTION WHEN USING BIL/BIH INSTRUCTION

(1) Execute Instruction after the $\overline{INT}$ Voltage level has stabilized above $V_{IH}$ or below $V_{IL}$.
(2) $\overline{INT}$ voltage level needs to be stabilized while BIL/BIH Instruction Execution.

There may be a malfunction by glitch on control signal if BIL/BIH Instruction Execution has exercized in unstablized $\overline{INT}$ signal level.



Avoid BIL/BIH Instruction Execution
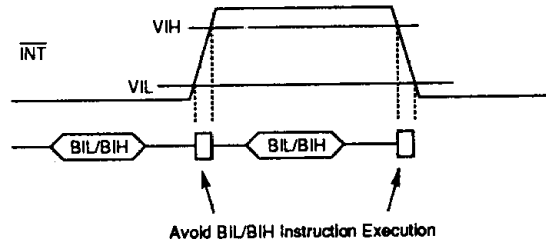
## ■ PRECAUTION TO USE BSR

If there is 2nd BSR programmed on the address which is directed by first BSR, 2nd BSR may not be executed correctly. For this reason, BSR should not be programmed on the address which is directed by first BSR.
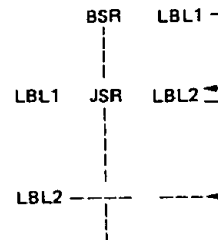
If necessary, please program as following.
(1) On the address which first BSR directed, NOP instruction should be inserted before second BSR.
(2) On the address which first BSR directed, JSR instruction should be programmed instead of 2nd BSR.
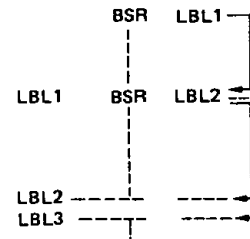


example of malfunction
of 2nd BSR execution
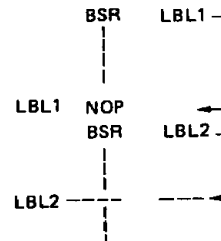


example of counter measure
(NOP is inserted)



example of counter measure
(JSR is used instead of BSR)