HD63705V0, HD637A05V0, HD637B05V0

(Limiting Supplies. For Development Only.)

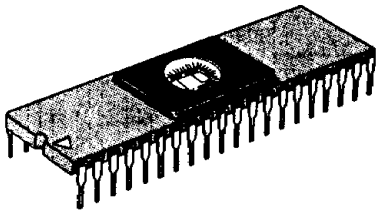
The HD63705V0 is an 8-bit CMOS single chip microcomputer unit (MCU) which includes 4k bytes of EPROM and is object code compatible with the HD6305V0.

This MCU contains a CPU, clock oscillator, 4k bytes of EPROM, 192 bytes of RAM, 31 parallel I/O pins, two timers and a serial communication interface (SCI).

In addition to CMOS technology, it provides power saving STOP, WAIT and STANDBY modes, further reducing its already low power consumption.

The HD63705V0 is available in a hermetically sealed 40-pin ceramic package with a glass window. This glass window allows for programming and EPROM erasure in the same sway as 27256 type EPROM.

HD63705V0C, HD637A05V0C
HD637B05V0C



(DC-40)

■ HARDWARE FEATURES

- Pin compatible with HD6305V0 and HD6305UO
- 4096 bytes of EPROM
- 192 bytes of RAM
- 31 I/O terminals
- Two Timers
  - — a 8-bit timer with a 7-bit prescaler (software programmable prescaler; event counter)
  - — a 15-bit timer (capable of being used as 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 mode
  - — Wait mode ..... Internal oscillator remains active. The CPU processing is halted. The timer, the SCI, and interrupts can function normally. The contents of all registers remain unchanged, except that I bit of the condition code register is cleared.
  - — Stop mode ..... Internal oscillator is halted. RAM, I/O terminals and all registers except I bit of the condition code register, bits 6 and 7 of the timer control register, and bits 4, 5, 6 and 7 of the SCI status register remain unchanged.
  - — Standby mode .... Internal oscillator is halted. The data of RAM is unchanged. The MCU internal registers are reset.
- Minimum instruction cycle time

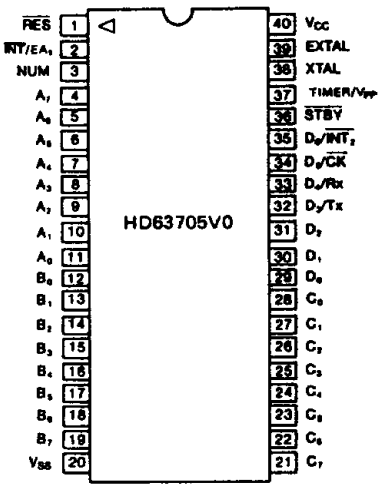
HD63705V0	1μs (f = 1MHz)
HD637A05V0	0.67μs (f = 1.5MHz)
HD637B05V0	0.5μs (f = 2MHz)

Wide operating range

HD63705V0	f = 0.1 ~ 1MHz (V_{CC} = 5V ± 10%)
HD637A05V0	f = 0.1 ~ 1MHz (V_{CC} = 1.5V ± 10%)
HD637B05V0	f = 0.1 ~ 2MHz (V_{CC} = 5V ± 10%)

- Operating modes .... MCU mode
  EPROM mode

■ PIN ARRANGEMENT



(Top View)

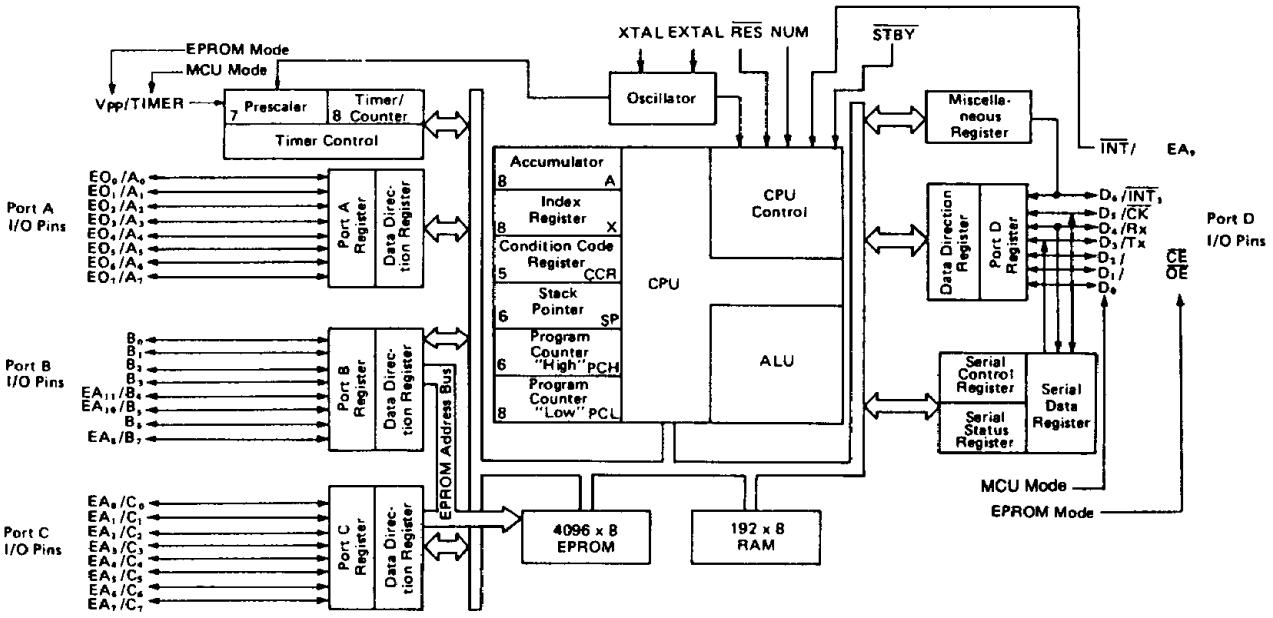
■ SOFTWARE FEATURES

- Similar to the HD6800
- Byte efficient instruction set
- Powerful bit manipulation instructions (Bit set, bit clear and bit test and branch are available for all RAM bits and all I/O terminals)
- Versatile interrupt handling
- Powerful indexed addressing for tables
- Full set of conditional branches
- 10 powerful addressing modes
- All addressing modes apply to ROM, RAM and I/O instructions
- 3 new instructions, STOP, WAIT and DDA, added to the HD6805 family instruction set
- Instruction set which is compatible with that of the HD6305V

■ PROGRAM DEVELOPMENT SUPPORT TOOLS

- Cross assembler software for use with IBM PC and compatibles
- In circuit emulator for use with IBM PCs and compatibles
- Programming socket adapter for programming the EPROM-on-chip device.

■ BLOCK DIAGRAM



(Precaution in Using a Single-Chip Microcomputer with EPROM)

(1) If the MCU is exposed to strong light especially a fluorescent light or the sunlight, EPROM data may be erased or the MCU malfunctions is caused by photocurrent. Therefore, after programming, it is recommended to put a shielding label on the glass window in the applications which expose the LSI to ambient light.

(2) Be careful not to rub the glass window with plastic or other materials that may generate electricity leading to malfunction of the LSI. Use conductive shielding labels that can distribute charge evenly.

■ ABSOLUTE MAXIMUM RATING

Item	Symbol	Value	Unit	Note
Supply Voltage	V_{CC}	−0.3 ~ +7.0	V	
Programming Voltage	V_{PP}	−0.3 ~ +15.0	V	1
Input Voltage	V_{in}	−0.3 ~ V_{CC} +0.3	V	2
Operating Voltage	T_{opr}	0 ~ +70	°C	
Storage Voltage	T_{str}	−55 ~ +125	°C	

(NOTE 1) Applies to TIMER/Vpp pin.

(NOTE 2) Applies to all other pins except for TIMER/Vpp.

(Precaution)

These devices contain circuits to protect the inputs against high static voltages or high electric fields. Be careful not to apply any voltage higher than the absolute maximum rating to these high input impedance circuits. For normal operation, we recommend the V_{in} and V_{out} be constrained to the range $V_{SS} \leqq (V_{in}$ or $V_{out}) \leqq V_{CC}$.

■ MCU MODE ELECTRICAL CHARACTERISTICS

• DC Characteristics (V_{CC} = 5.0V ± 10%, V_{SS} = GND, Ta = 0~+70°C, TIMER/V_{PP} = GND ~ V_{CC}, unless otherwise noted)

Item		Symbol	Test Condition	min	typ	max	unit
Input "High" Voltage	$\overline{RES}$, $\overline{STBY}$	V_{IH}		V_{CC}−0.5	–	V_{CC}+0.3	V
	EXTAL			V_{CC}x0.7	–	V_{CC}+0.3	
	All Others			2.0	–	V_{CC}+0.3	
Input "Low" Voltage	All Inputs	V_{IL}		−0.3	–	0.8	V
Input Leakage Current	TIMER/V_{PP}	$\lvert I_{IL} \rvert$	V_{in} = 0.5 ~ V_{CC}−0.5V	–	1	100	μA
	$\overline{INT}$, $\overline{STBY}$			–	–	1.0	
Three State Current	$A_0 \sim A_7$, $B_0 \sim B_7$, $C_0 \sim C_7$, $D_0 \sim D_6$	$\lvert I_{TSI} \rvert$		–	–	1.0	
Current Dissipation***	Operating	I_{CC}	f = 1MHz*	–	5	10	mA
	Wait			–	2	5	mA
	Stop			–	2	10	μA
	Standby			–	2	10	μA
Input capacitance	TIMER/V_{PP}	C_{in}	f = 1MHz, V_{in} = 0V	–	–	100	pF
	All Terminals except TIMER/V_{PP}			–	–	15	pF

*The value at f = XMHz is given by I_{CC} = (f = XMHz) = I_{CC} (f = 1 MHz) XX

**At Standby Mode

***V_{IH} min. = V_{CC}−1.0V, V_{IL} max. = 0.8V, Penetrate current is not included.

• AC Characteristics (V_{CC} = 5.0V ± 10%, V_{SS} = GND, Ta = 0 ~ +70°C, unless otherwise noted)

Item	Symbol	Test Condition	HD63705V0			HD637A05V0			HD637B05V0			Unit
			min	typ	max	min	typ	max	min	typ	max	
Clock Frequency	f_{cl}		0.4	–	4	0.4	–	6	0.4	–	8	MHz
Cycle Time	t_{cyc}		1.0	–	10	0.666	–	10	0.5	–	10	μs
$\overline{INT}$ 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}
TIMER Pulse Width	t_{TWL}		t_{cyc} +250	–	–	t_{cyc} +200	–	–	t_{cyc} +200	–	–	ns
Oscillation Start Time (Crystal)	t_{osc}	CL=22pF 20% Rs = 60Ω max	–	–	20	–	–	20	–	–	20	ms
Reset Delay Time	t_{RHL}	external capacitance 2.2 μF	80	–	–	80	–	–	80	–	–	ms

- **Port Characteristics (V_{CC} = 5.0V ± 10%, V_{SS} = GND, Ta = 0 ~ +70, unless otherwise noted)**

Item		Symbol	Test Condition	min	typ	max	Unit
Output "High" Voltage	Port A, B, C, D	V_{OH}	I_{OH} = –200μA	2.4	–	–	V
			I_{OH} = –10μA	V_{CC} –0.7	–	–	V
Output "Low" Voltage		V_{OL}	I_{OL} = 1.6mA	–	–	0.55	V
Input "High" Voltage	Port A, B, C, D	V_{IH}		2.0	–	V_{CC}+0.3	V
Input "Low" Voltage		V_{IL}		–0.3	–	0.8	V
Input Leakage Current		$\|I_{IL}\|$	V_{in} = 0.5 ~ V_{CC} –0.5V	–	–	1	μA

- **SCI Timing (V_{CC} = 5.0V ± 10%, V_{SS} = GND, Ta = 0 ~ +70°C, unless otherwise noted)**

Item	Symbol	Test Condition	HD63705V0			HD637A05V0			HD637B05V0			Unit
			min	typ	max	min	typ	max	min	typ	max	
Clock Cycle Time	t_{Scyc}	Fig. 1	1	–	32768	0.67	–	21845	0.5	–	16384	μs
Data Output Delay Time	t_{TXD}		–	–	250	–	–	250	–	–	250	ns
Data Set-up Time	t_{SRX}	Fig. 2	200	–	–	200	–	–	200	–	–	ns
Data Hold Time	t_{HRX}		100	–	–	100	–	–	100	–	–	ns

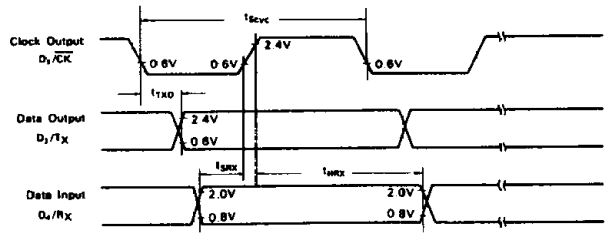


Figure 1 SCI Timing (Internal Clock)

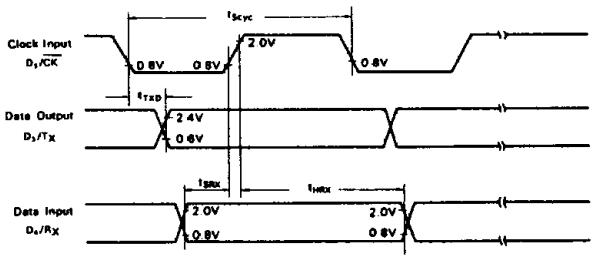


Figure 2 SCI Timing (External Clock)

■ EPROM MODE ELECTRICAL CHARACTERISTICS

■ PROGRAM OPERATION

● DC Characteristics (V_{CC} = 6V ± 0.25V, V_{PP} = 12.5V ± 0.3V, Ta = 25°C ± 5°C)

Item	Symbol	Test Condition	min	typ	max	Unit
Input Leakage Current	I_{LI}	V_{IN} = 6.25V/0.45V	–	–	2	μA
Output Voltage	V_{OL}	I_{OL} = 2.1mA	–	–	0.45	V
	V_{OH}	I_{OH} = –400μA	2.4	–	–	V
Power Supply Current (Active)	I_{CC}		–	–	30	mA
Input Voltage	V_{IL}		–0.1	–	0.8	V
	V_{IH}		2.2	–	V_{CC}+0.3	V
Programming Current	I_{PP}	$\overline{CE}$ = V_{IL}	–	–	40	mA

(Note) 1. V_{PP} (+12.5V) must be applied after V_{CC} (6V) is settled and must be removed before V_{CC}.
2. V_{PP} must not exceed + 15V. Be careful to prevent overshoot of the V_{PP} when switching to 12.5V.
3. The device must not be inserted into a board with V_{PP} at 12.5V to prevent damage to the reliability of the device.
4. When $\overline{CE}$=V_{IL}, V_{PP} must not be changed from V_{CC} ~ 12.5V or from 12.5V ~ V_{CC}.

● AC Characteristics (V_{CC} = 6V ± 0.25V, V_{PP} = 12.5V ± 0.3V, Ta = 25°C ± 5°C)

Item	Symbol	Test Condition	min	typ	max	Unit
Address Set-up Time	t_{AS}		2	–	–	μs
$\overline{OE}$ Set-up Time	t_{OES}		2	–	–	μs
Data Set-up Time	t_{DS}		2	–	–	μs
Address Hold Time	t_{AH}		0	–	–	μs
Data Hold Time	T_{DH}		2	–	–	μs
Output Disable Delay Time	t_{DF}*		0	–	130	ns
V_{PP} Set-up Time	t_{VPS}		2	–	–	μs
Program Pulse Width	t_{PW}		0.95	1.0	1.05	ms
V_{CC} Set-up Time	t_{VCS}		2	–	–	μs
$\overline{OE}$ Output Delay Time	t_{OE}		0	–	500	ns
Overprogramming $\overline{CE}$ Pulse Width	t_{OPW}		2.85	–	78.75	ms

* t_{DF} is defined when any lines are not connected to output and the output level can not be referred.

- **Switching Characteristics**

Test Condition
Input pulse level . . . . . 0.8V ~ 2.2V

Input rise/fall time . . . . . ≦ 20 ns
I/O timing reference level . . . . . input 1V, 2V
output 0.8V, 2V

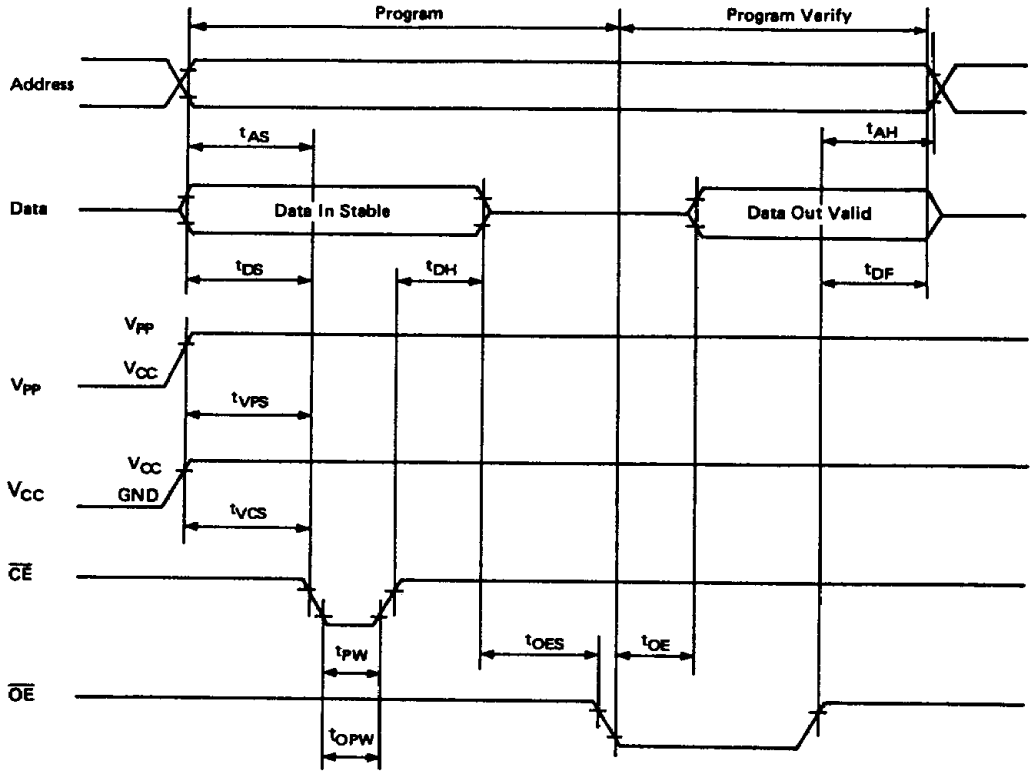


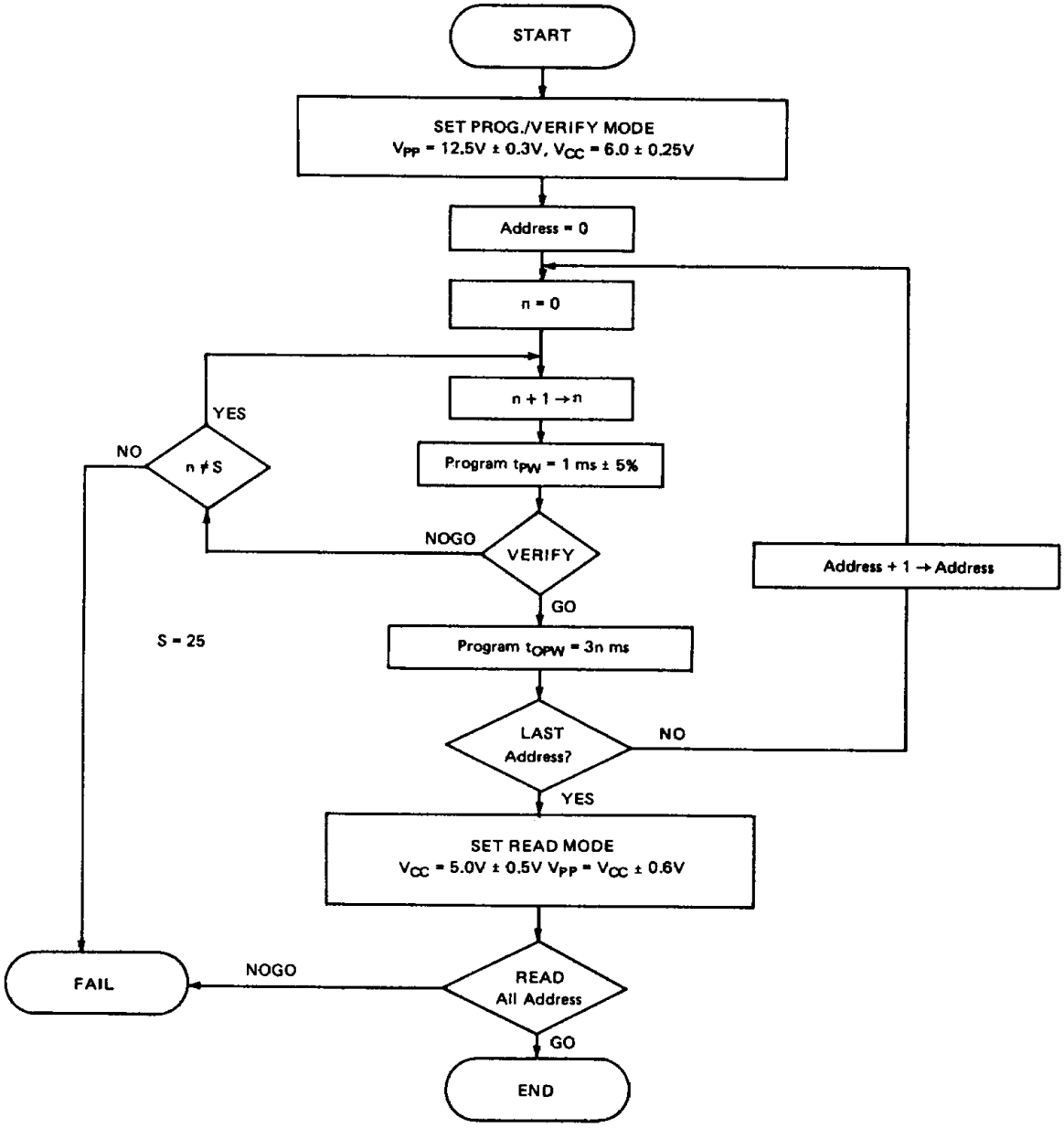
Figure 3 EPROM Program/Verify Timing

HITACHI
Hitachi America, Ltd. • Hitachi Plaza • 2000 Sierra Point Pkwy. • Brisbane, CA 94005-1819 • (415) 589-8300

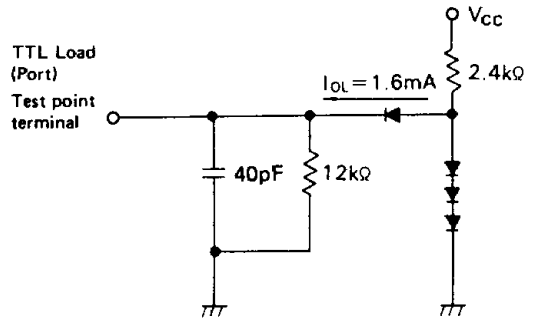
Figure 4
Not Applicable
to this
Product Line.

■ HIGH-SPEED PROGRAMMING

The HD63705V0 is applied to the high-speed programming method shown in the following flowchart. This method realizes faster programming without any voltage stress to the device nor deterioration in reliability of programmed data.



High-speed Programming Flowchart



(NOTE) 1. The load capacitance includes stray capacitahce caused by the probe, jig, etc.
2. All diodes are 1S2074Ⓗ.

Figure 5 Test Load

■ FUNCTIONAL PIN DESCRIPTION

The following paragraphs provide a brief description of HD63705V0 MCU input and output signals.

- **V_{CC}, V_{SS}**

These are the power supply inputs. V_{CC} = 5.0V ± 10%, V_{SS} = 0V (ground).

- **$\overline{INT}/EA_9$, $\overline{INT_2}$**

The MCU receives an external interrupt through these terminals. For details, see "INTERRUPT". The $\overline{INT_2}$ is used as the Port D_6 pin. In the EPROM mode, the $\overline{INT}$ is used as input of EA_9.

- **XTAL, EXTAL**

These pins are inputs of the internal oscillator circuit. A crystal (AT cut, 2.0 ~ 8.0 MHz) or ceramic resonator is connected to these pins. Refer to "INTERNAL OSCILLATOR" for using these inputs.

- **TIMER/V_{PP}**

This is an external input to control the internal timer circuit. See "TIMER" for details of the timer circuit.

In the EPROM mode, this pin is used when programming EPROM. The programming voltage V_{PP} is applied to this pin.

Once the voltage of 12.5V ± 0.3V is applied to this pin and $\overline{CE}$ and $\overline{OE}$ are set low and high respectively, data are written into EPROM through Port A (EO_0 ~ EO_7). EPROM addresses are input through the Port C (EA_0 ~ EA_7), Port B (EA_8, EA_{10} ~ EA_{11}) and $\overline{INT}$ (EA_9).

- **$\overline{RES}$**

This is used to reset the MCU. For details, see "RESET".

- **NUM**

This is not for user applications. Connect this pin to the V_{SS} in the MCU mode and to the V_{CC} in the EPROM mode.

- **Input/output pins (A_0 ~ A_7, B_0 ~ B_7, C_0 ~ C_7, D_0 ~ D_6)**

These 31 pins consist of three 8-bit I/O ports (A, B, C) and a 7-bit I/O port (D). Any pin may be programmed as an input or output by the state of the corresponding bit in the data direction register. The D_6 is also used as the $\overline{INT_2}$. When using the D_6 as a port, set the $\overline{INT_2}$ interrupt mask bit in the miscellaneous register to "1" to prevent the $\overline{INT_2}$ interrupt. For details, see "INPUT/OUTPUT PORTS".

- **$\overline{STBY}$**

This is used to put the MCU into the stand-by mode. Setting this pin to "low" level stops the internal oscillator and resets the MCU internal state. See "Stand-by Mode" for additional information.

The following are input/output pins for serial communication interface (SCI). These are used as D_3, D_4, or D_5. For details, see "SERIAL COMMUNICATION INTERFACE".

- **$\overline{CK}$ (D_5)**

This is used to input or output clocks when receiving or transmitting serial data.

- **R_x (D_4)**

This is used to receive serial data.

- **T_x (D_3)**

This is used to transmit serial data.

■ MEMORY MAP

The memory map of the HD63705V0 MCU is shown in Fig. 6. During the processing of the interrupt, the register contents are pushed onto the stack in the order shown in Fig. 7. The stack pointer decrements. The low order byte (PCL) of the program counter is stacked first and the high order byte (PCH) of the program, the index register (X), the accumulator (A) and the condition code register (CCR) are stacked in that order. For subroutine calls, program counter (PCH, PCL) contents are pushed onto the stack.

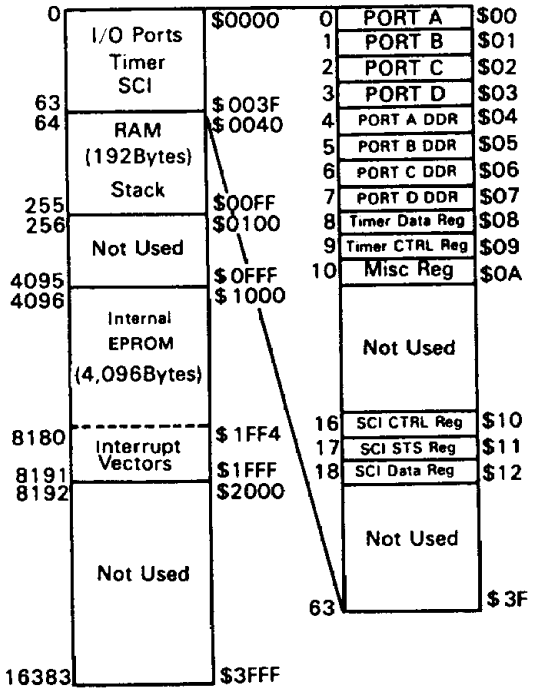
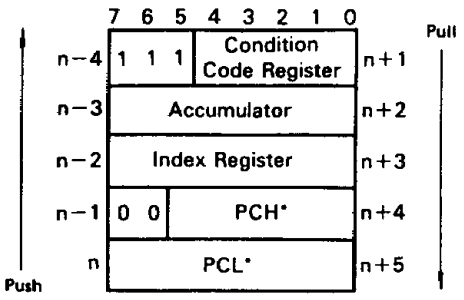


Figure 6 HD63705V0 MCU Memory Map



* For subroutine calls, only PCL and PCH are stacked.

Figure 7 Interrupt Stacking Order

■ REGISTER

This CPU contains five registers available to the programmer. They are shown in Fig. 8.

- **Accumulator (A)**

The accumulator is an 8-bit general purpose register which holds operands and results of the arithmetic operations or data manipulations.

- **Index Register (X)**

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value which is added to an offset to create an effective address.

The index register can also be used for data manipulations with read-modify-write instructions.

When not performing addressing operations, the register can be used as a temporary storage area.

- **Program Counter (PC)**

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 containing the address of the next free location of the stack. Initially, the stack pointer is set to location $00FF. It is decremented as a data is pushed on to the stack and incremented as data is then poped out of the stack. The 8 high-order bits of the stack pointer are fixed to 00000011. During an MCU reset or a reset stack pointer (RSP) instruction, the pointer is set to location $00FF. Subroutines and interrupts may be nested down to $00C1, which allows programmers to use up to 31 levels of subroutine calls and 12 levels of interrupt responses.

- **Condition Code Register (CCR)**

The condition code register is a 5-bit register indicating the results of the instruction just executed. These bits can be individually tested by conditional branch instructions. Each bit is described in the following paragraphs.

- **Half Carry (H)**

When set, this bit indicates that a carry occurred between bit 3 and 4 during an arithmetic operation (ADD, ADC).

- **Interrupt (I)**

Setting this bit masks all interrupts except for software ones. If an interrupt occurs while this bit is set, the interrupt is latched and is processed as soon as the interrupt bit(I) is cleared. (More precisely the interrupt enters the servicing routine after the instruction next to the CLI is executed.)

- **Negative (N)**

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation is negative. (Bit 7 in the result is a logical "1".)

- **Zero (Z)**

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

- **Carry/Borrow (C)**

When set, this bit indicates that a carry or borrow occurred during the last arithmetic operation. This bit is also affected by bit test and branch instructions, shifts and rotates.

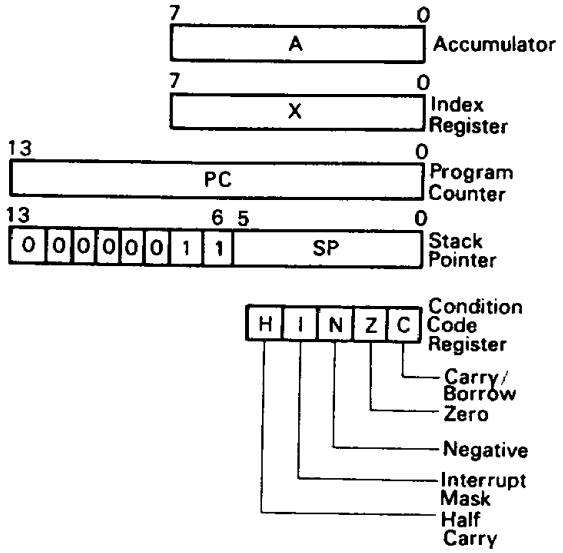


Figure 8 Programming Model

■ INTERRUPTS

The HD63705V0 can be interrupted six different ways through the external interrupt input pins ($\overline{INT}$, $\overline{INT_2}$), the internal timer interrupts (TIMER, $TIMER_2$), the serial interrupt (SCI) and the software interrupt instruction (SWI).

$\overline{INT_2}$ and TIMER, and SCI and TIMER2 respectively generate the same vector address although a different vector address is generated for TIMER during the wait mode as shown in Table 1.

When any interrupt occurs, processing is suspended, the present CPU state pushed onto the stack, the interrupt bit (I) in the condition code register is set, the address of the interrupt service routine is obtained from the appropriate interrupt vector address, and the interrupt routine is executed. The RTI instruction causes the CPU to return to normal processing. This instruction unstacks the previous CPU state and allows the CPU to resume processing of the program from the next instruction to the interrupted one. Table 1 lists the execution priority of interrupts and the vector addresses.

Fig. 9 shows a flowchart of the interrupt sequence. A diagram of the interrupt request source is shown in Fig. 10.

Either a level-sensitive and negative edge-sensitive trigger or negative edge-sensitive only trigger is available to the external interrupt $\overline{INT}$, depending on the state of bit 5 in the miscellaneous register. (Setting the bit selects level-sensitive and negative edge-sensitive trigger inputs and clearing it selects negative edge-sensitive only trigger inputs.) The $\overline{INT_2}$ pin is an edge-sensitive trigger input. An interrupt request occurs on the negative edge of $\overline{INT_2}$ and it is then latched. When a negative edge-sensitive trigger is used, the $\overline{INT}$ interrupt request is automatically cleared as soon as a program jumps to the $\overline{INT}$ service routine. The $\overline{INT_2}$ interrupt request is cleared when "0" is written in bit 7 of the miscellaneous register.

Requests for external interrupts ($\overline{INT}$, $\overline{INT_2}$), internal timer interrupts (TIMER, TIMER2) and serial interrupt (SCI) are held but not serviced while I bit of the condition code register is set. As soon as the I bit is cleared, the corresponding interrupts jumps to the interrupt service routines. The $\overline{INT_2}$ 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 $TIMER_2$ interrupt by bit 4 of the serial status register.

The state of the $\overline{INT}$ pin is tested by BIL and BIH instructions. The $\overline{INT}$ negative edge detect circuit and its latch circuit are independent of tests by these instructions. The state of $\overline{INT_2}$ pins is also independent.

Table 1 Interrupt Execution Priority

Interrupt	Priority	Vector Address
$\overline{RES}$	1	$1FFE, $1FFF
SWI	2	$1FFC, $1FFD
$\overline{INT}$	3	$1FFA, $1FFB
TIMER/$\overline{INT_2}$	4	$1FF8, $1FF9
TIMER (Wait Mode)	5	$1FF6, $1FF7
SCI/$TIMER_2$	6	$1FF4, $1FF5

■ MISCELLANEOUS REGISTER (MR: $000A)

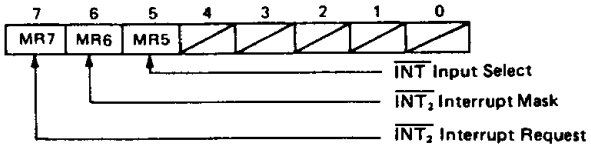
The miscellaneous register is used to determine whether a level-sensitive and negative edge-sensitive trigger or negative edge-sensitive only trigger is available to the external interrupt $\overline{INT}$. It is also used to control the $\overline{INT_2}$ interrupt.

The bit 7 is the $\overline{INT_2}$ interrupt request bit. This bit is set when a negative edge is detected on $\overline{INT_2}$ pin. The $\overline{INT_2}$ can be verified by software during the interrupt service routine (vector address : $1FF8, $1FF9). This bit must be cleared by software.

The bit 6 is the $\overline{INT_2}$ interrupt mask bit. When set, this bit inhibits the $\overline{INT_2}$ interrupt. Both read and write operations are available to the bit 7, but "1" can not be written into this bit by software. Thus, the $\overline{INT_2}$ interrupt can not be requested by software.

At reset, the bit 7 is cleared, the bit 6 is set and the bit 5 is cleared.

Miscellaneous Register (MR; $000A)



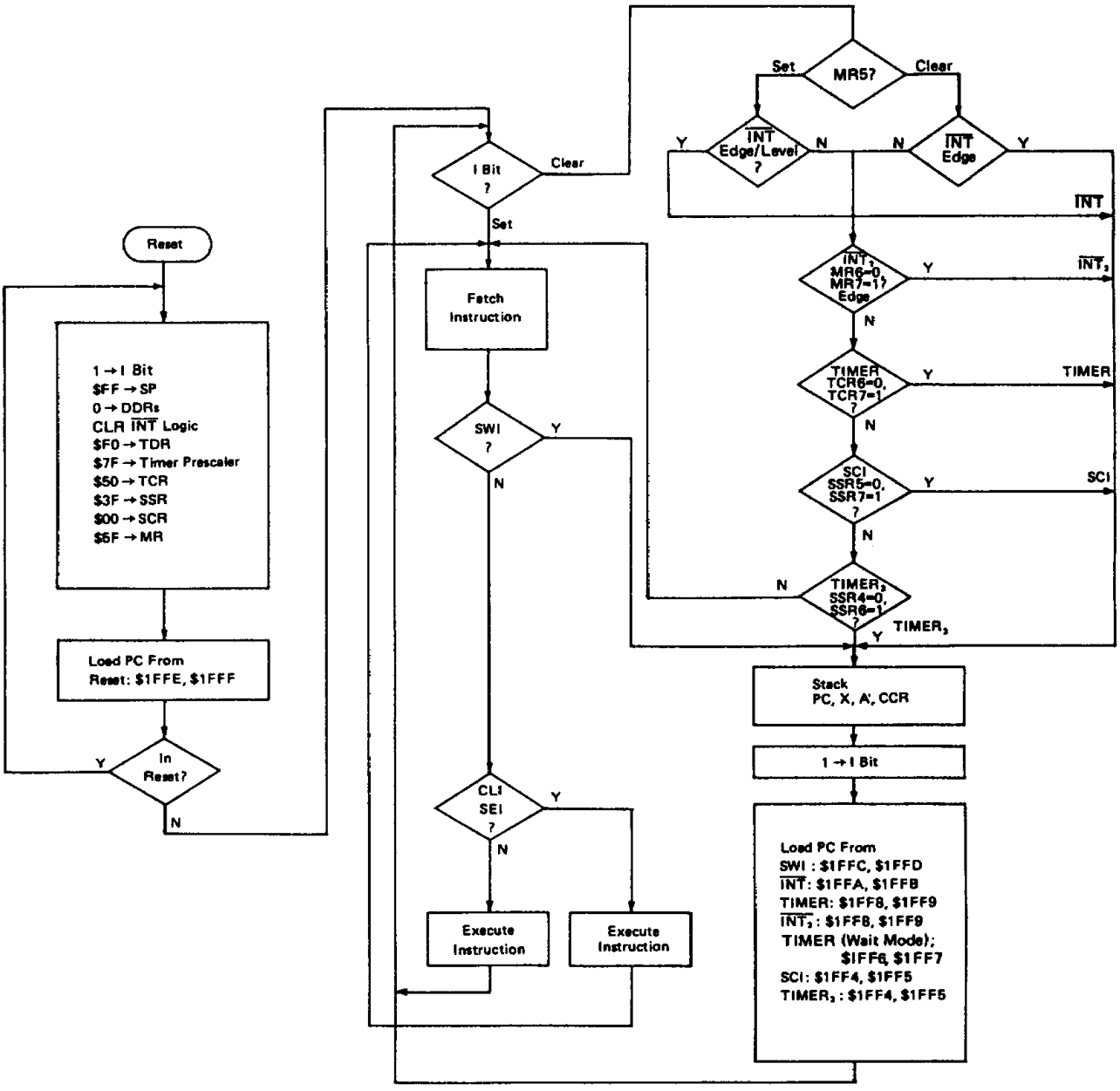


Figure 9 Interrupt Flowchart

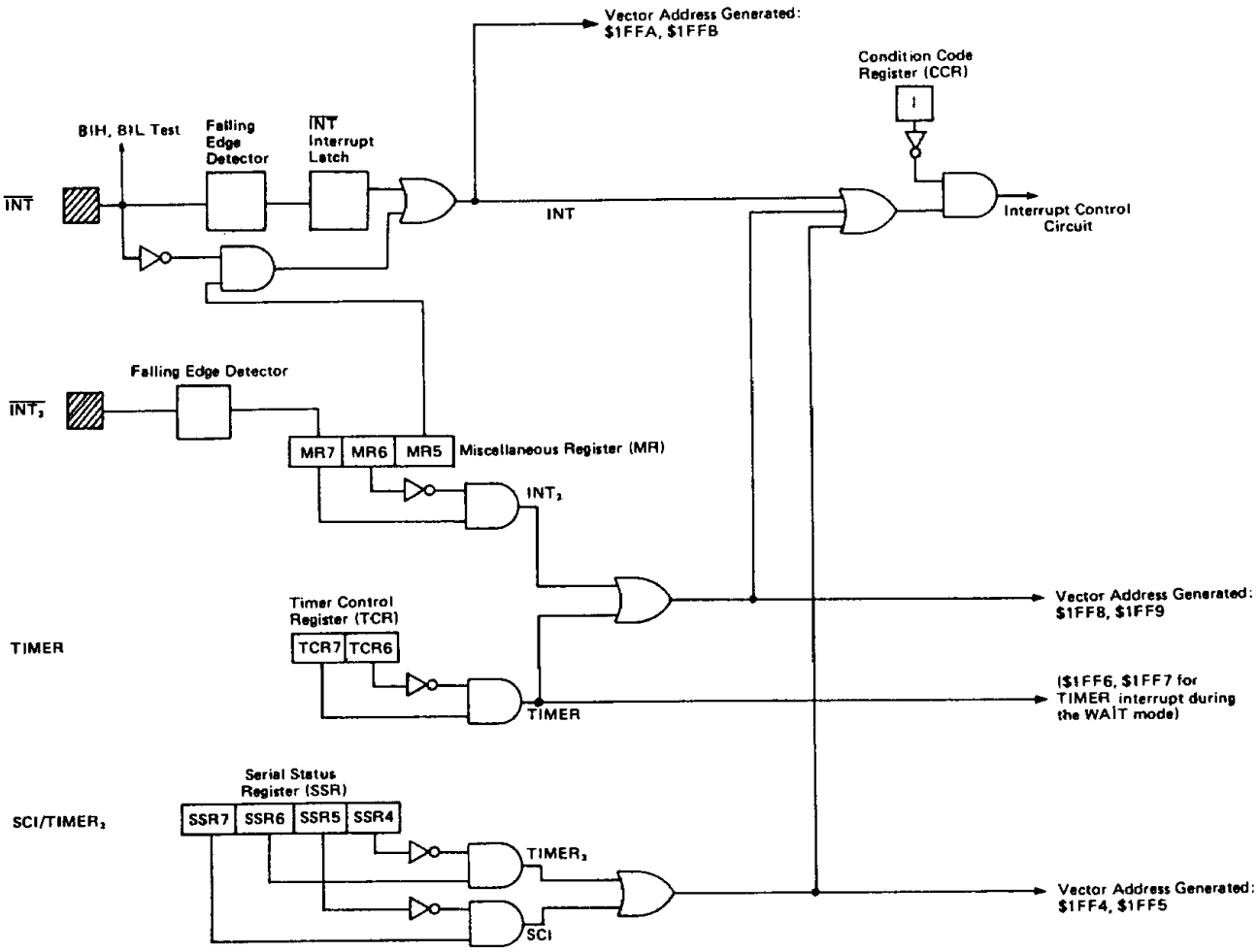


Figure 10 Interrupt Request Generation Circuitry

■ TIMER

Fig. 11 contains a block diagram of the MCU timer. The 8-bit counter may be loaded under program control and is decremented towards zero by the clock input. When the counter, that is, the timer data register (TDR) reaches zero, the timer interrupt request bit (bit 7) of the timer control register is set. When recognizing the interrupt request, the MCU proceeds to store the current CPU state on the stack, and then fetches the timer vector address from locations $1FF8 and $1FF9 (or $1FF6 and $1FF7 if in the wait mode) in order to execute the interrupt service routine. The timer interrupt can be masked by setting the interrupt mask bit (bit 6) of the timer control register. The mask bit (I) of the condition code register can also disable the timer interrupt.

The clock input to the timer can be from an external source applied to the TIMER input pin, or it can be the internal E signal (which is a clock obtained by dividing the oscillator clock by four). When the E signal is used as the source, it can be gated by an input applied to the TIMER pin.

The counter start counting down from "$FF" after it reaches zero. The counter may be monitored at any time by reading the contents of the timer data register. This allows a program to determine the length of the time since the occurrence of a timer interrupt and does not disturb the counter contents.

At reset, the prescaler and counter are initialized to "$7F" and "F0" respectively. The timer interrupt request bit (bit 7) is cleared and the timer interrupt request mask (bit 6) is set.

The timer interrupt request bit must be cleared by software.

TCR7	Timer Interrupt Request
0	Not Requested
1	Requested

TCR6	Timer Interrupt Mask
0	Interrupt Allowed
1	Interrupt Inhibited

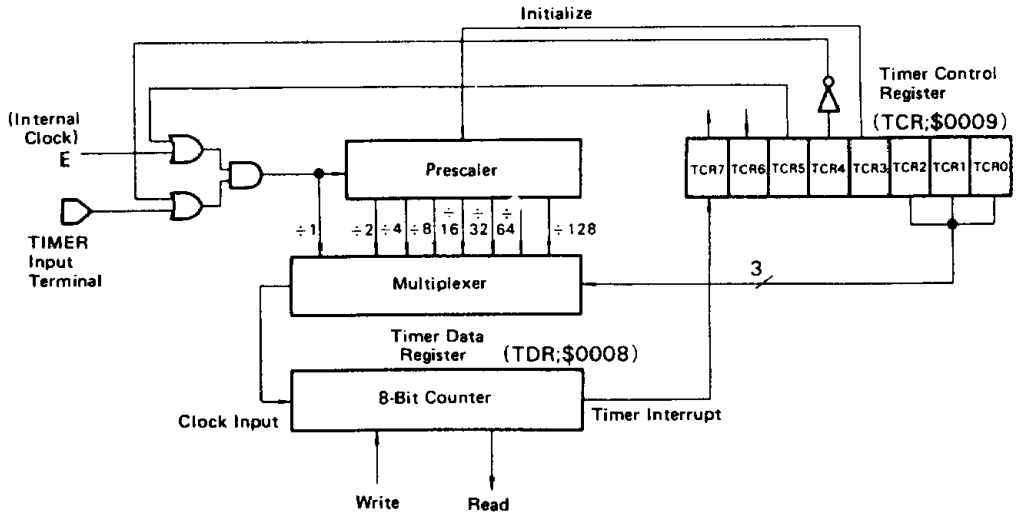
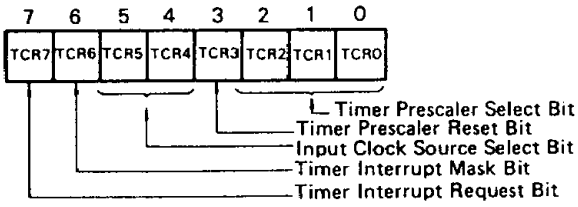


Figure 11 Timer Block Diagram

- **Timer Control Register (TCR:$0009)**

Selection of the input clock source, selection of the prescaler and the control of timer interrupt can be accomplished by setting the corresponding bits in the timer control register (TCR: $0009).

Timer Control Register (TCR: $0009)



One of four input clock sources (shown in Table 2) can be selected depending on the value written into the TCR4 and TCR5.

After reset, these bits are initialized to the "AND of External Clock (applied to TIMER pin) and Internal Clocks (E)" mode (TCR4=0, TCR5=0). Thus, if the TIMER pin is "1", the counter starts to count down from "$F0" just after reset.

Table 2 Input Clock Source Modes

TCR		Clock Input Source
Bit 5	Bit 4	
0	0	Internal Clock (E)
0	1	AND of External Clock (applied to TIMER pin) and Internal Clock (E)
1	0	No Clock
1	1	External Clock (through TIMER pin)

Writing "1" to the TCR3 initializes the prescaler to zero. A read of this bit always indicates a "0".

The coding of the TCR0 to TCR2 bits produces a division in the prescaler as shown in Table 3. One of eight outputs of the prescaler (÷1, ÷2, ÷ 4, ÷8, ÷16, ÷32, ÷64, ÷128) is selectable. After reset, these bits are set to the "÷1" mode.

Table 3 Prescaler Division Select

TCR			Prescaler Division
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 timer interrupt is enabled when the TCR6 bit is "0" and disabled when the bit is "1". When a timer interrupt occurs, the TCR7 bit is set to "1". This bit must be cleared by writing "0" into it.

- **SERIAL COMMUNICATION INTERFACE (SCI)**

The SCI is used for a serial transfer of 8-bit data. The transfer clock width ranges from 1 μs to about 32 ms (with a 4 MHz oscillator), and sixteen types of transfer clock widths are available. The SCI consists of three registers, an octal counter and a prescaler as shown in Fig. 10. It communicates with CPU via the data bus lines and with peripherals via bits 3, 4 and 5 of Port D. The following are descriptions of the registers and the data transfer operations.

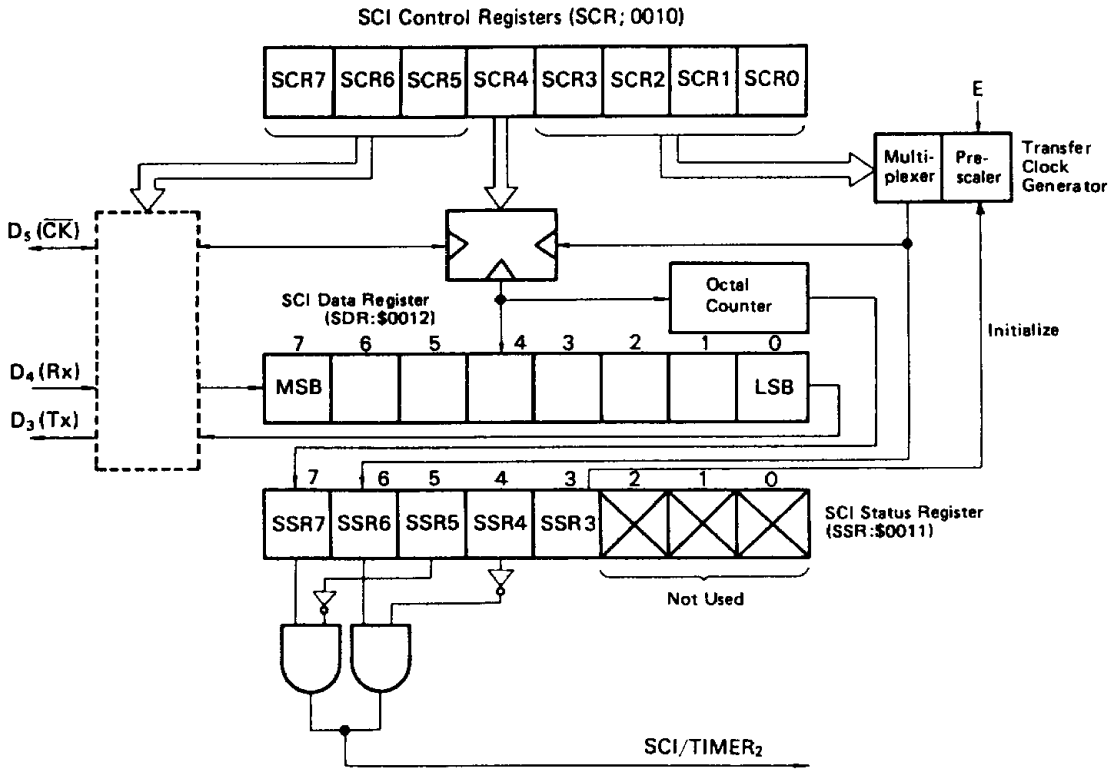


Figure 12 SCI Block Diagram

- **SCI CONTROL REGISTER (SCR: $0010)**

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

Bit 7 (SCR7)

When this bit is set, the DDR bit corresponding to the D_3 pin goes to "1" and the D_3 pin acts as output of SCI data. Resetting the MCU clears this bit.

SCR7	D_3
0	Used as I/O pin (by DDR)
1	Serial data output (DDR output)

Bit 6 (SCR6)

When this bit is set, the DDR bit corresponding to the D_4 pin goes to "1" and the D_4 pin acts as input of SCI data. Resetting the MCU clears this bit.

SCR6	D_4
0	Used as I/O pin (by DDR)
1	Serial data input (DDR output)

Bit 5, 4 (SCR5, SCR4)

These two bits select a clock source. Resetting the MCU clears all these bits.

SCR5	SCR4	Clock Source	D_5
0	0	–	Used as I/O pin (by DDR)
0	1	–	
1	0	Internal	Clock output (DDR output)
1	1	External	Clock input (DDR input)

Bit 3 ~ 0 (SCR3 ~ SCR0)

These bits select a transfer clock width. Resetting the MCU clears all these bits.

SCR3	SCR2	SCR1	SCR0	Transfer Clock Width	
				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 used for data transfer.

- **SCI Status Register (SSR: $0011)**

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

Bit 7 (SSR7)

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

Bit 6 (SSR6)

This is the $TIMER_2$ interrupt request bit. The $TIMER_2$ is also used as the serial clock generator, and this bit is set on the every negative edge of the internal transfer clock. When the MCU is reset, the bit is cleared. It can also be cleared by writing "0" into it. (For details, see "TIMER").

Bit 5 (SSR5)

This is the SCI interrupt mask bit which can be set or cleared by software. When this bit is "1", the SCI interrupt (SSR7) is masked. Resetting the MCU sets this bit to "1".

Bit 4 (SSR4)

This is the $TIMER_2$ interrupt mask bit which can be set or cleared by software. When this bit is "1", the $TIMER_2$ interrupt (SSR6) is masked. Resetting the MCU sets this bit to "1".

Bit 3 (SSR3)

Writing "1" into this bit initializes the prescaler of the transfer clock generator. A read of this bit always indicates "0".

Bit 2 ~ Bit 0

Not Used.

SSR 7	SCI Interrupt Request
0	Not Requested
1	Requested

SSR6	$TIMER_2$ Interrupt Request
0	Not Requested
1	Requested

SSR5	SCI Interrupt Mask
0	Interrupt Allowed
1	Interrupt Inhibited

SSR4	$TIMER_2$ Interrupt Mask
0	Interrupt Allowed
1	Interrupt Inhibited

- **Transmit Operations**

The transfer clock width and the clock source are determined by setting the corresponding SCI control register bits, and then the D_3 pin and the D_5 pin are set to serial data output pin and serial clock pin respectively. The transmit data should be moved from the accumulator or the index register into the SCI data register. Then, the data moved into the SCI data register is output through the D_3/Tx pin, starting with the LSB, synchronously with the negative edge of the serial clock. See Fig. 13. When 8 bits of data have been transmitted, the bit 7 of the SCI status register (interrupt request bit) is set on the positive edge of the last serial clock. This interrupt request can be masked by setting bit 5 of the SCI status register. After completion of the data transmission, the 8th bit of data (MSB) stays at the D_3/Tx pin. If the external clock source is selected, the transfer clock width determined by bit 0 to bit 3 of the SCI control register is ignored and the $D_5/\overline{CK}$ pin is set as input. If the internal clock source is selected, the $D_5/\overline{CK}$ is set as output and clocks are generated with the transfer clock width selected by bit 0 to 3 of the SCI control register.

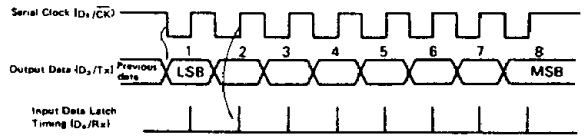


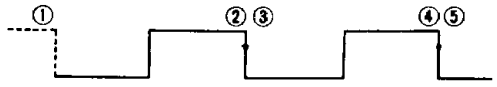
Figure 13 SCI Timing

- **Receive Operations**

The transfer clock width and the clock source are determined by setting the corresponding SCI control register bits, and the D_4 pin and the D_5 pin are set to serial data input pin and serial clock pin respectively. Then the receive operation is enabled by dummy-reading or -writing the SCI data register. (This procedure is not needed after a data is received. It is needed after reset or when no data is received yet.) The received data through the D_4/Rx pin is input to the SCI data register synchronously with the positive edge of the serial clock. (See Fig. 13.) At completion of 8-bit data reception, the bit 7 in the SCI status register (interrupt request bit) is set. This interrupt request can be masked by setting bit 5 of the SCI status register. If the external clock source is selected, the transfer clock width determined by bit 0 to bit 3 of the SCI control register is ignored and data is received synchronously with the clock input through the $D_5/\overline{CK}$ pin. If the internal clock source is selected, the $D_5/\overline{CK}$ acts as an output and clocks are output with the transfer clock selected by bit 0 to bit 3 of the SCI control register.

- **TIMER2**

The SCI transfer clock generator can be used as a timer. The clock which is selected by bit 3 to 0 in the SCI control register (4 μs to approx. 32 ms with 4 MHz oscillator) is input to the bit 6 in the SCI status register ($TIMER_2$ interrupt request bit), and this bit is set on every negative edge of the clock. Thus, TIMER2 can be used as a reload counter or clock.



① : Transfer clock generator is reset and TIMER2 interrupt mask bit (bit 4 of SCI status register) is cleared.

②, ④ : $TIMER_2$ interrupt request bit is set.

③, ⑤ : $TIMER_2$ interrupt request bit is cleared.

$TIMER_2$ is used as the SCI transfer clock generator. If wanting to use the $TIMER_2$ independently of the SCI, select external clock source (SCR5=1, SCR4=1).

If internal clock source is selected, reading from or writing to the SCI data register causes the prescaler of the transfer clock generator to be initialized.

■ I/O PORTS

There are 31 input/output pins (Ports A, B, C and D). Each I/O pin is programmed by setting the corresponding bit in the

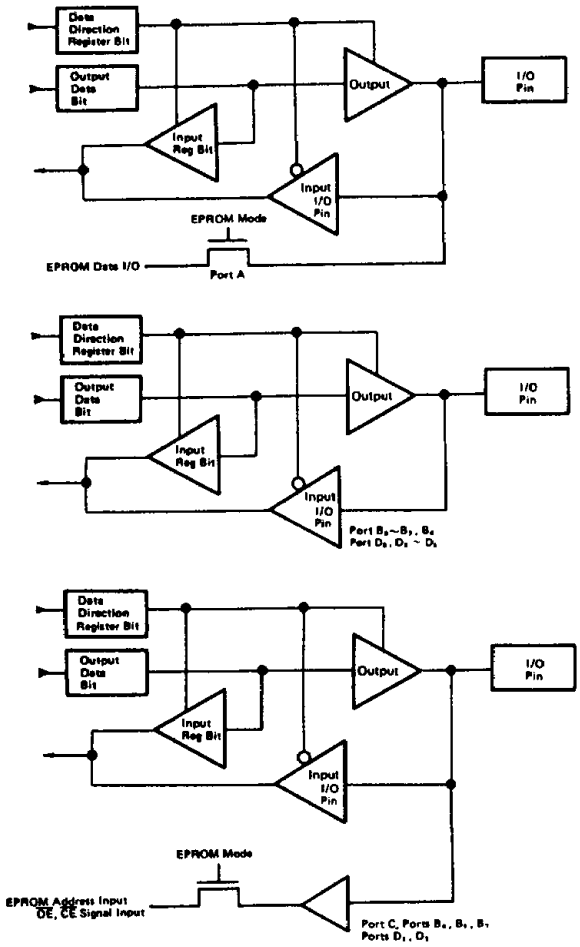


Figure 14 I/O Port (Ports A, B, C, D) Diagram

data direction register (DDR) to "1" for output or "0" for input. When programmed as outputs, all I/O ports read latched data regardless of the logic levels at the output pin due to output loading. (See Fig. 14)

On reset, the DDRs and data registers go to "0", which places all the ports in the input mode.

All input/output pins are TTL compatible and CMOS compatible as both inputs and outputs.

When not used, the I/O ports should be connected to V_{SS} via resistors. With no lines connected to these pins, power may be consumed despite their not being used.

In EPROM mode, port A is used as EPROM data buses ($EO_0 \sim EO_7$), port C as low order EPROM address buses ($EA_0 \sim EA_7$), ports B_4, B_5 and B_7 as EA_{11}, EA_{10} and EA_8 respectively. (The B_4, B_5 and B_7 pins are three of high order EPROM address buses.)

Whether the port A data buses are used as inputs or outputs depends on $\overline{OE}$ and $\overline{CE}$ signals regardless of the state of the DDR bits.

DDR Bit	Output Data Bit	Output State	Input to CPU
1	0	0	0
1	1	1	1
0	X	3-State	Pin

■ MODE SELECTION

The HD63705V0 is capable of operating in two modes, the MCU mode and the EPROM mode.

The operating mode is determined by mode program pins; NUM pin and $\overline{STBY}$ pin as shown in Table 4.

• MCU Mode

In this mode, all ports are available. (See Fig. 15.)

Table 4 Mode Selection

Mode	NUM	$\overline{STBY}$	EPROM	RAM	Interrupt Vector	Operation Mode
MCU Mode	"L"	*	I	I	I	Single Chip Mode
EPROM Mode	"H"	"L"	I	*	*	EPROM Programming Mode

"L" = logic "0", "H" = logic "1", I; Internal, *Don't care

• EPROM Mode

In this mode, the EPROM can be programmed. For detail, see "PROGRAMMING THE EPROM."

• Mode and Port

Table 5 shows MCU port condition in each mode.

Table 5 MCU Port Condition

Port \ Mode	MCU Mode	EPROM Mode
Port A	I/O Port	Data Bus ($EO_0 \sim EO_7$)
Port B	I/O Port	Address Bus EA_8 (B_7), EA_{10}, EA_{11} (B_5, B_4) (Note 1)
Port C	I/O Port	Address Bus ($EA_0 \sim EA_7$)
Port D	I/O Port	$\overline{OE}$(D_1), $\overline{CE}$(D_2) (Note 2)
$\overline{INT}$	Input	Address Bus (EA_9)
Timer	Input	Program Voltage V_{PP}

(Note 1) $B_0 \sim B_3$ are not used. B_3 should be connected to V_{SS}. B_6 is not used.
(Note 2) D_0 or $D_3 \sim D_5$ are not used. D_6 should be connected to V_{SS}.

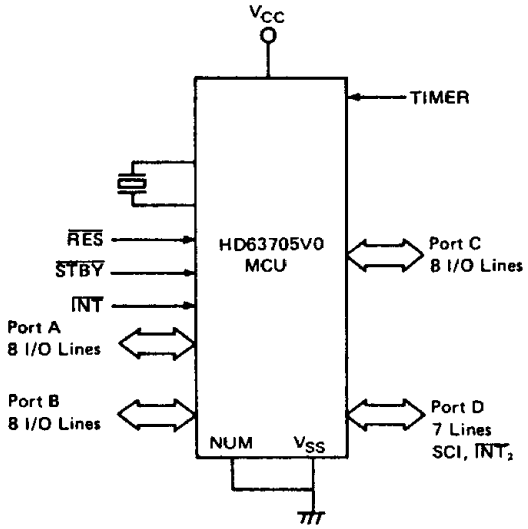


Figure 15

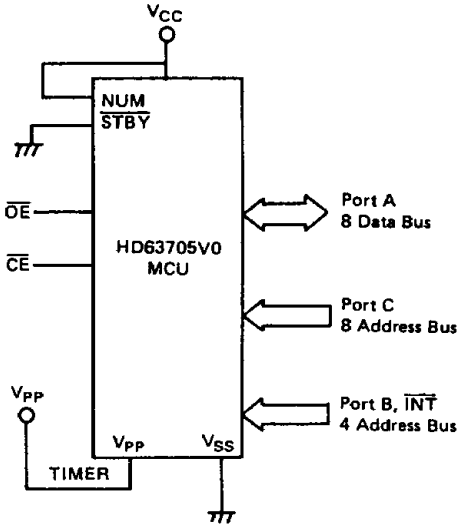


Figure 16

■ MEMORY MAP

A memory map for each operating mode is shown in Fig. 17.

The first 32 locations of the MCU mode map are reserved for the MCU internal register area as shown in Table 6.

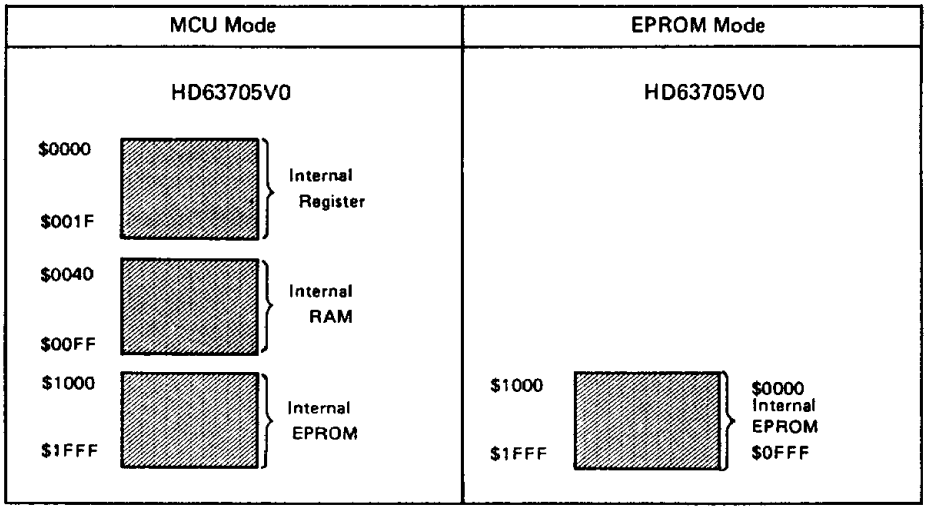


Figure 17 HD63705V0 Memory Map

Table 6 Internal Register

(R/W) : Read/Write Register
(R) : Read Only Register
(W) : Write Only Register

Register	Address	Read/Write *1 /Initial Value after Reset							
		7	6	5	4	3	2	1	0
Port A Data Register	$00	R/W							
		$00							
Port B Data Register	$01	R/W							
		$00							
Port C Data Register	$02	R/W							
		$00							
Port D Data Register	$03	Not Used	R/W						
		1	$00						
Port A Data Direction Register	$04	R/W							
		$00							
Port B Data Direction Register	$05	R/W							
		$00							
Port C Data Direction Register	$06	R/W							
		$00							
Port D Data Direction Register	$07	Not Used	R/W						
		1	0	0	0	0	0	0	0
Timer Data Register	$08	R/W							
		$F0							
Timer Control Register	$09	R/W							
		0	1	0	1	0	0	0	0
Miscellaneous Register	$0A	R/W			Not Used				
		0*2	1	0	1	1	1	1	1
Not Used	$0B ⟩ $0F								
SCI Control Register	$10	R/W							
		$00							
SCI Status Register	$11	R/W							
		0*2	0*2	1	1	1	1	1	1
SCI Data Register	$12	R/W							
		Undefined							

*1 R: Read Only W: Write Only R/W: Readable/Writeable
*2 Only "0" can be written.

■ PROGRAMMING THE EPROM

In EPROM mode, where the MCU functions are not available, the HD63705V0 can program its own internal EPROM of 4k bytes in the same way as 27256 EPROM type. The MCU enter the EPROM mode if NUM is set to "high" state and $\overline{STBY}$ to "Low" state as shown in Table 4. In this mode, Ports $A_0 \sim A_7$ are used as data bus, Ports $C_0 \sim C_7$ and Ports B_4, B_5, B_7 as address bus, Port D_1 as $\overline{OE}$ input, and Port D_2 as $\overline{CE}$ input.

• Programming/Verification

When $\overline{CE}$ pin is set to "Low" state and $\overline{OE}$ pin is set to "High" state after the program voltage (V_{PP}) is applied to V_{PP} pin, the data are written into the PEROM through Port A one byte for each EPROM location. When $\overline{OE}$ pin is set to "Low" after programming, the programmed data is output through Port A. This allows the user to verify the data. I/O timing of these signals is shown in Fig. 3 and 4. When both $\overline{CE}$ and $\overline{OE}$ pins are returned to "High", Port A will be put in a high impedance state and EPROM programming/verification will be inhibited.

Table 7 shows the state of each pin in EPROM mode.

Table 7 Pin Conditions in EPROM Mode

	V_{CC}	V_{SS}	TIMER (V_{PP})	$\overline{CE}$	$\overline{OE}$	Port $A_0 \sim A_7$		NUM	$\overline{STBY}$	Port B_3, D_6
Programming	+6	GND	V_{PP}	"L"	"H"	Data input	Address input	"H"	"L"	GND
Verification	+6	GND	V_{PP}	Dont't care	"L"	Data output	Address input	"H"	"L"	GND
Programming/ Verification Disable	+6	GND	V_{PP}	"H"	"H"	High impedance	Don't care	"H"	"L"	GND
Read	+5	GND	+5	"L"	"L"	Data output	Address input	"H"	"L"	GND
Output Disable	+5	GND	+5	"L"	"H"	High impedance	Don't care	"H"	"L"	GND

"H"; V_{IH} level, "L"; V_{IL} level

• Erasure

The HD63705V0 EPROM can be erased by exposure to ultraviolet light with a wave length of 2537A and a minimum integrated dose of 15W sec/cm^2. The LSI should be positioned about one inch from an ultraviolet lamp (12,000 μW/cm^2 rating) and exposed to light for twenty to thirty minutes.

The erasure sets all bits of EPROM to the "1" state.

(NOTE) Any stair or foreign material at the surface of the EPROM window lowers the transmission rate, prolonging the erasing time. Remove them from the window with alcohol or other solvent that does not damage the package. Don't rub the window hard but wipe out softly.

■ RESET

The MCU can be reset in two ways: by the external reset input and by the initial power up (power-on reset). Refer to Fig. 18 for the power on reset. On power up, the RESET input must remain "Low" for a minimum of t_{osc}. This time allows the internal oscillator to stabilize.

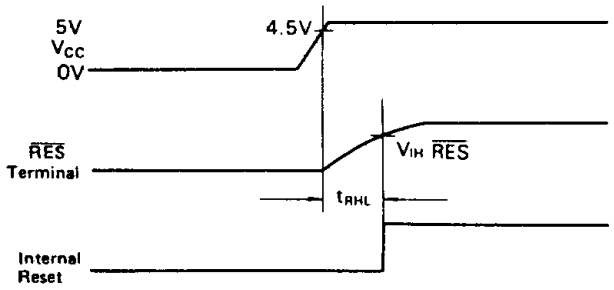


Figure 18 Power and Reset Timing

Connecting a capacitance to the $\overline{RES}$ input, as shown in Fig. 19, provides sufficient delay.

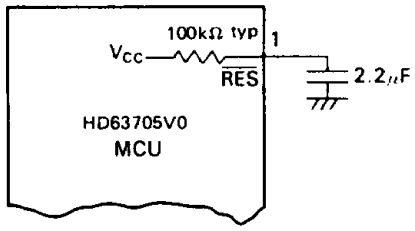


Figure 19 Power-ON Reset Delay

• INTERNAL OSCILLATOR

The internal oscillator circuit is designed to meet the requirement for a minimum of external components. Connecting a crystal (AT cut 2.0 ~ 8.0 MHz) or a ceramic resonator between pins 38 and 39 drives the internal oscillator with varying degrees of stability.

The different connection methods are shown in Fig. 20. Fig. 21 and 22 illustrate crystal specifications and typical arrangement of the crystal respectively.

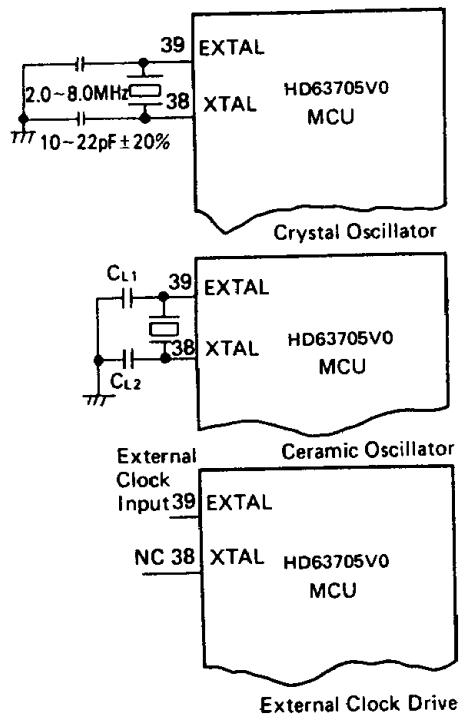


Figure 20 Internal Oscillator Options

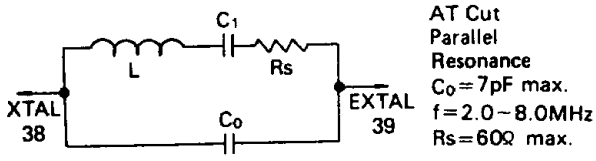
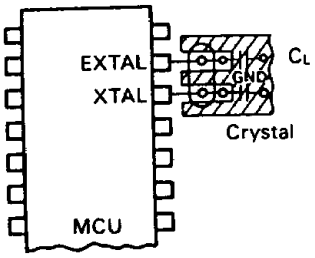


Figure 21 Crystal Parameters



(Note) Keep crystal leads and circuit connections as short as possible. Do not allow these connections to cross others.

Figure 22 Typical Arrangement of Crystal

■ LOW POWER MODES

The HD63705V0 has three low power modes: wait, stop, and standby.

• Wait Mode

The MCU enters the wait mode by the execution of WAIT instruction. In this mode, the oscillator remains active but the internal clock is not provided to the CPU. The CPU processing is stopped, while the peripherals, that is, the timer and the serial communication interface system remain active. (Note: Once the MCU enters the wait mode, serial communication interface cannot be triggered.) In the wait mode the I bit in the condition code register is cleared. All other registers, RAM, and input/output pins remain unchanged.

The MCU can be released from this mode by interrupts ($\overline{INT}$, TIMER/$\overline{INT_2}$, SCI/TIMER$_2$), $\overline{RES}$ and $\overline{STBY}$. The $\overline{RES}$ reset the MCU and the $\overline{STBY}$ puts the MCU in the standby mode described later. Once the CPU accepts an interrupt, the MCU reverts from the wait mode to the active mode and then executes the interrupt routine. If interrupts other than $\overline{INT}$ (TIMER/$\overline{INT_2}$, SCI/TIMER$_2$) are masked by the timer control register, the miscellaneous register or the serial status register, no interrupt is requested to the CPU, so the MCU remains in the wait mode.

Fig. 23 shows a flowchart of WAIT instruction execution.

• Stop Mode

The MCU enters the stop mode by the execution of STOP instruction. In this mode, the internal oscillator is turned off, causing the CPU processing and peripherals to be halted. The I bit in the condition code register is cleared. In the timer control register (TCR) the bit 7 and bit 6 go to "0" and "1" respectively. In the SCI status register (SSR), the bits 6 and 7 go to "0" and the bits 4 and 5 go to "1". All other registers, RAM, and input/output pins remain in their previous state.

The MCU is released from this mode by external interrupts ($\overline{INT}$, $\overline{INT_2}$), $\overline{RES}$ and $\overline{STBY}$. The $\overline{RES}$ resets the MCU and the $\overline{STBY}$ puts the MCU in the standby mode described later. Once an interrupt is accepted, the CPU reverts from the stop mode to the active mode and executes the interrupt routine. If $\overline{INT_2}$ is masked by the miscellaneous register, there is no interrupt request, so the MCU does not leave the stop mode.

Fig. 24 shows a flowchart of STOP instruction execution.

Fig. 25 shows a timing diagram of the release from the stop mode. In case of the release by an interrupt, the internal oscillator is turned on upon input of the interrupt and after a delay time that allows the oscillator to stabilize, the CPU becomes active. In case of restarting by $\overline{RES}$, setting $\overline{RES}$ input to "0" starts the internal oscillator and then setting $\overline{RES}$ to "1" restarts the CPU processing. The duration of $\overline{RES}$="0" must exceed t_{osc} to allow the oscillator to stabilize.

• Standby Mode

The MCU enters the standby mode when applying a logic low to the $\overline{STBY}$ input. In this mode, all processing is halted and the MCU internal state is reset, while the contents of RAM is held. I/O pins are put in a high impedance state.

Logic high on the $\overline{STBY}$ pin allows the MCU to leave this mode. The external reset input ($\overline{RES}$) should be used to resume the CPU processing. Refer to the $\overline{RES}$ and the $\overline{STBY}$ timing diagram of Fig. 26.

Table 4 lists the state of the MCU in low power modes. Transition between the modes is shown in Fig. 27.

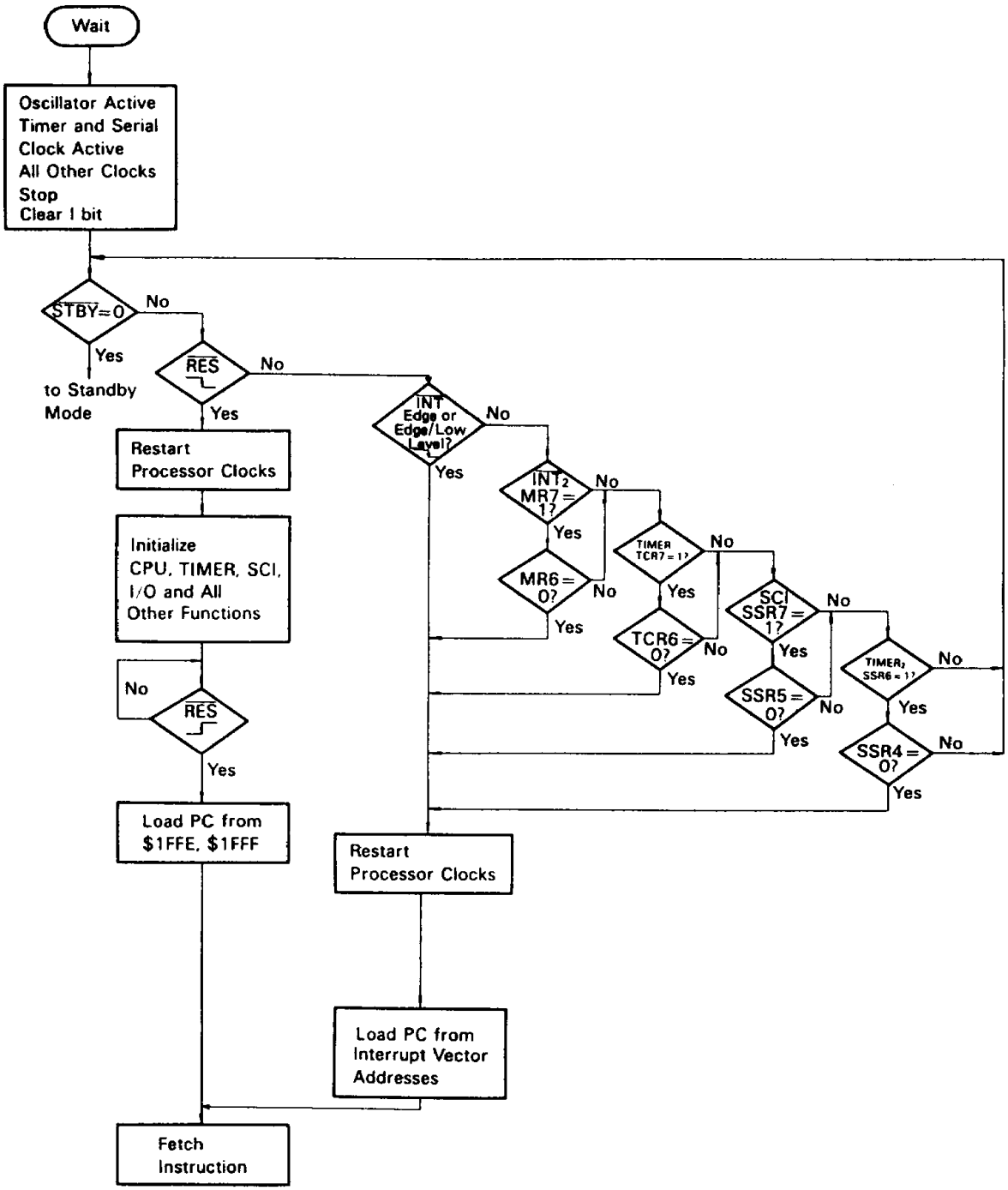


Figure 23 Wait Mode Flowchart

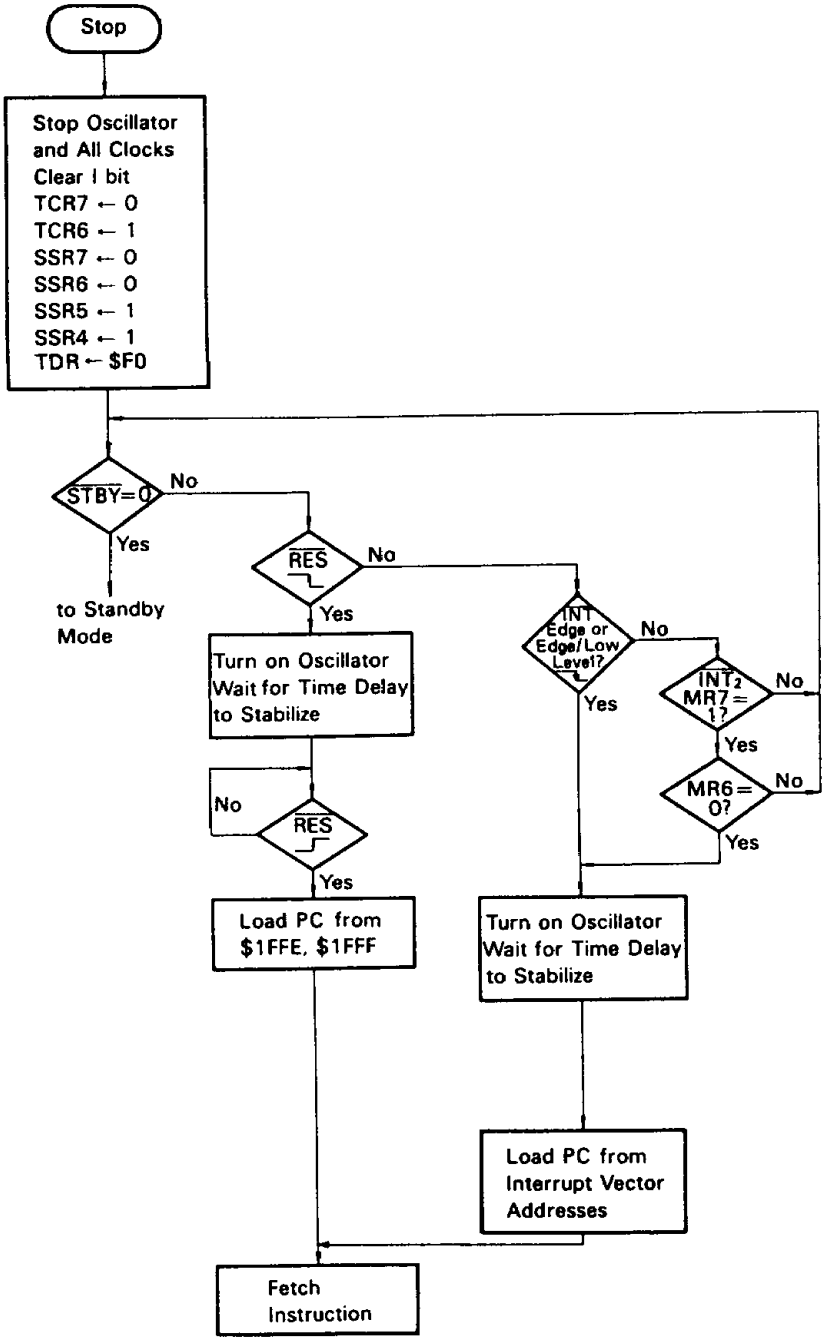
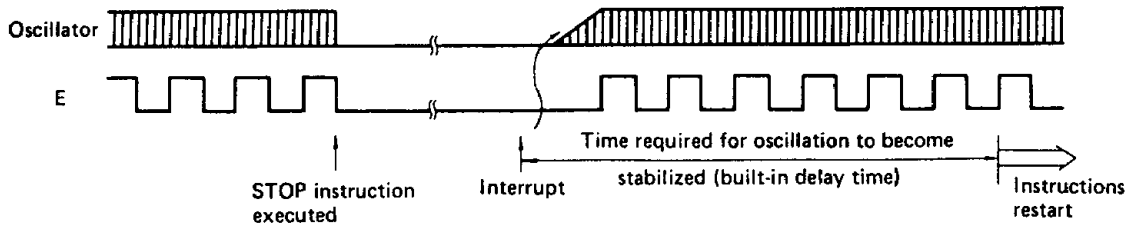
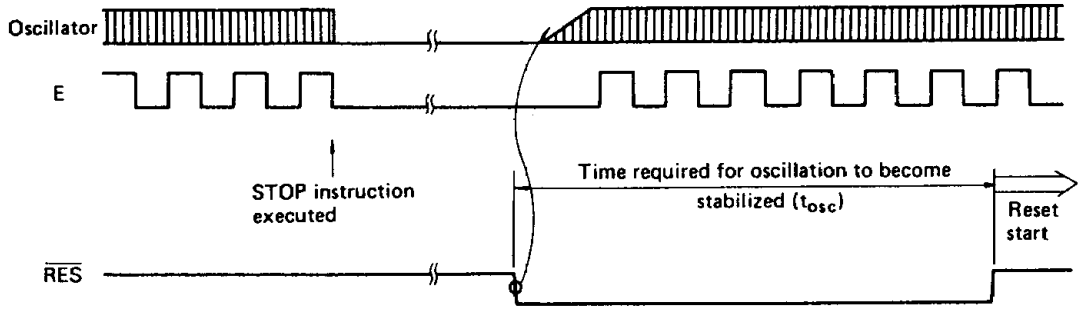


Figure 24 Stop Mode Flowchart



(a) Restart by Interrupt



(b) Restart by Reset

Figure 25 Timing Chart of Releasing from Stop Mode

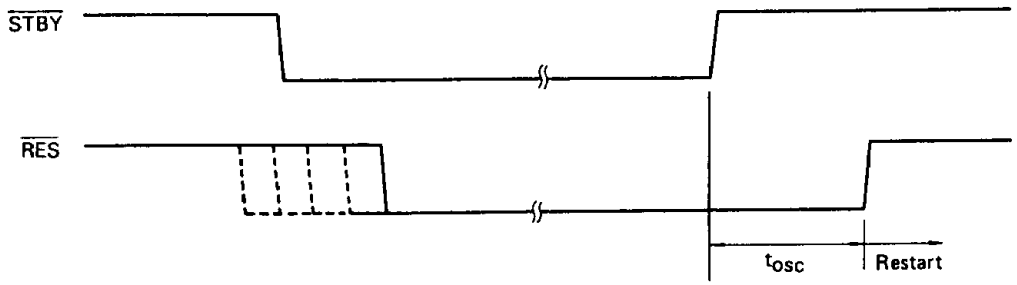


Figure 26 Timing Chart of Releasing from Standby Mode

Table 8 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	Keep	Keep	Keep	$\overline{STBY}$, $\overline{RES}$, $\overline{INT}$, $\overline{INT_2}$, each interrupt request of TIMER, $TIMER_2$, SCI
STOP		STOP instruction	Stop	Stop	Stop	Keep**	Keep	Keep	$\overline{STBY}$, $\overline{RES}$, $\overline{INT}$, $\overline{INT_2}$
Standby	Hardware	$\overline{STBY}$="Low"	Stop	Stop	Stop	Reset	Keep	High impedance	$\overline{STBY}$="High"

*Register in the CPU (except I bit in the CCR)

**See Figure 24

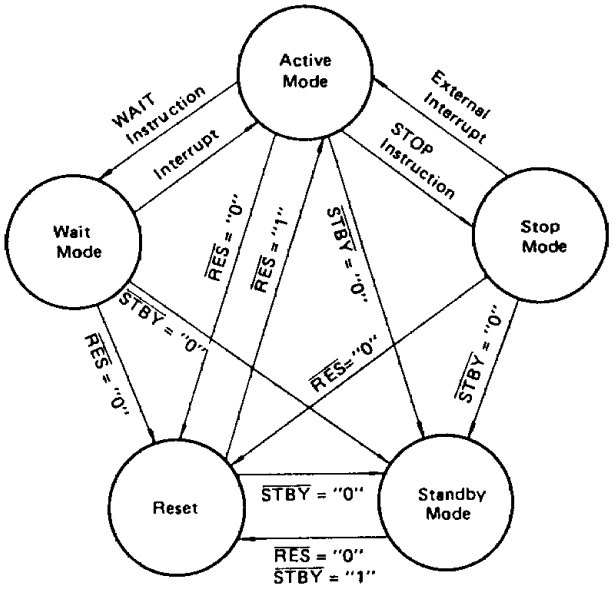


Figure 27 Transitions among Active Mode, Wait Mode, Stop Mode, Standby Mode and Reset

■ BIT MANIPULATION

The HD63705V0 MCU can set or clear one bit of RAM or I/O port with a single instruction (BSET or BCLR). Any memory or I/O port bit in page 0 ($00 to $FF) can be tested by using the BRSET and BRCLR instructions and the program branches to proper routines depending on the result of the test. The bit manipulation in the RAM or I/O allows the user to have individual flags in RAM or to handle a single I/O bits as individual control pins. The example in Fig. 28 illustrates the usefulness of the bit manipulation and test instruction. In the example, the program is constructed assuming that bit 0 of Port A is connected to a zero crossing detector circuit and that bit 1 of Port A is connected to the trigger of a TRIAC.

This program uses only 7 bytes of ROM and provides turn-on of the TRIAC within 10 μs of zero-crossing. The timer is also incorporated to provide turn-on with a needed time of delay which can permit pulse-width modulation of the power.

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

Figure 28 Example of Bit Manipulation

■ ADDRESSING MODES

The HD63705V0 MCU has ten addressing modes. They are explained in the following paragraphs.

- **Immediate**

See Fig. 29. The immediate addressing mode provides access to a constant which does not change during program execution. This access requires an instruction length of 2 bytes. The Effective Address (EA) is the PC and the operand is fetched from the byte following the operation code.

- **Direct**

See Fig. 30. In the direct addressing mode, the address of the operand is contained in the 2nd byte of the instruction. The user can directly address the lowest 256 bytes in memory. The address space of page 0 includes all RAM and I/O registers so that the direct addressing mode may be utilized.

- **Extended**

See Fig. 31. Extended addressing is used for referencing to all memory locations. The EA is contained in the two bytes following the opcode. An extended addressing instruction requires a length of 3 bytes.

- **Relative**

See Fig. 32. Relative addressing is only used in branch instructions. When a branch occurs, the program counter is loaded with the contents of the byte following the opcode. EA=(PC)+2+Rel. where Rel. indicates the 8 bit signed data in the location following the instruction opcode. If no branch occurs, Rel.=0. When a branch takes place, the program jumps to any byte within the range +129 ~ −127 of the present instruction. These branch instructions are two bytes long.

- **Indexed (No Offset)**

See Fig. 33. This addressing mode can access the lowest 256 memory locations. These instruction are one byte long. The EA is the contents of the index register.

- **Indexed (8-bit Offset)**

See Fig. 34. The EA is obtained by adding the contents of index register to that of byte following the opcode. This addressing mode can access the lowest 511 memory locations. These instructions are 2 bytes long.

- **Indexed (16-bit Offset)**

See Fig. 35. The EA is the sum of the contents of the index register and two bytes following the opcode. This addressing mode can access all memory locations. In this mode all instructions are three bytes long.

- **Bit Set/Clear**

See Fig. 36. This addressing mode is available to the BSET and BCLR instructions which can set or clear any bit on page 0. The lower 3 bits of the opcode specify the bit to be set or cleared. The byte following the opcode indicates an address within page 0.

- **Bit Test and Branch**

See Fig. 37. This addressing mode is available to the BRSET and BRCLR instructions which can test the bits within page 0 and can be branched in the relative addressing mode. The address of the byte to be tested is in the single byte immediately following the opcode byte. Individual bits in the byte are specified by the lower 3 bits of the opcode. The relative value is in the third byte and is added to the PC when a branch condition is met. These instructions are 3 bytes long. The value of the tested bit is written in the carry bit of the condition code

register.

- **Implied**

See Fig. 38. The implied addressing mode has no EA. All information necessary to execute the instruction is contained in the opcode. Direct manipulation in the accumulator and index register is included in this implied addressing mode. Instructions such as SWI and RTI are also used in this mode. All instructions used in this mode are one byte long.

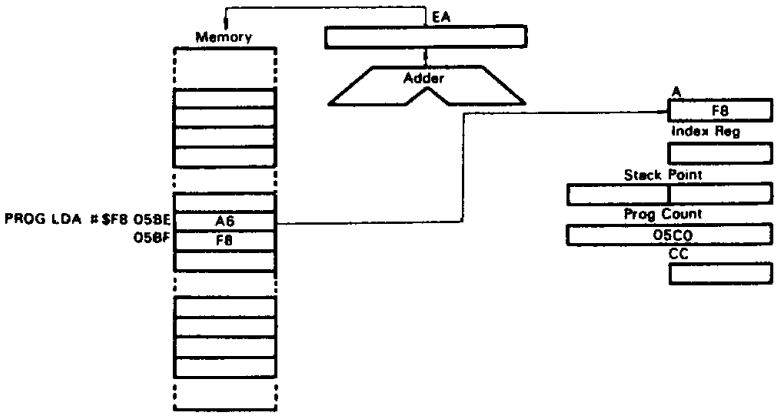


Figure 29 Example of Immediate Addressing

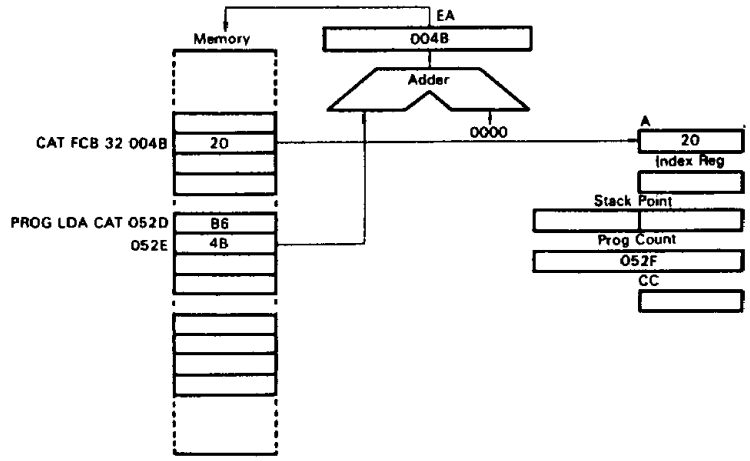


Figure 30 Example of Direct Addressing

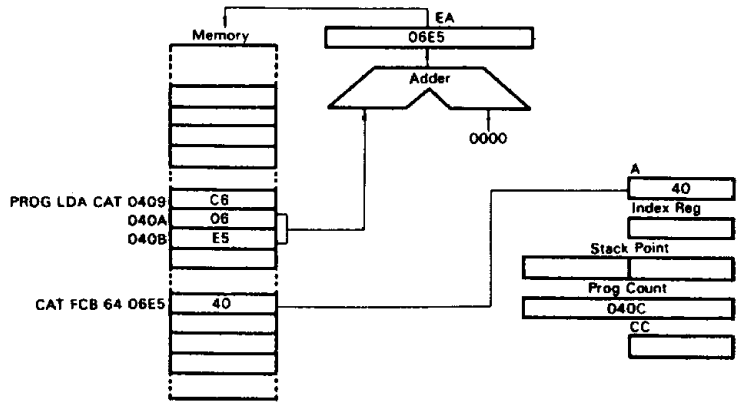


Figure 31 Example of Extended Addressing

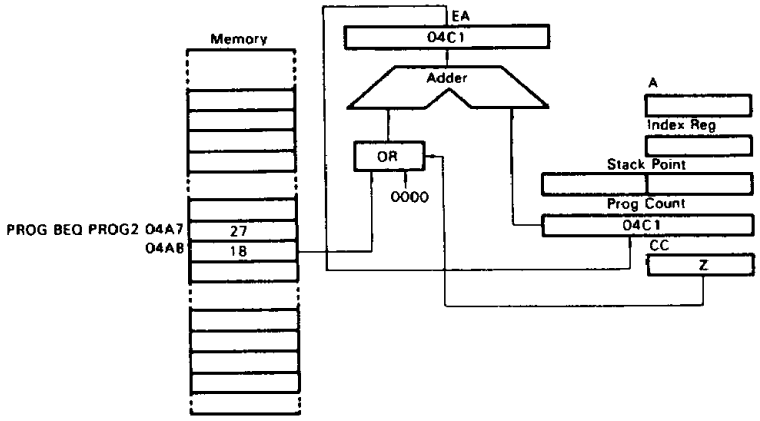


Figure 32 Example of Relative Addressing

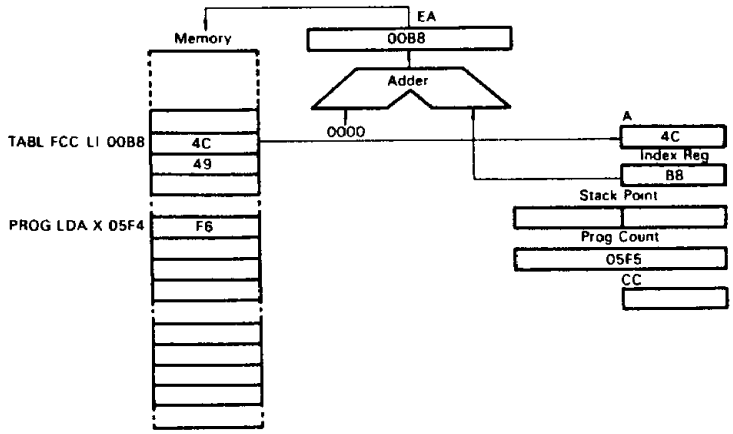


Figure 33 Example of Indexed (No Offset) Addressing

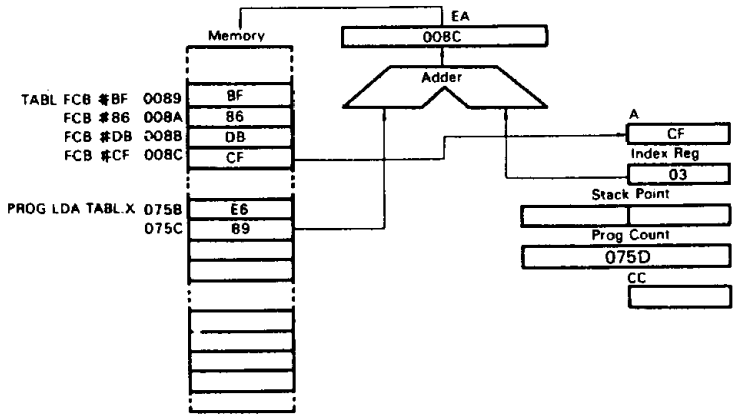


Figure 34 Example of Indexed (8-bit Offset) Addressing

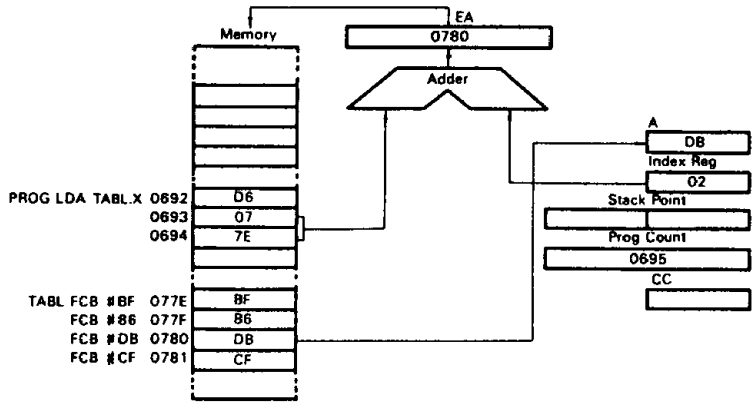


Figure 35 Example of Indexed (16-bit Offset) Addressing

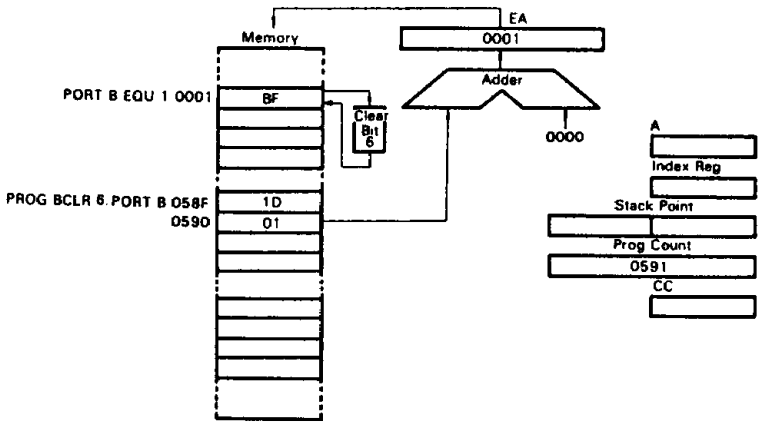


Figure 36 Example of Bit Set/Clear Addressing

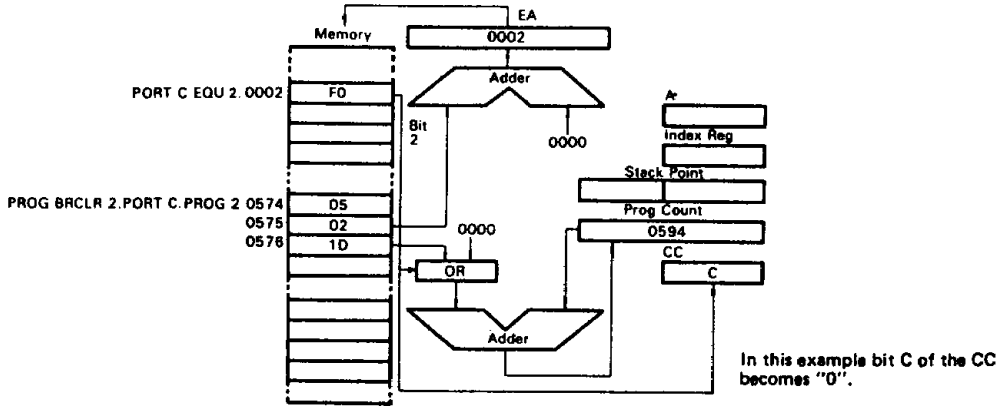


Figure 37 Example of Bit Test and Branch Addressing

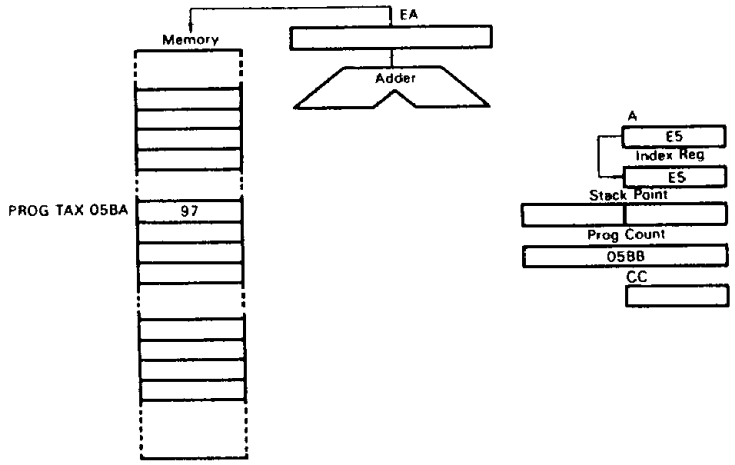


Figure 38 Example of Implied Addressing

■ INSTRUCTION SET

The HD63705V0 MCU has 62 basic instructions. They can be classified into five categories: register/memory, read-modify-write, branch, bit manipulation, and control. The details of each instruction are shown in the following tables. All the instructions in a given type are presented in individual tables.

- **Register/Memory Instructions**

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

- **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 zero (TST) instruction is an exception to the read-modify-write instructions since it does not write data. See Table 10.

- **Branch Instructions**

The branch instruction cause a branch from the program when a certain condition is met. See Table 11.

- **Bit Manipulation Instructions**

These instructions are used on any bit in the lowest 256 bytes of the memory. Two groups are available; one either sets or clears and the other performs the bit and test branch operations. See Table 12.

- **Control Instructions**

These instructions control the MCU operation during program execution. See Table 13.

- **Alphabetical Listing**

Table 14 lists the complete instruction set in alphabetical order.

- **Operation Code Map**

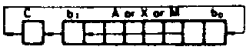
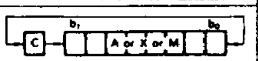
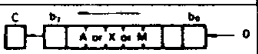
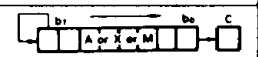
Table 15 is an operation code map for the instructions used on the MCU.

Table 9 Register/Memory Instructions

Operations	Mnemonic	Addressing Modes: Immediate			Direct			Extended			Indexed (No Offset)			Indexed (8-Bit Offset)			Indexed (16-Bit Offset)			Boolean/ Arithmetic Operation	Condition Code: H	I	N	Z	C
		OP	#	~	OP	#	~	OP	#	~	OP	#	~	OP	#	~	OP	#	~						
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	•	•	^	^	•
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	•	•	^	^	•
Store A in Memory	STA	—	—	—	B7	2	3	C7	3	4	F7	1	4	E7	2	4	D7	3	5	A→M	•	•	^	^	•
Store X in Memory	STX	—	—	—	BF	2	3	CF	3	4	FF	1	4	EF	2	4	DF	3	5	X→M	•	•	^	^	•
Add Memory to A	ADD	AB	2	2	BB	2	3	CB	3	4	FB	1	3	EB	2	4	DB	3	5	A+M→A	^	•	^	^	^
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	^	•	^	^	^
Subtract Memory	SUB	A0	2	2	B0	2	3	C0	3	4	F0	1	3	E0	2	4	D0	3	5	A−M→A	•	•	^	^	^
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	•	•	^	^	^
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	•	•	^	^	•
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	•	•	^	^	•
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	•	•	^	^	•
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	•	•	^	^	^
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	•	•	^	^	^
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	•	•	^	^	•
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 10 Read/Modify/Write Instructions

Operations	Mnemonic	Addressing Modes: Implied(A)			Implied(X)			Direct			Indexed (No Offset)			Indexed (8-Bit Offset)			Boolean/Arithmetic Operation	Condition Code: H	I	N	Z	C
		OP	#	~	OP	#	~	OP	#	~	OP	#	~	OP	#	~						
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	•	•	^	^	•
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	•	•	^	^	•
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	$\overline{A}$→A or $\overline{X}$→X or $\overline{M}$→M	•	•	^	^	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	•	•	^	^	^
Rotate Left Thru Carry	ROL	49	1	2	59	1	2	39	2	5	79	1	5	69	2	6		•	•	^	^	^
Rotate Right Thru Carry	ROR	46	1	2	56	1	2	36	2	5	76	1	5	66	2	6		•	•	^	^	^
Logical Shift Left	LSL	48	1	2	58	1	2	38	2	5	78	1	5	68	2	6		•	•	^	^	^
Logical Shift Right	LSR	44	1	2	54	1	2	34	2	5	74	1	5	64	2	6		•	•	0	^	^
Arithmetic Shift Right	ASR	47	1	2	57	1	2	37	2	5	77	1	5	67	2	6		•	•	^	^	^
Arithmetic Shift Left	ASL	48	1	2	58	1	2	38	2	5	78	1	5	68	2	6	Equal to LSL	•	•	^	^	^
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	•	•	^	^	•

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

Table 11 Branch Instructions

Operations	Mnemonic	Addressing Modes			Branch Test	Condition Code				
		Relative								
		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 12 Bit Manipulation Instructions

Operations	Mnemonic	Addressing Modes						Boolean/ Arithmetic Operation	Branch Test	Condition Code				
		Bit Set/Clear			Bit Test and Branch									
		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 13 Control Instructions

Operations	Mnemonic	Addressing Modes: Implied OP	#	~	Boolean Operation	Condition Code: 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	0 → I	●	0	●	●	●
Wait	WAIT	8F	1	4	0 → I	●	0	●	●	●

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 14 Instruction Set (in Alphabetical Order)

Mnemonic	Addressing Modes: Implied	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8-Bit)	Indexed (16-Bit)	Bit Set/ Clear	Bit Test & Branch	Condition Code: 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)	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

Table 15 Instruction Set (in Alphabetical Order)

Mnemonic	Addressing Modes										Condition Code				
	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					×						•	•	•	•	•
BRCLR										×	•	•	•	•	^
BRSET										×	•	•	•	•	^
BSET									×		•	•	•	•	•
BSR					×						•	•	•	•	•
CLC	×										•	•	•	•	0
CLI	×										•	0	•	•	•
CLR	×		×			×	×				•	•	0	1	•
CMP		×	×	×		×	×	×			•	•	^	^	^
COM	×		×			×	×				•	•	^	^	1
CPX		×	×	×		×	×	×			•	•	^	^	^
DAA	×										•	•	^	^	^
DEC	×		×			×	×				•	•	^	^	•
EOR		×	×	×		×	×	×			•	•	^	^	•
INC	×		×			×	×				•	•	^	^	•
JMP			×	×		×	×	×			•	•	•	•	•
JSR			×	×		×	×	×			•	•	•	•	•
LDA		×	×	×		×	×	×			•	•	^	^	•
LDX		×	×	×		×	×	×			•	•	^	^	•
LSL	×		×			×	×				•	•	^	^	^
LSR	×		×			×	×				•	•	0	^	^
NEG	×		×			×	×				•	•	^	^	^
NOP	×										•	•	•	•	•
ORA		×	×	×		×	×	×			•	•	^	^	•
ROL	×		×			×	×				•	•	^	^	^
ROR	×		×			×	×				•	•	^	^	^
RSP	×										•	•	•	•	•
RTI	×										?	?	?	?	?
RTS	×										•	•	•	•	•
SBC		×	×	×		×	×	×			•	•	^	^	^
SEC	×										•	•	•	•	1
SEI	×										•	1	•	•	•
STA			×	×		×	×	×			•	•	^	^	•
STOP	×										•	0	•	•	•
STX			×	×		×	×	×			•	•	^	^	•
SUB		×	×	×		×	×	×			•	•	^	^	^
SWI	×										•	1	•	•	•
TAX	×										•	•	•	•	•
TST	×		×			×	×				•	•	^	^	•
TXA	×										•	•	•	•	•
WAIT	×										•	0	•	•	•

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

Table 16 OP code Map

	Bit Manipulation		Branch	Read/Modify/Write					Control		Register/Memory						
	Test & Branch	Set/ Clear	Rel	DIR	A	X	,X1	,X0	IMP	IMP	IMM	DIR	EXT	,X2	,X1	,X0	
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	← HIGH
0	BRSET0	BSET0	BRA	NEG					RTI*	—	SUB						0
1	BRCLR0	BCLR0	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 HD63705V0.

DAA

Converts the contents of the accumulator into BCD code.

WAIT

Causes the MCU to enter the wait mode. For this mode, see "Wait Mode".

STOP

Causes the MCU to enter the stop mode. For this mode, see "Stop Mode".

■ APPLICATION NOTES

- **The EPROM Programming and Maintenance**

(1) The EPROM Programming and Data Retention

An EPROM memory cell is programmed by hot electrons injected to the floating gate with applying high voltage at the control gate and the drain. The electrons have been trapped by the potential barrier at the polysilicon-oxide (SiO_2) by which the floating gate is completely sorrounded. The programmed cell becomes a "0".

The memory cell will be discharged by;

① Exposure to ultraviolet light; discharged by photo emitting electrons (erasure principle)

② Heat; discharged by thermal emitting electrons

③ Applied with high voltage; discharged by high electric field.

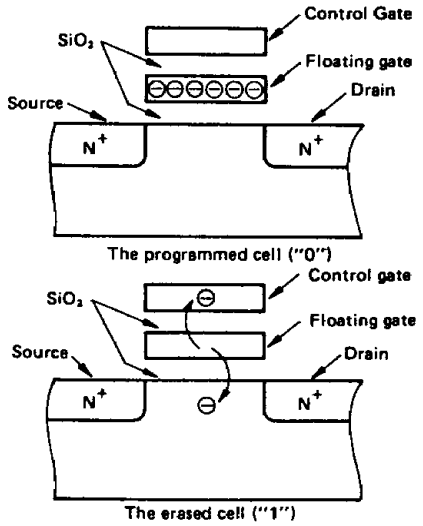


Figure 39 Cross-Section of An EPROM Memory Cell.

Charge loss from the normal cell by case ② or ③ is negligible. But if there are some defects at the SiO_2, the cell will be rapidly discharged through the defects. Such a defective part is rejected by manufacturing screenings.
The erased, or discharged, cell is a "1".

(2) Precaution of the EPROM Programming
The EPROM memory cell should be programmed with the specified voltage and timing. The higher program voltage V_{PP} or the longer program pulse width t_{PW} is applied, the more will be the quantity of electrons injected to the floating gate. However, a p-n junction will be broken permanently if V_{PP} is applied to more than maximum ratings. Especially V_{PP} overshoot of an EPROM programmer should be checked.
Negative-noise to device pins may cause a parasitic transistor effect and reduce the breakdown voltage.

(3) Precaution for using the MCU in the ceramic package with a window

Static charge on the window surface may adversely affect the function of the MCU. The charge will be caused by rubbing the window with plastics or dry cloths, or louching a charged body on it. They can be discharged by exposure to ultraviolet light for a short time. It is recommended to program the memory cell again after exposure, since the electrons trapped at the floating gate will reduce. The methods to prevent static charge on the window are follows.

① Connect the body of an operator to the ground.
② Do not rub the window with plastics or dry cloths.
③ Do not use coolant sprays which contain some ions.
④ Use a conductive opaque label.

The data stored in EPROM may be losed or the MCU may malfunction by photocurrent if the MCU is exposed to strong light like a fluorescent lamp or the sunlight. Therefore, it is recommended to cover the window with an opaque label.

Table 17 EPROM Programmers and Socket Adapters for the HD63705V0

EPROM Programmer		Socket Adapter	
Maker	Type No.	Maker	Type No.
DATA I/O (U.S.A.)	121B 29B 22B	Hitachi Ltd.	H35VSA01A
		Data I/O	HD63705V0 (for 29B)
AVAL CORP. (JAPAN)	PKW-1000	Hitachi Ltd.	H35VSA01A

■ 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.

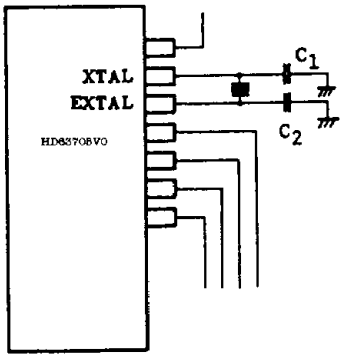


Figure 40 Design of Oscillation Circuit Board

■ PRECAUTION 2—SENDING/RECEIVING PROGRAM OF SERIAL DATA

Reading from or Writing into the SCI data register (SDR: $0012 during sending/receiving of serial data may make sending/receiving operation of SCI out of order.

■ PRECAUTION 3—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, the MCU executes the corresponding interrupt processing routine after the 4 dummy cycles.

When external interrupts (INT $\overline{INT_2}$) are held at the I bit mask, the MCU does not enter into the STOP mode by the STOP instruction execution. The MCU executes the corresponding interrupt processing routine after the 4 dummy cycles, in this case, either.

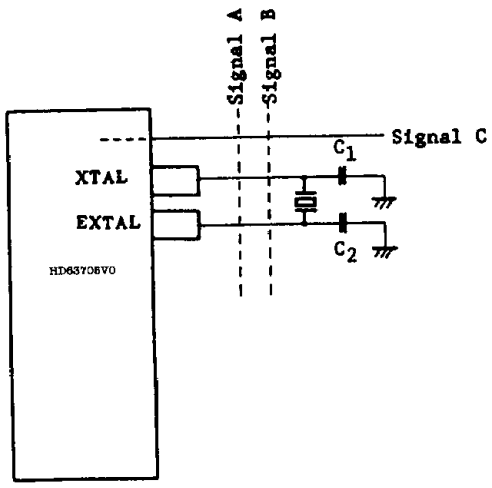


Figure 41 Example of Circuit Causing Trouble in Oscillation