

HD6303R, HD63A03R, HD63B03R

CMOS MPU (Micro Processing Unit)

The HD6303R is an 8-bit CMOS micro processing unit which has the completely compatible instruction set with the HD6301V1. 128 bytes RAM, Serial Communication Interface (SCI), parallel I/O ports and multi function timer are incorporated in the HD6303R. It is bus compatible with HMCS6800 and can be expanded up to 65k words. Like the HMCS6800 family, I/O levels is TTL compatible with +5.0V single power supply. As the HD6303R is CMOS MPU, power dissipation is extremely low. And also HD6303R has Sleep Mode and Stand-by Mode as lower power dissipation mode. Therefore, flexible low power consumption application is possible.

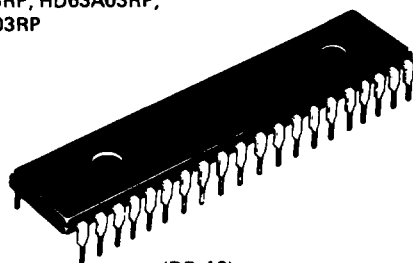
FEATURES

- Object Code Upward Compatible with the HD6800, HD6801, HD6802
- Multiplexed Bus ($D_0 \sim D_7/A_0 \sim A_7$), Non Multiplexed Bus
- Abundant On-Chip Functions Compatible with the HD6301V1; 128 Bytes RAM, 13 Parallel I/O Lines, 16-bit Timer, Serial Communication Interface (SCI)
- Low Power Consumption Mode; Sleep Mode, Stand-By Mode
- Minimum Instruction Execution Time
 $1\mu s$ ($f=1\text{MHz}$), $0.67\mu s$ ($f=1.5\text{MHz}$), $0.5\mu s$ ($f=2.0\text{MHz}$)
- Bit Manipulation, Bit Test Instruction
- Error Detecting Function; Address Trap, Op Code Trap
- Up to 65k Words Address Space

TYPE OF PRODUCTS

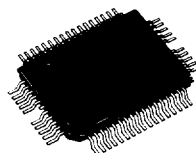
| Type No. | Bus Timing |
|----------|------------|
| HD6303R | 1.0 MHz |
| HD63A03R | 1.5 MHz |
| HD63B03R | 2.0 MHz |

HD6303RP, HD63A03RP,
HD63B03RP



(DP-40)

HD6303RF, HD63A03RF,
HD63B03RF



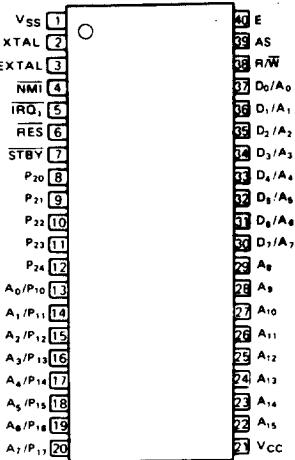
(FP-54)

HD6303RCG, HD63A03RCG,
HD63B03RCG



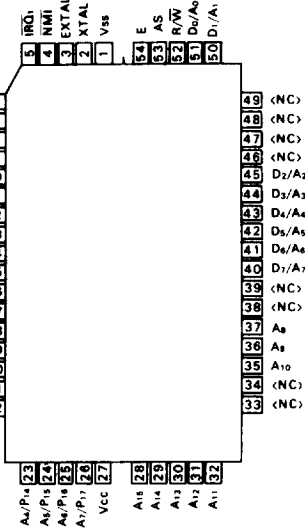
(CG-40)

- PIN ARRANGEMENT
- HD6303RP, HD63A03RP, HD63B03RP



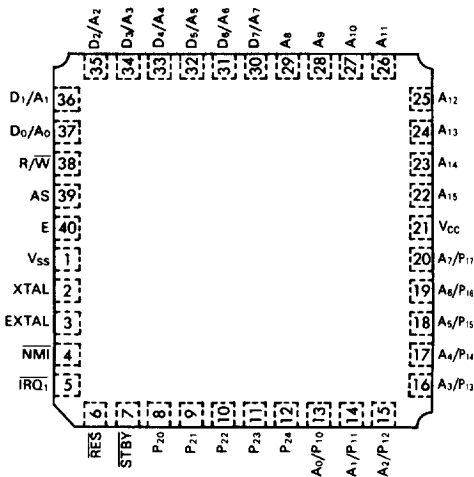
(Top View)

- HD6303RF, HD63A03RF, HD63B03RF



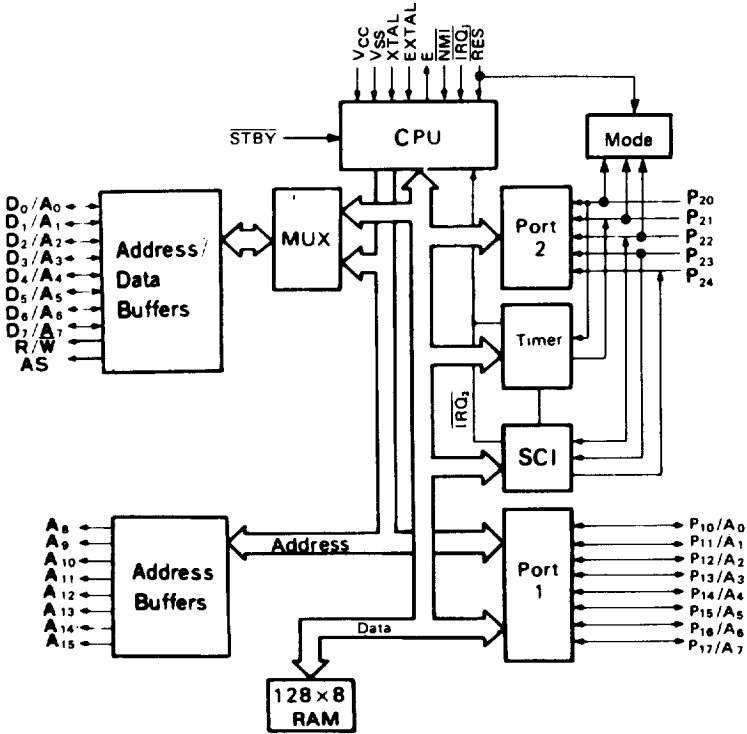
(Top View)

- HD6303RCG, HD63A03RCG, HD63B03RCG



(Top View)

■ BLOCK DIAGRAM



■ ABSOLUTE MAXIMUM RATINGS

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

(NOTE) This product has protection circuits in input terminal from high static electricity voltage and high electric field. But be careful not to apply overvoltage more than maximum ratings to these high input impedance protection circuits. To assure the normal operation, we recommend V_{in} , V_{out} : $V_{SS} \leq (V_{in} \text{ or } V_{out}) \leq V_{CC}$.

■ ELECTRICAL CHARACTERISTICS

● DC CHARACTERISTICS ($V_{CC} = 5.0V \pm 10\%$, $V_{SS} = 0V$, $T_a = 0 \sim +70^\circ C$, unless otherwise noted.)

| Item | Symbol | Test Condition | min | typ | max | Unit |
|---|---|----------------|--|--------------|--------------|---------|
| Input "High" Voltage | RES, STBY | V_{IH} | $V_{CC}-0.5$ | — | $V_{CC}+0.3$ | V |
| | EXTAL | | $V_{CC} \times 0.7$ | — | | |
| | Other Inputs | | 2.0 | — | | |
| Input "Low" Voltage | All Inputs | V_{IL} | -0.3 | — | 0.8 | V |
| Input Leakage Current | NMI, $\overline{IRQ_1}$, RES, STBY | $ I_{in} $ | $V_{in} = 0.5 \sim V_{CC}-0.5V$ | — | 1.0 | μA |
| Three State (off-state) Leakage Current | $P_{10} \sim P_{17}$, $P_{20} \sim P_{24}$, $D_0 \sim D_7$, $A_8 \sim A_{15}$ | $ I_{TSI} $ | $V_{in} = 0.5 \sim V_{CC}-0.5V$ | — | 1.0 | μA |
| Output "High" Voltage | All Outputs | V_{OH} | $I_{OH} = -200\mu A$ | 2.4 | — | V |
| | | | $I_{OH} = -10\mu A$ | $V_{CC}-0.7$ | — | V |
| Output "Low" Voltage | All Outputs | V_{OL} | $I_{OL} = 1.6mA$ | — | 0.55 | V |
| Input Capacitance | All Inputs | C_{in} | $V_{in} = 0V$, $f = 1.0MHz$, $T_a = 25^\circ C$ | — | 12.5 | pF |
| Standby Current | Non Operation | I_{CC} | — | 2.0 | 15.0 | μA |
| Current Dissipation* | | I_{CC} | Operating ($f=1MHz^{**}$) | — | 6.0 | mW |
| | | | Sleeping ($f=1MHz^{**}$) | — | 1.0 | |
| RAM Stand-By Voltage | | V_{RAM} | 2.0 | — | — | V |

* $V_{IH} \text{ min} = V_{CC}-1.0V$, $V_{IL} \text{ max} = 0.8V$

** Current Dissipation of the operating or sleeping condition is proportional to the operating frequency. So the typ. or max. values about Current Dissipations at $f = x \text{ MHz}$ operation are decided according to the following formula;

typ. value ($f = x \text{ MHz}$) = typ. value ($f = 1 \text{ MHz}$) $\times x$

max. value ($f = x \text{ MHz}$) = max. value ($f = 1 \text{ MHz}$) $\times x$

(both the sleeping and operating)

- AC CHARACTERISTICS ($V_{CC} = 5.0V \pm 10\%$, $V_{SS} = 0V$, $T_a = 0 \sim +70^\circ C$, unless otherwise noted.)

BUS TIMING

| Item | Symbol | Test Condition | HD6303R | | | HD63A03R | | | HD63B03R | | | Unit |
|-------------------------------------|------------------------------------|----------------|---------|-----|-----|----------|-----|-----|----------|-----|-----|---------|
| | | | min | typ | max | min | typ | max | min | typ | max | |
| Cycle Time | t_{cyc} | | 1 | — | 10 | 0.666 | — | 10 | 0.5 | — | 10 | μs |
| Address Strobe Pulse Width "High" | PW_{ASH} | | 220 | — | — | 150 | — | — | 110 | — | — | ns |
| Address Strobe Rise Time | t_{ASr} | | — | — | 20 | — | — | 20 | — | — | 20 | ns |
| Address Strobe Fall Time | t_{ASf} | | — | — | 20 | — | — | 20 | — | — | 20 | ns |
| Address Strobe Delay Time | t_{ASD} | | 60 | — | — | 40 | — | — | 20 | — | — | ns |
| 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} | | 450 | — | — | 300 | — | — | 220 | — | — | ns |
| Address Strobe to Enable Delay Time | t_{ASED} | | 60 | — | — | 40 | — | — | 20 | — | — | ns |
| Address Delay Time | t_{AD1} | Fig. 1 | — | — | 250 | — | — | 190 | — | — | 160 | ns |
| | t_{AD2} | | — | — | 250 | — | — | 190 | — | — | 160 | ns |
| Address Delay Time for Latch | t_{ADL} | Fig. 2 | — | — | 250 | — | — | 190 | — | — | 160 | ns |
| Data Set-up Time | Write t_{DSW} | | 230 | — | — | 150 | — | — | 100 | — | — | ns |
| | Read t_{DSR} | | 80 | — | — | 60 | — | — | 50 | — | — | ns |
| Data Hold Time | Read t_{HR} | | 0 | — | — | 0 | — | — | 0 | — | — | ns |
| | Write t_{HW} | | 20 | — | — | 20 | — | — | 20 | — | — | ns |
| Address Set-up Time for Latch | t_{ASL} | | 60 | — | — | 40 | — | — | 20 | — | — | ns |
| Address Hold Time for Latch | t_{AHL} | | 30 | — | — | 20 | — | — | 20 | — | — | ns |
| Address Hold Time | t_{AH} | | 20 | — | — | 20 | — | — | 20 | — | — | ns |
| $A_0 \sim A_7$ Set-up Time Before E | t_{ASM} | | 200 | — | — | 110 | — | — | 60 | — | — | ns |
| Peripheral Read Access Time | Non-Multiplexed Bus (t_{ACCN}) | | — | — | 650 | — | — | 395 | — | — | 270 | ns |
| | Multiplexed Bus (t_{ACCM}) | | — | — | 650 | — | — | 395 | — | — | 270 | ns |
| Oscillator stabilization Time | t_{RC} | Fig. 8 | 20 | — | — | 20 | — | — | 20 | — | — | ms |
| Processor Control Set-up Time | t_{PCS} | Fig. 9 | 200 | — | — | 200 | — | — | 200 | — | — | ns |

PERIPHERAL PORT TIMING

| Item | Symbol | Test Condition | HD6303R | | | HD63A03R | | | HD63B03R | | | Unit |
|---|----------------------|----------------|---------|-----|-----|----------|-----|-----|----------|-----|-----|------|
| | | | min | typ | max | min | typ | max | min | typ | max | |
| Peripheral Data Set-up Time | Port 1, 2 t_{PDSU} | Fig. 3 | 200 | — | — | 200 | — | — | 200 | — | — | ns |
| Peripheral Data Hold Time | Port 1, 2 t_{PDH} | Fig. 3 | 200 | — | — | 200 | — | — | 200 | — | — | ns |
| Delay Time, Enable Negative Transition to Peripheral Data Valid | Port 1, 2* t_{PWD} | Fig. 4 | — | — | 300 | — | — | 300 | — | — | 300 | ns |

* Except P_{31}

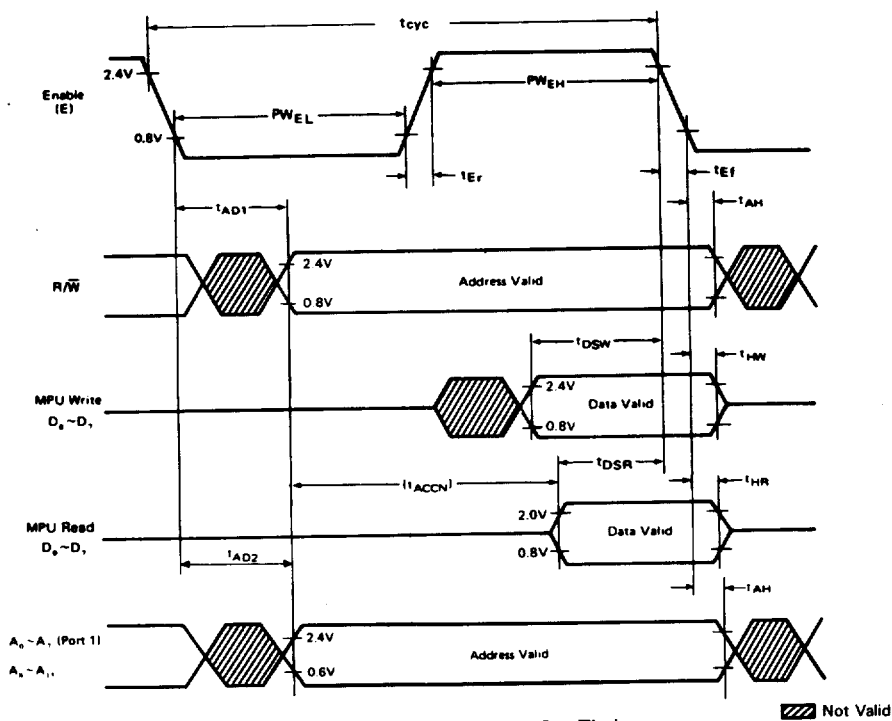


Figure 2 Non-Multiplexed Bus Timing

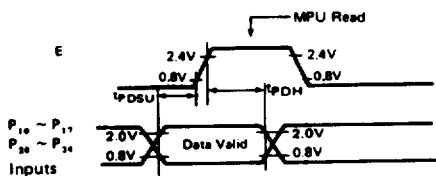
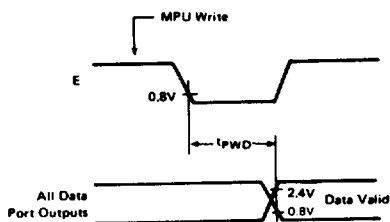


Figure 3 Port Data Set-up and Hold Times (MPU Read)



Note) Port 2: Except P₂₁
Figure 4 Port Data Delay Times (MPU Write)

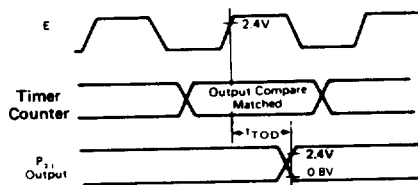


Figure 5 Timer Output Timing

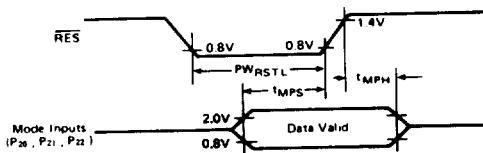
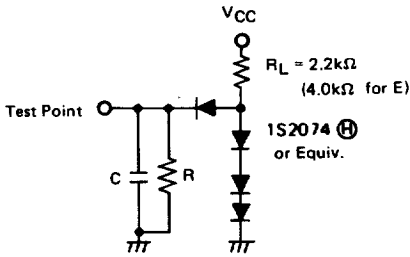


Figure 6 Mode Programming Timing



$C = 90\text{pF}$ for AS, R/W, $D_0/A_0 \sim D_7/A_7$, and $A_8 \sim A_{15}$
 $= 30\text{pF}$ for $P_{20} \sim P_{24}$ and $A_0/P_{10} \sim A_7/P_{17}$
 $= 40\text{pF}$ for E
 $R = 12\text{k}\Omega$

Figure 7 Bus Timing Test Loads (TTL Load)

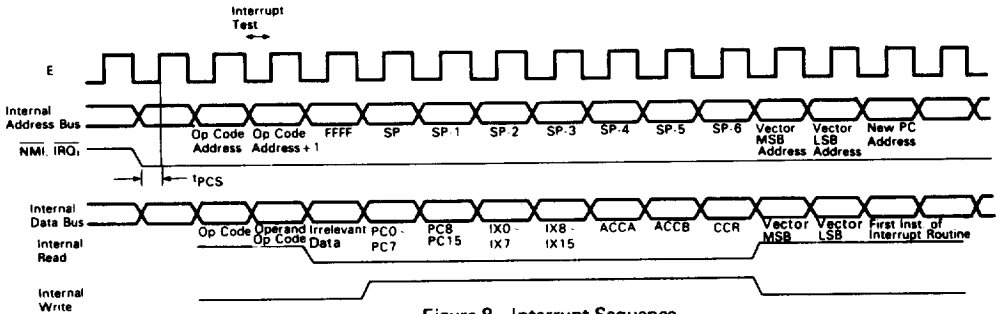


Figure 8 Interrupt Sequence

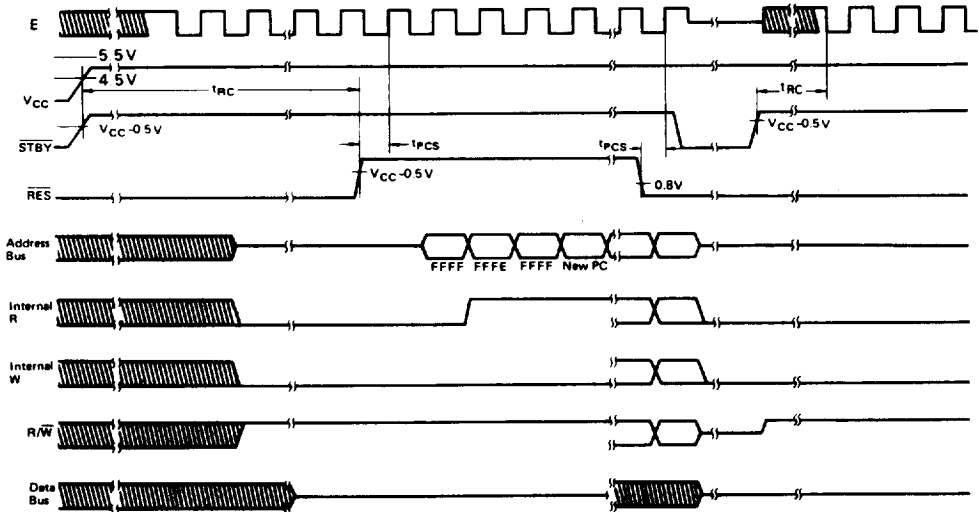


Figure 9 Reset Timing

FUNCTIONAL PIN DESCRIPTION

V_{CC} , V_{SS}

These two pins are used for power supply and GND. Recommended power supply voltage is $5\text{V} \pm 10\%$. 3 to 6V can be used for low speed operation (100 ~ 500 kHz).

XTAL, EXTERNAL

These two pins are connected with parallel resonant funda-

mental crystal, AT cut. For instance, in order to obtain the system clock 1MHz, a 4MHz resonant fundamental crystal is used because the device by 4 circuitry is included. EXTERNAL accepts an external clock input of duty 50% ($\pm 10\%$) to drive. For external driving XTAL pin should be open. An example of connection circuit is shown in Fig. 10. The crystal and capacitors should be mounted as close as possible to the pins.

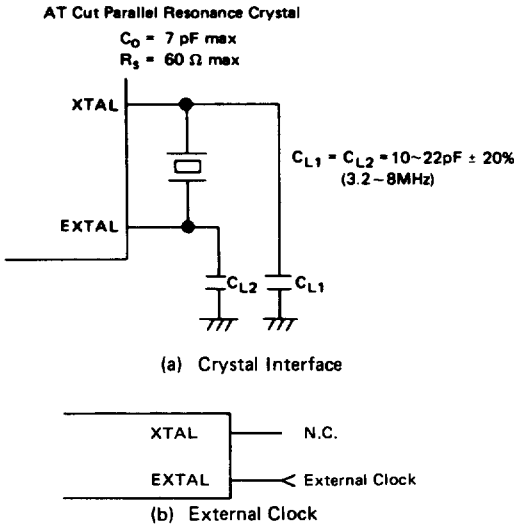


Figure 10 Connection Circuit

• Standby (STBY)

This pin is used to place the MPU in the standby mode. If this goes to "Low" level, the oscillation stops, the internal clock is tied to V_{SS} or V_{CC} and the MPU is reset. In order to retain information in RAM during standby, write "0" into RAM enable bit (RAME). RAME is bit 6 of the RAM Control Register at address \$0014. This disables the RAM, so the contents of RAM is guaranteed. For details of the standby mode, see the Standby section.

• Reset (RES)

This input is used to reset the MPU. RES must be held "Low" for at least 20ms when the power starts up. It should be noted that, before clock generator stabilize, the internal state and I/O ports are uncertain, because MPU can not be reset without clock. To reset the MPU during system operation, it must be held "Low" for at least 3 system clock cycles. From the third cycle, all address buses become "high-impedance" and it continues while RES is "Low". If RES goes to "High", CPU does the following.

- (1) I/O Port 2 bits 2,1,0 are latched into bits PC2, PC1, PC0 of program control register.
- (2) The contents of the two Start Addresses, \$FFFE, \$FFFF are brought to the program counter, from which program starts (see Table 1).
- (3) The interrupt mask bit is set. In order to have the CPU recognize the maskable interrupts $\overline{IRQ_1}$ and $\overline{IRQ_2}$, clear it before those are used.

• Enable (E)

This output pin supplies system clock. Output is a single-phase, TTL compatible and 1/4 the crystal oscillation frequency. It will drive two LS TTL load and 40pF.

• Non Maskable Interrupt (NMI)

When the falling edge of the input signal of this pin is recognized, NMI sequence starts. The current instruction is continued to complete, even if NMI signal is detected. Interrupt mask bit in Condition Code Register has no effect on NMI detection. In response to NMI interrupt, the information of

Program Counter, Index Register, Accumulators, and Condition Code Register are stored on the stack. On completion of this sequence, vectoring address \$FFFC and \$FFFD are generated to load the contents to the program counter. Then the CPU branch to a non maskable interrupt service routine.

• Interrupt Request ($\overline{IRQ_1}$)

This level sensitive input requests a maskable interrupt sequence. When $\overline{IRQ_1}$ goes to "Low", the CPU waits until it completes the current instruction that is being executed. Then, if the interrupt mask bit in Condition Code Register is not set, CPU begins interrupt sequence; otherwise, interrupt request is neglected.

Once the sequence has started, the information of Program Counter, Index Register, Accumulator, Condition Code Register are stored on the stack. Then the CPU sets the interrupt mask bit so that no further maskable interrupts may be responded.

Table 1 Interrupt Vectoring memory map

| Highest Priority | Vector | | Interrupt |
|------------------|--------|------|--|
| | MSB | LSB | |
| | FFFE | FFFF | RES |
| | FFEE | FFEF | TRAP |
| | FFFC | FFFD | NMI |
| | FFFA | FFFB | Software Interrupt (SWI) |
| | FFF8 | FFF9 | $\overline{IRQ_1}$ (or \overline{IS}) |
| | FFF6 | FFF7 | ICF (Timer Input Capture) |
| | FFF4 | FFF5 | OCF (Timer Output Compare) |
| | FFF2 | FFF3 | TOF (Timer Overflow) |
| Lowest Priority | FFF0 | FFF1 | SCI (DRIF + ORFE + TDRE) |

At the end of the cycle, the CPU generates 16 bit vectoring addresses indicating memory addresses \$FFF8 and \$FFF9, and loads the contents to the Program Counter, then branch to an interrupt service routine.

The Internal Interrupt will generate signal ($\overline{IRQ_2}$) which is quite the same as $\overline{IRQ_1}$ except that it will use the vector address \$FFFO to \$FFF7.

When $\overline{IRQ_1}$ and $\overline{IRQ_2}$ are generated at the same time, the former precede the latter. Interrupt Mask Bit in the condition code register, if being set, will keep the both interrupts off.

On occurrence of Address error or Op-code error, TRAP interrupt is invoked. This interrupt has priority next to RES. Regardless of the interrupt Mask Bit condition, the CPU will start an interrupt sequence. The vector for this interrupt will be \$FFEE, \$FFEF.

• Read/Write (R/ \overline{W})

This TTL compatible output signal indicates peripheral and memory devices whether CPU is in Read ("High"), or in Write ("Low"). The normal stand-by state is Read ("High"). Its output will drive one TTL load and 90pF.

• Address Strobe (AS)

In the multiplexed mode, address strobe signal appears at this pin. It is used to latch the lower 8 bits addresses multiplexed with data at $D_0/A_0 \sim D_7/A_7$. The 8-bit latch is controlled by address strobe as shown in Figure 15. Thereby, $D_0/A_0 \sim D_7/A_7$ can become data bus during E pulse. The timing chart of this signal is shown in Figure 1.

Address Strobe (AS) is sent out even if the internal address is accessed.

• PORTS

There are two I/O ports on HD6303R MPU (one 8-bit ports and one 5-bit port). Each port has an independent write-only data direction register to program individual I/O pins for

input or output.*

When the bit of associated Data Direction Register is "1", I/O pin is programmed for output, if "0", then programmed for an input.

There are two ports: Port 1, Port 2. Addresses of each port and associated Data Direction Register are shown in Table 2.

* Only one exception is bit 1 of Port 2 which becomes either a data input or a timer output. It cannot be used as an output port.

Table 2 Port and Data Direction Register Addresses

| Ports | Port Address | Data Direction Register Address |
|------------|--------------|---------------------------------|
| I/O Port 1 | \$0002 | \$0000 |
| I/O Port 2 | \$0003 | \$0001 |

• I/O Port 1

This is an 8-bit port, each bit being defined individually as input or outputs by associated Data Direction Register. The 8-bit output buffers have three-state capability, maintaining in high impedance state when they are used for input. In order to be read accurately, the voltage on the input lines must be more than 2.0V for logic "1" and less than 0.8V for logic "0".

These are TTL compatible. After the MPU has been reset, all I/O lines are configured as inputs in Multiplexed mode. In Non Multiplexed mode, Port 1 will be output line for lower order address lines ($A_0 \sim A_7$), which can drive one TTL load and 30 pF.

• I/O Port 2

This port has five lines, whose I/O direction depends on its data direction register. The 5-bit output buffers have three-state capability, going high impedance state when used as inputs. In order to be read accurately, the voltage on the input lines must be more than 2.0V for logic "1" and less than 0.8V for logic "0". After the MPU has been reset, I/O lines are configured as inputs. These pins on Port 2 ($P_{20} \sim P_{22}$ of the chip) are used to program the mode of operation during reset. The values of these three pins during reset are latched into the upper 3 bits (bit 7, 6 and 5) of Port 2 Data Register which is explained in the MODE SELECTION section.

In all modes, Port 2 can be configured as I/O lines. This port also provides access to the Serial I/O and the Timer. However, note that bit 1 (P_{21}) is the only pin restricted to data input or Timer output.

• BUS

• $D_0/A_0 \sim D_7/A_7$

This TTL compatible three-state buffer can drive one TTL load and 90 pF.

Non Multiplexed Mode

In this mode, these pins become only data bus ($D_0 \sim D_7$).

Multiplexed Mode

These pins become both the data bus ($D_0 \sim D_7$) and lower bits of the address bus ($A_0 \sim A_7$). An address strobe output is "High" when the address is on the pins.

• $A_8 \sim A_{15}$

Each line is TTL compatible and can drive one TTL load and

90 pF. After reset, these pins become output for upper order address lines ($A_8 \sim A_{15}$).

■ MODE SELECTION

The operation mode after the reset must be determined by the user wiring the P_{20} , P_{21} , and P_{22} externally. These three pins are lower order bits; I/O 0, I/O 1, I/O 2 of Port 2. They are latched into the control bits PC0, PC1, PC2 of I/O Port 2 register when RES goes "High". I/O Port 2 Register is shown below.

Port 2 DATA REGISTER

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------|-----|-----|-----|-------|-------|-------|-------|-------|
| \$0003 | PC2 | PC1 | PC0 | I/O 4 | I/O 3 | I/O 2 | I/O 1 | I/O 0 |

An example of external hardware used for Mode Selection is shown in Figure 11. The HD14053B is used to separate the peripheral device from the MPU during reset. It is necessary if the data may conflict between peripheral device and Mode generation circuit.

No mode can be changed through software because the bits 5, 6, and 7 of Port 2 Data Register are read-only. The mode selection of the HD6303R is shown in Table 3.

The HD6303R operates in two basic modes: (1) Multiplexed Mode, (2) Non Multiplexed Mode.

• Multiplexed Mode

The data bus and the lower order address bus are multiplexed in the $D_0/A_0 \sim D_7/A_7$ and can be separated by the Address Strobe.

Port 2 is configured for 5 parallel I/O or Serial I/O, or Timer, or any combination thereof. Port 1 is configured for 8 parallel I/O.

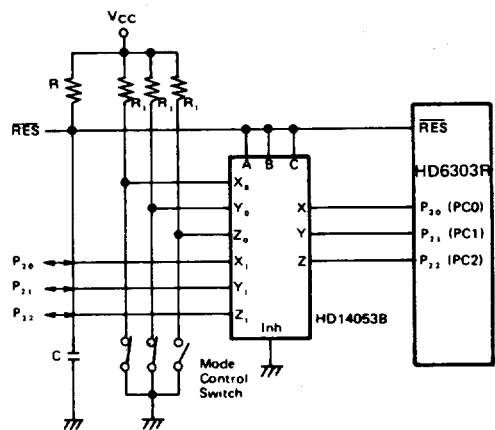
• Non Multiplexed Mode

In this mode, the HD6303R can directly address HMCS6800 peripherals with no external logic. $D_0/A_0 \sim D_7/A_7$ become a data bus and Port 1 becomes $A_0 \sim A_7$ address bus.

In this mode, the HD6303R is expandable up to 65k words with no external logic.

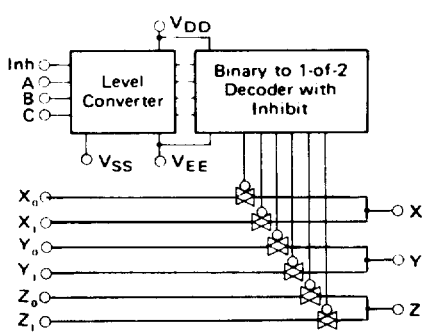
• Lower Order Address Bus Latch

Because the data bus is multiplexed with the lower order address bus in $D_0/A_0 \sim D_7/A_7$ in the multiplexed mode, address bits must be latched. It requires the 74LS373 Transparent octal D-type to latch the LSB. Latch connection of the HD6303R is shown in Figure 15.



Note 1) Figure of Multiplexed Mode
2) RC≈Reset Constant
3) R₁ = 10kΩ

Figure 11 Recommended Circuit for Mode Selection



| Truth Table | | | | | | |
|---------------|--------|---|---|----------------|----------------|----------------|
| Control input | | | | On Switch | | |
| Inhibit | Select | | | | | |
| | C | B | A | HD140538 | | |
| 0 | 0 | 0 | 0 | Z ₀ | Y ₀ | X ₀ |
| 0 | 0 | 0 | 1 | Z ₀ | Y ₀ | X ₁ |
| 0 | 0 | 1 | 0 | Z ₀ | Y ₁ | X ₀ |
| 0 | 0 | 1 | 1 | Z ₀ | Y ₁ | X ₁ |
| 0 | 1 | 0 | 0 | Z ₁ | Y ₀ | X ₀ |
| 0 | 1 | 0 | 1 | Z ₁ | Y ₀ | X ₁ |
| 0 | 1 | 1 | 0 | Z ₁ | Y ₁ | X ₀ |
| 0 | 1 | 1 | 1 | Z ₁ | Y ₁ | X ₁ |
| 1 | X | X | X | — | | |

Figure 12 HD14053B Multiplexers/De-Multiplexers

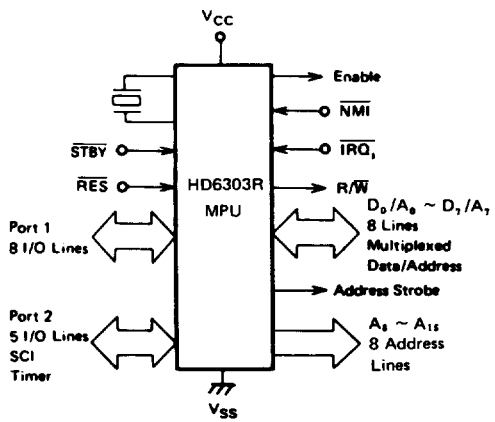


Figure 13 HD6303R MPU Multiplexed Mode

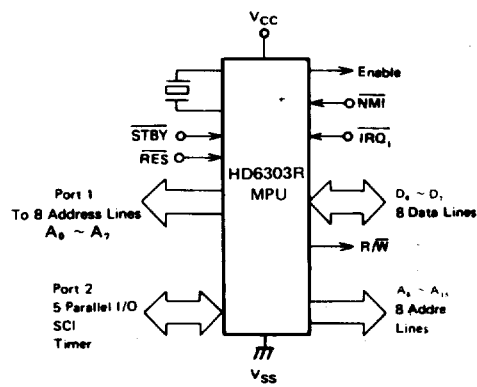


Figure 14 HD6303R MPU Non Multiplexed Mode

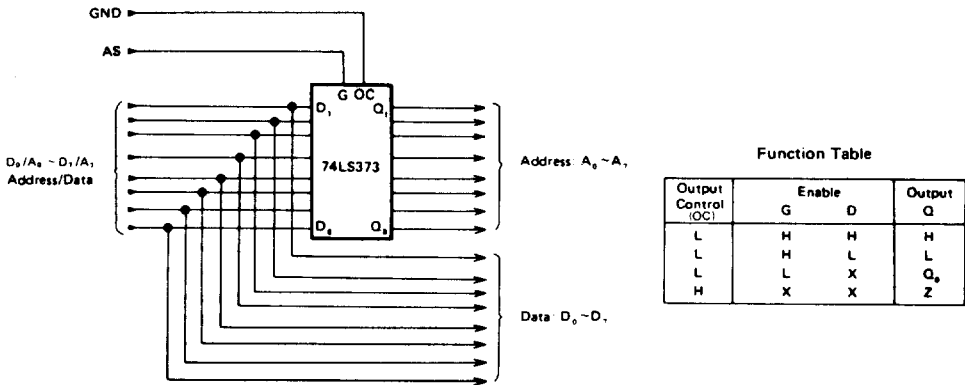


Figure 15 Latch Connection

Table 3 Mode Selection

| Operating Mode | P ₂₀ | P ₂₁ | P ₂₂ |
|----------------------|-----------------|-----------------|-----------------|
| Multiplexed Mode | L | H | L |
| | L | L | H |
| Non Multiplexed Mode | H | L | L |

L: logic "0"

H: logic "1"

MEMORY MAP

The MPU can provide up to 65k byte address space. Figure 16 shows a memory map for each operating mode. The first 32 locations of each map are for the CPU's internal register only, as shown in Table 4.

Table 4 Internal Register Area

| Register | Address |
|--|---------|
| Port 1 Data Direction Register** | 00* |
| Port 2 Data Direction Register** | 01 |
| Port 1 Data Register | 02* |
| Port 2 Data Register | 03 |
| Timer Control and Status Register | 08 |
| Counter (High Byte) | 09 |
| Counter (Low Byte) | 0A |
| Output Compare Register (High Byte) | 0B |
| Output Compare Register (Low Byte) | 0C |
| Input Capture Register (High Byte) | 0D |
| Input Capture Register (Low Byte) | 0E |
| Rate and Mode Control Register | 10 |
| Transmit/Receive Control and Status Register | 11 |
| Receive Data Register | 12 |
| Transmit Data Register | 13 |
| RAM Control Register | 14 |
| Reserved | 15-1F |

* External address in Non Multiplexed Mode

** 1 = Output, 0 = Input

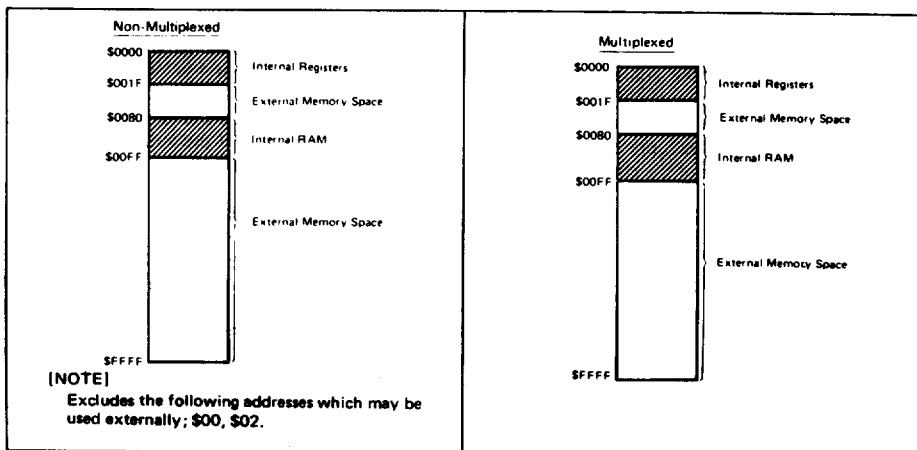


Figure 16 HD6303R Memory Maps

■ PROGRAMMABLE TIMER

The HD6303R contains 16-bit programmable timer which may be used to make measurement of input waveform. In addition to that it can generate an output waveform by itself. For both input and output waveform, the pulse width may vary from a few microseconds to several seconds.

The timer hardware consists of

- an 8-bit control and status register
- a 16-bit free running counter
- a 16-bit output compare register, and
- a 16-bit input capture register

A block diagram of the timer is shown in Figure 17.

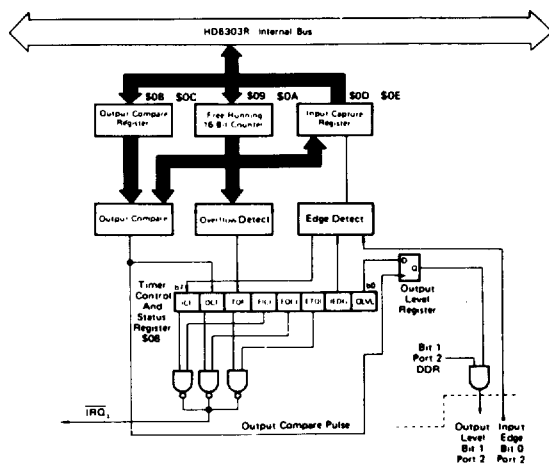


Figure 17 Programmable Timer Block Diagram

• Free Running Counter (\$0009: \$000A)

The key element in the programmable timer is a 16-bit free running counter, that is driven by an E (Enable) clock to increment its values. The counter value will be read out by the CPU software at any time with no effects on the counter. Reset will clear the counter.

When the MSB of this counter is read, the LSB is stored in temporary latch. The data is fetched from this latch by the subsequent read of LSB. Thus consistent double byte data can be read from the counter.

When the CPU writes arbitrary data to the MSB (\$09), the value of \$FFF8 is being pre-set to the counter (\$09, \$0A) regardless of the write data value. Then the CPU writes arbitrary data to the LSB (\$0A), the data is set to the "Low" byte of the counter, at the same time, the data preceedingly written in the MSB (\$09) is set to "High" byte of the counter.

When the data is written to this counter, a double byte store instruction (ex. STD) must be used. If only the MSB of counter is written, the counter is set to \$FFF8.

The counter value written to the counter using the double byte store instruction is shown in Figure 18.

To write to the counter can disturb serial operations, so it should be inhibited during using the SCI. If external clock mode is used for SCI, this will not disturb serial operations.

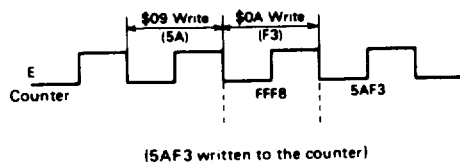


Figure 18 Counter Write Timing

• Output Compare Register (\$000B: \$000C)

This is a 16-bit read/write register which is used to control an output waveform. The contents of this register are constantly being compared with current value of the free running counter.

When the contents match with the value of the free running counter, a flag (OCF) in the timer control/status register (TCSR) is set and the current value of an output level bit (OLVL) in the TCSR is transferred to Port 2 bit 1. When bit 1 of the Port 2 data direction register is "1" (output), the OLVL value will appear on the bit 1 of Port 2. Then, the value of Output Compare Register and Output level bit may be changed for the next compare.

The output compare register is set to \$FFFF during reset.

The compare function is inhibited at the cycle of writing to the high byte of the output compare register and at the cycle just after that to ensure valid compare. It is also inhibited in same manner at writing to the free running counter.

In order to write a data to Output Compare Register, a double byte store instruction (ex. STD) must be used.

• Input Capture Register (\$000D: \$000E)

The input capture register is a 16-bit read-only register used to hold the current value of free running counter captured when the proper transition of an external input signal occurs.

The input transition change required to trigger the counter transfer is controlled by the input edge bit (IEDG).

To allow the external input signal to go in the edge detect unit, the bit of the Data Direction Register corresponding to bit 0 of Port 2 must have been cleared (to zero).

To insure input capture in all cases, the width of an input pulse requires at least 2 Enable cycles.

• Timer Control/Status Register (TCSR) (\$0008)

This is an 8-bit register. All 8-bits are readable and the lower 5 bits may be written. The upper 3 bits are read-only, indicating the timer status information as is shown below.

- (1) A proper transition has been detected on the input pin (ICF).
- (2) A match has been found between the value in the free running counter and the output compare register (OCF).
- (3) When counting up to \$0000 (TOF).

Each flag has an individual enable bit in TCSR which determines whether or not an interrupt request may occur (IRQ₂). If the I-bit in Condition Code Register has been cleared, a priority vectored address occurs corresponding to each flag. A description of each bit is as follows.

| Timer Control / Status Register | | | | | | | |
|---------------------------------|-----|-----|-----|-----|------|------|--------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ICF | OCF | TOF | EIC | EOC | ETOF | IEDG | OLVL |
| | | | | | | | \$0008 |

Bit 0 OLVL (Output Level); When a match is found in the value between the counter and the output com-

pare register, this bit is transferred to the Port 2 bit 1. If the DDR corresponding to Port 2 bit 1 is set "1", the value will appear on the output pin of Port 2 bit 1.

Bit 1 IEDG (Input Edge): This bit control which transition of an input of Port 2 bit 0 will trigger the data transfer from the counter to the input capture register. The DDR corresponding to Port 2 bit 0 must be clear in advance of using this function. When IEDG = 0, trigger takes place on a negative edge ("High"-to-"Low" transition). When IEDG = 1, trigger takes place on a positive edge ("Low"-to-"High" transition).

Bit 2 ETOI (Enable Timer Overflow Interrupt): When set, this bit enables TOF interrupt to generate the interrupt request (IRQ₂). When cleared, the interrupt is inhibited.

Bit 3 EOCI (Enable Output Compare Interrupt): When set, this bit enables OCF interrupt to generate the interrupt request (IRQ₂). When cleared, the interrupt is inhibited.

Bit 4 EICI (Enable Input Capture Interrupt): When set, this bit enables ICF interrupt to generate the interrupt request (IRQ₂). When cleared, the interrupt is inhibited.

Bit 5 TOF (Timer Over Flow Flag): This read-only bit is set at the transition of \$FFFF to \$0000 of the counter. It is cleared by CPU read of TCSR (with TOF set) followed by a CPU read of the counter (\$0009).

Bit 6 OCF (Output Compare Flag): This read-only bit is set when a match is found in the value between the output compare register and the counter. It is cleared by a read of TCSR (with OCF set) followed by a CPU write to the output compare register (\$000B or \$000C).

Bit 7 ICF (Input Capture Flag): The read-only bit is set by a proper transition on the input, and is cleared by a read of TCSR (with ICF set) followed by a CPU read of Input Capture Register (\$000D).

Reset will clear each bit of Timer Control and Status Register.

SERIAL COMMUNICATION INTERFACE

The HD6303R contains a full-duplex asynchronous Serial Communication Interface (SCI). SCI may select the several kinds of the data rate. It consists of a transmitter and a receiver which operate independently but with the same data format and the same data rate. Both of transmitter and receiver communicate with the CPU via the data bus and with the outside world through Port 2 bit 2, 3 and 4. Description of hardware, software and register is as follows.

Wake-Up Feature

In typical multiprocessor applications the software protocol will usually have the designated address at the initial byte of the message. The purpose of Wake-Up feature is to have the non-selected MPU neglect the remainder of the message. Thus the non-selected MPU can inhibit the all further interrupt process until the next message begins.

Wake-Up feature is re-enabled by a ten consecutive "1"s which indicates an idle transmit line. Therefore software protocol must put an idle period between the messages and must prevent it within the message.

With this hardware feature, the non-selected MPU is re-enabled (or "waked-up") by the next message.

Programmable Options

The HD6303R has the following programmable features.

- data format; standard mark/space (NRZ)
- clock source; external or internal
- baud rate; one of 4 rates per given E clock frequency or 1/8 of external clock
- wake-up feature; enabled or disabled
- interrupt requests; enabled or masked individually for transmitter and receiver
- clock output; internal clock enabled or disabled to Port 2 bit 2
- Port 2 (bits 3, 4); dedicated or not dedicated to serial I/O individually

Serial Communication Hardware

The serial communications hardware is controlled by 4 registers as shown in Figure 19. The registers include:

- an 8-bit control/status register
- a 4-bit rate/mode control register (write-only)
- an 8-bit read-only receive data register
- an 8-bit write-only transmit data register

Besides these 4 registers, Serial I/O utilizes Port 2 bit 3 (input) and bit 4 (output). Port 2 bit 2 can be used when an option is selected for the internal-clock-out or the external-clock-in.

Transmit/Receive Control Status Register (TRCSR)

TRCS Register consists of 8 bits which all may be read while only bits 0 to 4 may be written. The register is initialized to \$20 on RES. The bits of the TRCS Register are explained below.

Transmit / Receive Control Status Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|------|------|------|-----|----|-----|----|----|-------------|
| RDRF | ORFE | TDRE | RIE | RE | TIE | TE | WU | ADDR \$0011 |

Bit 0 WU (Wake Up): Set by software and cleared by hardware on receipt of ten consecutive "1"s. While this bit is "1", RDRF and ORFE flags are not set even if data are received or errors are detected. Therefore received data are ignored. It should be noted that RE flag must have already been set in advance of WU flag's set.

Bit 1 TE (Transmit Enable): This bit enables transmitter. When this bit is set, bit 4 of Port 2 DDR is also forced to be set. It remains set even if TE is cleared. Preamble of ten consecutive "1"s is transmitted just after this bit is set, and then transmitter becomes ready to send data. If this bit is cleared, the transmitter is disabled and serial I/O affects nothing on Port 2 bit 4.

Bit 2 TIE (Transmit Interrupt Enable): When this bit is set, TDRE (bit 5) causes an IRQ₂ interrupt. When cleared, TDRE interrupt is masked.

Bit 3 RE (Receive Enable): When set, Port 2 bit 3 can be used as an input of receive regardless of DDR value for this bit. When cleared, the receiver is disabled.

Bit 4 RIE (Receive Interrupt Enable): When this bit is set, RDRF (bit 7) or ORFE (bit 6) cause an IRQ₂ interrupt. When cleared, this interrupt is masked.

- Bit 5 TDRE (Transmit Data Register Empty);** When the data is transferred from the Transmit Data Register to Output Shift Register, this bit is set by hardware. The bit is cleared by reading the status register followed by writing the next new data into the Transmit Data Register. TDRE is initialized to 1 by RES.
- Bit 6 ORFE (Over Run Framing Error);** When overrun or framing error occurs (receive only), this bit is set by hardware. Over Run Error occurs if the attempt is made to transfer the new byte to the receive data register while the RDRF is "1". Framing Error occurs when the bit counter is not synchronized with the boundary of the byte in the re-

ceiving bit stream. When Framing Error is detected, RDRF is not set. Therefore Framing Error can be distinguished from Overrun Error. That is, if ORFE is "1" and RDRF is "1", Overrun Error is detected. Otherwise Framing Error occurs. The bit is cleared by reading the status register followed by reading the receive data register, or by RES.

Bit 7 RDRF (Receive Data Register Full); This bit is set by hardware when the data is transferred from the receive shift register to the receive data register. It is cleared by reading the status register followed by reading the receive data register, or by RES.

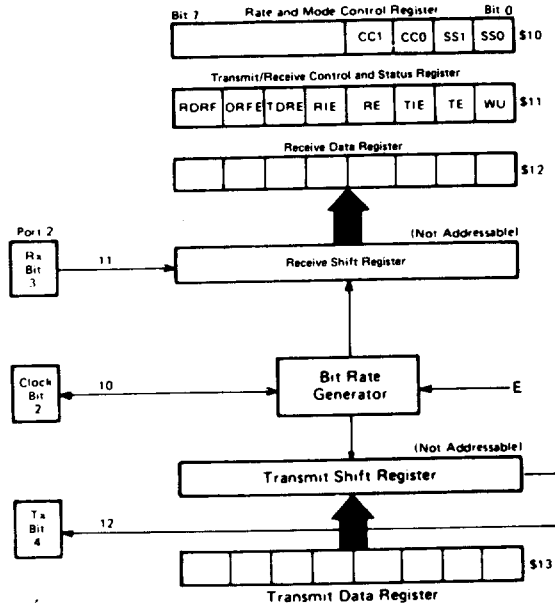


Figure 19 Serial I/O Register

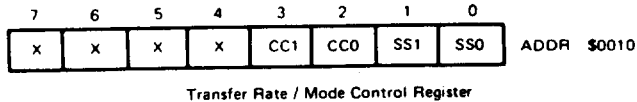


Table 5 SCI Bit Times and Transfer Rates

| SS1 : SS0 | XTAL | | 2.4576 MHz | 4.0 MHz | 4.9152MHz |
|-----------|----------|--|------------------------|-------------------------|--------------------------|
| | E | | 614.4 kHz | 1.0 MHz | 1.2288MHz |
| 0 0 | E ÷ 16 | | 26 μ s/38,400 Baud | 16 μ s/62,500 Baud | 13 μ s/76,800Baud |
| 0 1 | E ÷ 128 | | 208 μ s/4,800 Baud | 128 μ s/7812.5 Baud | 104.2 μ s/ 9,600Baud |
| 1 0 | E ÷ 1024 | | 1.67ms/600 Baud | 1.024ms/976.6 Baud | 833.3 μ s/ 1,200Baud |
| 1 1 | E ÷ 4096 | | 6.67ms/150 Baud | 4.096ms/244.1 Baud | 3.333ms/ 300Baud |

Table 6 SCI Format and Clock Source Control

| CC1: CC0 | Format | Clock Source | Port 2 Bit 2 | Port 2 Bit 3 | Port 2 Bit 4 |
|----------|--------|--------------|--------------|--------------|--------------|
| 0 0 | — | — | — | — | — |
| 0 1 | NRZ | Internal | Not Used*** | .. | .. |
| 1 0 | NRZ | Internal | Output* | .. | .. |
| 1 1 | NRZ | External | Input | .. | .. |

* Clock output is available regardless of values for bits RE and TE.

** Bit 3 is used for serial input if RE = "1" in TRCS.

Bit 4 is used for serial output if TE = "1" in TRCS.

*** This pin can be used as I/O port.

• Transfer Rate/Mode Control Register (RMCR)

The register controls the following serial I/O functions:

- Bauds rate
- data format
- clock source
- Port 2 bit 2 feature

It is 4-bit write-only register, cleared by RES. The 4 bits are considered as a pair of 2-bit fields. The lower 2 bits control the bit rate of internal clock while the upper 2 bits control the format and the clock select logic.

Bit 0 SS0 } Speed Select
Bit 1 SS1 }

These bits select the Baud rate for the internal clock. The rates selectable are function of E clock frequency of the CPU. Table 5 lists the available Baud Rates.

Bit2 CC0 } Clock Control/Format Select
Bit 3 CC1 }

They control the data format and the clock select logic. Table 6 defines the bit field.

• Internally Generated Clock

If the user wish to use externally an internal clock of the serial I/O, the following requirements should be noted.

- CC1, CC0 must be set to "10".
- The maximum clock rate must be E/16.
- The clock rate is equal to the bit rate.
- The values of RE and TE have no effect.

• Externally Generated Clock

If the user wish to supply an external clock to the Serial I/O, the following requirements should be noted.

- The CC1, CC0 must be set to "11" (See Table 6).
- The external clock must be set to 8 times of the desired baud rate.
- The maximum external clock frequency is E/2 clock.

• Serial Operations

The serial I/O hardware must be initialized by the software before operation. The sequence will be normally as follows.

- Writing the desired operation control bits of the Rate and Mode Control Register.
- Writing the desired operation control bits of the TRCS register.

If Port 2 bit 3, 4 are used for serial I/O, TE, RE bits may be kept set. When TE, RE bit are cleared during SCI operation, and subsequently set again, it should be noted that TE, RE must be kept "0" for at least one bit time of the current baud rate. If TE, RE are set again within one bit time, there may be the case where the initializing of internal function for transmitter and receiver does not take place correctly.

• Transmit Operation

Data transmission is enabled by the TE bit in the TRCS

register. When set, the output of the transmit shift register is connected with Port 2 bit 4 which is unconditionally configured as an output.

After RES, the user should initialize both the RMC register and the TRCS register for desired operation. Setting the TE bit causes a transmission of ten-bit preamble of "1"s. Following the preamble, internal synchronization is established and the transmitter is ready to operate. Then either of the following states exists.

- (1) If the transmit data register is empty (TDRE = 1), the consecutive "1"s are transmitted indicating an idle states.
- (2) If the data has been loaded into the Transmit Data Register (TDRE = 0), it is transferred to the output shift register and data transmission begins.

During the data transfer, the start bit ("0") is first transferred. Next the 8-bit data (beginning at bit 0) and finally the stop bit ("1"). When the contents of the Transmit Data Register is transferred to the output shift register, the hardware sets the TDRE flag bit: If the CPU fails to respond to the flag within the proper time, TDRE is kept set and then a continuous string of 1's is sent until the data is supplied to the data register.

• Receive Operation

The receive operation is enabled by the RE bit. The serial input is connected with Port 2 bit 3. The receiver operation is determined by the contents of the TRCS and RMC register. The received bit stream is synchronized by the first "0" (start bit). During 10-bit time, the data is strobed approximately at the center of each bit. If the tenth bit is not "1" (stop bit), the system assumes a framing error and the ORFE is set.

If the tenth bit is "1", the data is transferred to the receive data register, and the RDRF flag is set. If the tenth bit of the next data is received and still RDRF is preserved set, then ORFE is set indicating that an overrun error has occurred.

After the CPU read of the status register as a response to RDRF flag or ORFE flag, followed by the CPU read of the receive data register, RDRF or ORFE will be cleared.

■ RAM CONTROL REGISTER

The register assigned to the address \$0014 gives a status information about standby RAM.

| RAM Control Register | | | | | | | |
|----------------------|----------|-----|---|---|---|---|-----|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 0 |
| \$0014 | STBY PWR | RAM | x | x | x | x | x x |

Bit 0 Not used.

Bit 1 Not used.

Bit 2 Not used.

- Bit 3 Not used.
- Bit 4 Not used.
- Bit 5 Not used.
- Bit 6 RAM Enable.

Using this control bit, the user can disable the RAM. RAM Enable bit is set on the positive edge of $\overline{\text{RES}}$ and RAM is enabled. The program can write "1" or "0". If RAME is cleared, the RAM address becomes external address and the CPU may read the data from the outside memory.

Bit 7 Standby Bit

This bit can be read or written by the user program. It is cleared when the V_{CC} voltage is removed. Normally this bit is set by the program before going into stand-by mode. When the CPU recovers from stand-by mode, this bit should be checked. If it is "1", the data of the RAM is retained during stand-by and it is valid.

GENERAL DESCRIPTION OF INSTRUCTION SET

The HD6303R has an upward object code compatible with the HD6801 to utilize all instruction sets of the HMCS6800. The execution time of the key instruction is reduced to increase the system throughput. In addition, the bit operation instruction, the exchange instruction between the index and the accumulator, the sleep instruction are added. This section describes:

- CPU programming model (See Fig. 20)
- Addressing modes
- Accumulator and memory manipulation instructions (See Table 7)
- New instructions
- Index register and stack manipulation instructions (See Table 8)
- Jump and branch instructions (See Table 9)
- Condition code register manipulation instructions (See Table 10)
- Op-code map (See Table 11)
- Cycle-by-cycle operation (See Table 12)

CPU Programming Model

The programming model for the HD6303R is shown in Figure 20. The double accumulator is physically the same as the accumulator A concatenated with the accumulator B, so that the contents of A and B is changed with executing operation of an accumulator D.

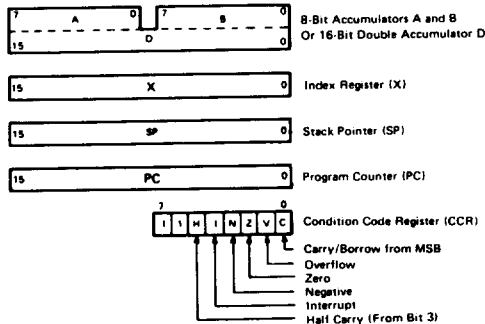


Figure 20 CPU Programming Model

CPU Addressing Modes

The HD6303R has seven address modes which depend on both of the instruction type and the code. The address mode for

every instruction is shown along with execution time given in terms of machine cycles (Table 7 to 11). When the clock frequency is 4 MHz, the machine cycle will be microseconds.

Accumulator (ACCX) Addressing

Only the accumulator (A or B) is addressed. Either accumulator A or B is specified by one-byte instructions.

Immediate Addressing

In this mode, the operand is stored in the second byte of the instruction except that the operand in LDS and LDX, etc are stored in the second and the third byte. These are two or three-byte instructions.

Direct Addressing

In this mode, the second byte of instruction indicates the address where the operand is stored. Direct addressing allows the user to directly address the lowest 256 bytes in the machine; locations zero through 255. Improved execution times are achieved by storing data in these locations. For system configuration, it is recommended that these locations should be RAM and be utilized preferably for user's data realm. These are two-byte instructions except the AIM, OIM, EIM and TIM which have three-byte.

Extended Addressing

In this mode, the second byte indicates the upper 8 bit addresses where the operand is stored, while the third byte indicates the lower 8 bits. This is an absolute address in memory. These are three-byte instructions.

Indexed Addressing

In this mode, the contents of the second byte is added to the lower 8 bits in the Index Register. For each of AIM, OIM, EIM and TIM instructions, the contents of the third byte are added to the lower 8 bits in the Index Register. In addition, the resulting "carry" is added to the upper 8 bits in the Index Register. The result is used for addressing memory. Because the modified address is held in the Temporary Address Register, there is no change to the Index Register. These are two-byte instructions but AIM, OIM, EIM, TIM have three-byte.

Implied Addressing

In this mode, the instruction itself gives the address; stack pointer, index register, etc. These are 1-byte instructions.

Relative Addressing

In this mode, the contents of the second byte is added to the lower 8 bits in the program counter. The resulting carry or borrow is added to the upper 8 bits. This helps the user to address the data within a range of -126 to +129 bytes of the current execution instruction. These are two-byte instructions.

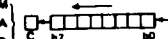
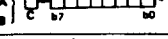
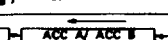
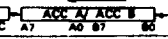
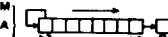
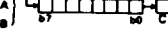

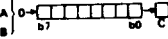
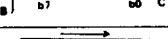
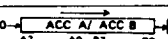
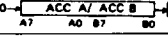
Table 7 Accumulator, Memory Manipulation Instructions

| Operations | Mnemonic | Addressing Modes | | | | | | | | | | Boolean/ Arithmetic Operation | Condition Code Register | | | | | |
|-----------------------------|----------|------------------|-----|--------|-----|-------|-----|--------|-----|---------|-----|---|----------------------------|---|---|---|---|---|
| | | IMMED | | DIRECT | | INDEX | | EXTEND | | IMPLIED | | | Register | | | | | |
| | | OP | ~ # | OP | ~ # | OP | ~ # | OP | ~ # | OP | ~ # | | 5 | 4 | 3 | 2 | 1 | 0 |
| Add | ADDA | 8B | 2 2 | 9B | 3 2 | AB | 4 2 | 8B | 4 3 | | | A + M → A | 1 | 0 | 1 | 1 | 1 | 1 |
| | ADD8 | C8 | 2 2 | 0B | 3 2 | EB | 4 2 | F8 | 4 3 | | | B + M → B | 1 | 0 | 1 | 1 | 1 | 1 |
| Add Double | ADDD | C3 | 3 3 | D3 | 4 2 | E3 | 5 2 | F3 | 5 3 | | | A : B + M : M + 1 → A : B | 0 | 0 | 1 | 1 | 1 | 1 |
| Add Accumulators | ABA | | | | | | | | | 1B | 1 1 | A + B → A | 1 | 0 | 1 | 1 | 1 | 1 |
| Add With Carry | ADCA | 89 | 2 2 | 99 | 3 2 | A9 | 4 2 | 89 | 4 3 | | | A + M + C → A | 1 | 0 | 1 | 1 | 1 | 1 |
| | ADCB | C9 | 2 2 | D9 | 3 2 | E9 | 4 2 | F9 | 4 3 | | | B + M + C → B | 1 | 0 | 1 | 1 | 1 | 1 |
| AND | ANDA | 84 | 2 2 | 94 | 3 2 | A4 | 4 2 | B4 | 4 3 | | | A · M → A | 0 | 0 | 1 | 1 | R | 0 |
| | ANDB | C4 | 2 2 | D4 | 3 2 | E4 | 4 2 | F4 | 4 3 | | | B · M → B | 0 | 0 | 1 | 1 | R | 0 |
| Bit Test | BIT A | 85 | 2 2 | 95 | 3 2 | A5 | 4 2 | B5 | 4 3 | | | A · M | 0 | 0 | 1 | 1 | R | 0 |
| | BIT B | C5 | 2 2 | D5 | 3 2 | E5 | 4 2 | F5 | 4 3 | | | B · M | 0 | 0 | 1 | 1 | R | 0 |
| Clear | CLR | | | | | 6F | 5 2 | 7F | 5 3 | | | 00 → M | 0 | 0 | R | S | R | R |
| | CLRA | | | | | | | | | 4F | 1 1 | 00 → A | 0 | 0 | R | S | R | R |
| | CLRB | | | | | | | | | 5F | 1 1 | 00 → B | 0 | 0 | R | S | R | R |
| Compare | CMPA | 81 | 2 2 | 91 | 3 2 | A1 | 4 2 | B1 | 4 3 | | | A - M | 0 | 0 | 1 | 1 | 1 | 1 |
| | CMPB | C1 | 2 2 | D1 | 3 2 | E1 | 4 2 | F1 | 4 3 | | | B - M | 0 | 0 | 1 | 1 | 1 | 1 |
| Compare Accumulators | CBA | | | | | | | | | 11 | 1 1 | A - B | 0 | 0 | 1 | 1 | 1 | 1 |
| Complement, 1's | COM | | | | | 63 | 6 2 | 73 | 6 3 | | | $\bar{M} \rightarrow M$ | 0 | 0 | 1 | 1 | R | S |
| | COMA | | | | | | | | | 43 | 1 1 | $\bar{A} \rightarrow A$ | 0 | 0 | 1 | 1 | R | S |
| | COMB | | | | | | | | | 53 | 1 1 | $\bar{B} \rightarrow B$ | 0 | 0 | 1 | 1 | R | S |
| Complement, 2's (Negate) | NEG | | | | | 60 | 6 2 | 70 | 6 3 | | | 00 - M → M | 0 | 0 | 1 | 1 | 1 | 2 |
| | NEGA | | | | | | | | | 40 | 1 1 | 00 - A → A | 0 | 0 | 1 | 1 | 1 | 2 |
| | NEGB | | | | | | | | | 50 | 1 1 | 00 - B → B | 0 | 0 | 1 | 1 | 1 | 2 |
| Decimal Adjust, A | DAA | | | | | | | | | 19 | 2 1 | Converts binary add of BCD characters into BCD format | 0 | 0 | 1 | 1 | 1 | 3 |
| Decrement | DEC | | | | | 6A | 6 2 | 7A | 6 3 | | | M - 1 → M | 0 | 0 | 1 | 1 | 4 | 0 |
| | DECA | | | | | | | | | 4A | 1 1 | A - 1 → A | 0 | 0 | 1 | 1 | 4 | 0 |
| | DECB | | | | | | | | | 5A | 1 1 | B - 1 → B | 0 | 0 | 1 | 1 | 4 | 0 |
| Exclusive OR | EORA | 88 | 2 2 | 98 | 3 2 | A8 | 4 2 | 88 | 4 3 | | | A ⊕ M → A | 0 | 0 | 1 | 1 | R | 0 |
| | EORB | C8 | 2 2 | 08 | 3 2 | E8 | 4 2 | F8 | 4 3 | | | B ⊕ M → B | 0 | 0 | 1 | 1 | R | 0 |
| Increment | INC | | | | | 6C | 6 2 | 7C | 6 3 | | | M + 1 → M | 0 | 0 | 1 | 1 | 5 | 0 |
| | INCA | | | | | | | | | 4C | 1 1 | A + 1 → A | 0 | 0 | 1 | 1 | 5 | 0 |
| | INCB | | | | | | | | | 5C | 1 1 | B + 1 → B | 0 | 0 | 1 | 1 | 5 | 0 |
| Load Accumulator | LDAA | 86 | 2 2 | 96 | 3 2 | A6 | 4 2 | 86 | 4 3 | | | M → A | 0 | 0 | 1 | 1 | R | 0 |
| | LDAB | C6 | 2 2 | D6 | 3 2 | E6 | 4 2 | F6 | 4 3 | | | M → B | 0 | 0 | 1 | 1 | R | 0 |
| Load Double Accumulator | LDD | CC | 3 3 | DC | 4 2 | EC | 5 2 | FC | 5 3 | | | M + 1 → B, M → A | 0 | 0 | 1 | 1 | R | 0 |
| Multiply Unsigned | MUL | | | | | | | | | 3D | 7 1 | A × B → A : B | 0 | 0 | 0 | 0 | 0 | 1 |
| OR, Inclusive | ORAA | 8A | 2 2 | 9A | 3 2 | AA | 4 2 | 8A | 4 3 | | | A + M → A | 0 | 0 | 1 | 1 | R | 0 |
| | ORAB | CA | 2 2 | DA | 3 2 | EA | 4 2 | FA | 4 3 | | | B + M → B | 0 | 0 | 1 | 1 | R | 0 |
| Push Data | PSHA | | | | | | | | | 36 | 4 1 | A → M _{SP} , SP - 1 → SP | 0 | 0 | 0 | 0 | 0 | 0 |
| | PSHB | | | | | | | | | 37 | 4 1 | B → M _{SP} , SP - 1 → SP | 0 | 0 | 0 | 0 | 0 | 0 |
| Pull Data | PULA | | | | | | | | | 32 | 3 1 | SP + 1 → SP, M _{SP} → A | 0 | 0 | 0 | 0 | 0 | 0 |
| | PULB | | | | | | | | | 33 | 3 1 | SP + 1 → SP, M _{SP} → B | 0 | 0 | 0 | 0 | 0 | 0 |
| Rotate Left | ROL | | | | | 69 | 6 2 | 79 | 6 3 | | | | 0 | 0 | 1 | 1 | 6 | 1 |
| | ROLA | | | | | | | | | 49 | 1 1 | | 0 | 0 | 1 | 1 | 6 | 1 |
| | ROLB | | | | | | | | | 59 | 1 1 | | 0 | 0 | 1 | 1 | 6 | 1 |
| Rotate Right | ROR | | | | | 66 | 6 2 | 76 | 6 3 | | | | 0 | 0 | 1 | 1 | 6 | 1 |
| | RORA | | | | | | | | | 46 | 1 1 | | 0 | 0 | 1 | 1 | 6 | 1 |
| | RORB | | | | | | | | | 56 | 1 1 | | 0 | 0 | 1 | 1 | 6 | 1 |

Note) Condition Code Register will be explained in Note of Table 10.

(to be continued)

Table 7 Accumulator, Memory Manipulation Instructions

| Operations | Mnemonic | Addressing Modes | | | | | | | | | | | | Boolean/ Arithmetic Operation | Condition Code Register | | | | | | |
|----------------------------------|----------|------------------|-----|--------|-----|-------|-----|--------|-----|---------|-----|-----|-----|----------------------------------|---|---|---|---|---|---|---|
| | | IMMED | | DIRECT | | INDEX | | EXTEND | | IMPLIED | | | | | | | | | | | |
| | | OP | ~ # | OP | ~ # | OP | ~ # | OP | ~ # | OP | ~ # | 5 | 4 | | 3 | 2 | 1 | 0 | | | |
| Shift Left Arithmetic | ASL | | | | | 6B | 6 2 | 7B | 6 3 | | | | | M |  | • | • | • | • | ⑥ | • |
| | ASLA | | | | | | | | | 4B | 1 1 | | | A |  | • | • | • | • | ⑥ | • |
| | ASLB | | | | | | | | | | 5B | 1 1 | | | B |  | • | • | • | • | ⑥ |
| Double Shift Left, Arithmetic | ASLD | | | | | | | | | | 0B | 1 1 | | C |  | • | • | • | • | ⑥ | • |
| Shift Right Arithmetic | ASR | | | | | 67 | 6 2 | 77 | 6 3 | | | | | M |  | • | • | • | • | ⑥ | • |
| | ASRA | | | | | | | | | 47 | 1 1 | | | A |  | • | • | • | • | ⑥ | • |
| | ASRB | | | | | | | | | | 57 | 1 1 | | | B |  | • | • | • | • | ⑥ |
| Shift Right Logical | LSR | | | | | 64 | 6 2 | 74 | 6 3 | | | | | M |  | • | • | • | R | ⑥ | • |
| | LSRA | | | | | | | | | 44 | 1 1 | | | A |  | • | • | • | R | ⑥ | • |
| | LSRB | | | | | | | | | | 54 | 1 1 | | | B |  | • | • | • | R | ⑥ |
| Double Shift Right Logical | LSRD | | | | | | | | | | 04 | 1 1 | | 0 |  | • | • | • | R | ⑥ | • |
| Store Accumulator | STAA | | | | | 97 | 3 2 | A7 | 4 2 | B7 | 4 3 | | | A → M | | • | • | • | • | R | • |
| | STAB | | | | | D7 | 3 2 | E7 | 4 2 | F7 | 4 3 | | | B → M | | • | • | • | • | R | • |
| Store Double Accumulator | STD | | | | | DD | 4 2 | ED | 5 2 | FD | 5 3 | | | A → M B → M + 1 | | • | • | • | • | R | • |
| Subtract | SUBA | 80 | 2 2 | 90 | 3 2 | A0 | 4 2 | B0 | 4 3 | | | | | A - M → A | | • | • | • | • | I | • |
| | SUBB | C0 | 2 2 | D0 | 3 2 | E0 | 4 2 | F0 | 4 3 | | | | | B - M → B | | • | • | • | • | I | • |
| Double Subtract | SUBD | 83 | 3 3 | 93 | 4 2 | A3 | 5 2 | B3 | 5 3 | | | | | A : B - M : M + 1 → A : B | | • | • | • | • | I | • |
| Subtract Accumulators | SBA | | | | | | | | | | 10 | 1 1 | | A - B → A | | • | • | • | • | I | • |
| Subtract With Carry | SBCA | 82 | 2 2 | 92 | 3 2 | A2 | 4 2 | B2 | 4 3 | | | | | A - M - C → A | | • | • | • | • | I | • |
| | SBCB | C2 | 2 2 | D2 | 3 2 | E2 | 4 2 | F2 | 4 3 | | | | | B - M - C → B | | • | • | • | • | I | • |
| Transfer Accumulators | TAB | | | | | | | | | | 16 | 1 1 | | A → B | | • | • | • | • | R | • |
| | TBA | | | | | | | | | | 17 | 1 1 | | B → A | | • | • | • | • | R | • |
| Test Zero or Minus | TST | | | | | 6D | 4 2 | 7D | 4 3 | | | | | M - 00 | | • | • | • | • | R | R |
| | TSTA | | | | | | | | | | 4D | 1 1 | | A - 00 | | • | • | • | • | R | R |
| | TSTB | | | | | | | | | | | 5D | 1 1 | | B - 00 | | • | • | • | • | R |
| And Immediate | AIM | | | | | 71 | 6 3 | 61 | 7 3 | | | | | M-IMM → M | | • | • | • | • | R | • |
| OR Immediate | OIM | | | | | 72 | 6 3 | 62 | 7 3 | | | | | M+IMM → M | | • | • | • | • | R | • |
| EOR Immediate | EIM | | | | | 75 | 6 3 | 65 | 7 3 | | | | | M⊕IMM → M | | • | • | • | • | R | • |
| Test Immediate | TIM | | | | | 7B | 4 3 | 6B | 5 3 | | | | | M-IMM | | • | • | • | • | R | • |

Note) Condition Code Register will be explained in Note of Table 10.

● New Instructions

In addition to the HD6801 Instruction Set, the HD6303R has the following new instructions:

AIM----(M) · (IMM) → (M)

Evaluates the AND of the immediate data and the memory, places the result in the memory.

OIM----(M) + (IMM) → (M)

Evaluates the OR of the immediate data and the memory, places the result in the memory.

EIM----(M) ⊕ (IMM) → (M)

Evaluates the EOR of the immediate data and the contents of memory, places the result in memory.

TIM----(M) · (IMM)

Evaluates the AND of the immediate data and the memory, changes the flag of associated condition code register

Each instruction has three bytes; the first is op-code, the second is immediate data, the third is address modifier.

XGDX--(ACCD) ↔ (IX)

Exchanges the contents of accumulator and the index register.

SLP----The MPU is brought to the sleep mode. For sleep mode, see the "sleep mode" section.

Table 8 Index Register, Stack Manipulation Instructions

| Pointer Operations | Mnemonic | Addressing Modes | | | | | | | | | | Boolean/ Arithmetic Operation | Condition Code Register | | | | | | | | | | |
|------------------------|----------|------------------|-----|--------|-----|-------|-----|--------|-----|---------|-----|----------------------------------|-------------------------|----|---|-------------|--|---|---|---|---|---|---|
| | | IMMED. | | DIRECT | | INDEX | | EXTEND | | IMPLIED | | | 5 | 4 | 3 | 2 | 1 | 0 | | | | | |
| | | OP | ~ # | OP | ~ # | OP | ~ # | OP | ~ # | OP | ~ # | | H | I | N | Z | V | C | | | | | |
| Compare Index Reg | CPX | 8C | 3 | 3 | 9C | 4 | 2 | AC | 5 | 2 | BC | 5 | 3 | | | X ← M:M + 1 | • | • | 1 | 1 | 1 | 1 | |
| Decrement Index Reg | DEX | | | | | | | | | | | | | 09 | 1 | 1 | X ← X - 1 | • | • | • | 1 | • | • |
| Decrement Stack Pntr | DES | | | | | | | | | | | | | 34 | 1 | 1 | SP ← SP - 1 | • | • | • | • | • | • |
| Increment Index Reg | INX | | | | | | | | | | | | | 08 | 1 | 1 | X ← X + 1 | • | • | • | 1 | • | • |
| Increment Stack Pntr | INS | | | | | | | | | | | | | 31 | 1 | 1 | SP ← SP + 1 | • | • | • | • | • | • |
| Load Index Reg | LDX | CE | 3 | 3 | DE | 4 | 2 | EE | 5 | 2 | FE | 5 | 3 | | | | M → X _H , (M + 1) → X _L | • | • | 1 | 1 | R | • |
| Load Stack Pntr | LDS | BE | 3 | 3 | 9E | 4 | 2 | AE | 5 | 2 | BE | 5 | 3 | | | | M → SP _H , (M + 1) → SP _L | • | • | 1 | 1 | R | • |
| Store Index Reg | STX | | | | DF | 4 | 2 | EF | 5 | 2 | FF | 5 | 3 | | | | X _H → M, X _L → (M + 1) | • | • | 1 | 1 | R | • |
| Store Stack Pntr | STS | | | | 9F | 4 | 2 | AF | 5 | 2 | BF | 5 | 3 | | | | SP _H → M, SP _L → (M + 1) | • | • | 1 | 1 | R | • |
| Index Reg → Stack Pntr | TXS | | | | | | | | | | | | | 35 | 1 | 1 | X ← SP | • | • | • | • | • | • |
| Stack Pntr → Index Reg | TSX | | | | | | | | | | | | | 30 | 1 | 1 | SP ← X | • | • | • | • | • | • |
| Add | ABX | | | | | | | | | | | | | 3A | 1 | 1 | B ← B + X | • | • | • | • | • | • |
| Push Data | PSHX | | | | | | | | | | | | | 3C | 5 | 1 | X _L → M _{sp} , SP ← SP - 1 X _H → M _{sp} , SP ← SP - 1 | • | • | • | • | • | • |
| Pull Data | PULX | | | | | | | | | | | | | 38 | 4 | 1 | SP ← SP + 1, M _{sp} → X _H SP ← SP + 1, M _{sp} → X _L | • | • | • | • | • | • |
| Exchange | XGDX | | | | | | | | | | | | | 18 | 2 | 1 | ACCD ← IX | • | • | • | • | • | • |

Note) Condition Code Register will be explained in Note of Table 10.

Table 9 Jump, Branch Instruction

| Operations | Mnemonic | Addressing Modes | | | | | | | | | | Branch Test | Condition Code Register | | | | | |
|--------------------------|----------|------------------|-----|--------|-----|-------|-----|--------|-----|---------|------|---------------------------|-------------------------|---|---|---|---|---|
| | | RELATIVE | | DIRECT | | INDEX | | EXTEND | | IMPLIED | | | 5 | 4 | 3 | 2 | 1 | 0 |
| | | OP | ~ # | OP | ~ # | OP | ~ # | OP | ~ # | OP | ~ # | | H | I | N | Z | V | C |
| Branch Always | BRA | 20 | 3 2 | | | | | | | | | None | • | • | • | • | • | • |
| Branch Never | BRN | 21 | 3 2 | | | | | | | | | None | • | • | • | • | • | • |
| Branch If Carry Clear | BCC | 24 | 3 2 | | | | | | | | | C = 0 | • | • | • | • | • | • |
| Branch If Carry Set | BCS | 25 | 3 2 | | | | | | | | | C = 1 | • | • | • | • | • | • |
| Branch If = Zero | BEQ | 27 | 3 2 | | | | | | | | | Z = 1 | • | • | • | • | • | • |
| Branch If > Zero | BGE | 2C | 3 2 | | | | | | | | | $N \oplus V = 0$ | • | • | • | • | • | • |
| Branch If > Zero | BGT | 2E | 3 2 | | | | | | | | | $Z + (N \oplus V) = 0$ | • | • | • | • | • | • |
| Branch If Higher | BHI | 22 | 3 2 | | | | | | | | | C + Z = 0 | • | • | • | • | • | • |
| Branch If < Zero | BLE | 2F | 3 2 | | | | | | | | | $Z + (N \oplus V) = 1$ | • | • | • | • | • | • |
| Branch If Lower Or Same | BLS | 23 | 3 2 | | | | | | | | | C + Z = 1 | • | • | • | • | • | • |
| Branch If < Zero | BLT | 2D | 3 2 | | | | | | | | | $N \oplus V = 1$ | • | • | • | • | • | • |
| Branch If Minus | BMI | 28 | 3 2 | | | | | | | | | N = 1 | • | • | • | • | • | • |
| Branch If Not Equal Zero | BNE | 26 | 3 2 | | | | | | | | | Z = 0 | • | • | • | • | • | • |
| Branch If Overflow Clear | BVC | 28 | 3 2 | | | | | | | | | V = 0 | • | • | • | • | • | • |
| Branch If Overflow Set | BVS | 29 | 3 2 | | | | | | | | | V = 1 | • | • | • | • | • | • |
| Branch If Plus | BPL | 2A | 3 2 | | | | | | | | | N = 0 | • | • | • | • | • | • |
| Branch To Subroutine | BSR | 8D | 5 2 | | | | | | | | | | • | • | • | • | • | • |
| Jump | JMP | | | | | 6E | 3 2 | 7E | 3 3 | | | | • | • | • | • | • | • |
| Jump To Subroutine | JSR | | | 9D | 5 2 | AD | 5 2 | BD | 6 3 | | | | • | • | • | • | • | • |
| No Operation | NOP | | | | | | | | | 01 | 1 1 | Advances Prog. Cntr. Only | • | • | • | • | • | • |
| Return From Interrupt | RTI | | | | | | | | | 3B | 10 1 | | — ① — | • | • | • | • | • |
| Return From Subroutine | RTS | | | | | | | | | 39 | 5 1 | | | • | • | • | • | • |
| Software Interrupt | SWI | | | | | | | | | 3F | 12 1 | | | • | S | • | • | • |
| Wait for Interrupt* | WAI | | | | | | | | | 3E | 9 1 | | | • | ① | • | • | • |
| Sleep | SLP | | | | | | | | | 1A | 4 1 | | | • | • | • | • | • |

Note) *WAI puts R/W high; Address Bus goes to FFFF; Data Bus goes to the three state.
Condition Code Register will be explained in Note of Table 10.

Table 10 Condition Code Register Manipulation Instructions

| Operations | Mnemonic | Addressing Modes | | | Boolean Operation | Condition Code Register | | | | | | |
|----------------------|----------|------------------|---|---|-------------------|-------------------------|---|---|---|---|---|---|
| | | IMPLIED | | | | 5 | 4 | 3 | 2 | 1 | 0 | |
| | | OP | ~ | # | | H | I | N | Z | V | C | |
| Clear Carry | CLC | 0C | 1 | 1 | 0 → C | • | • | • | • | • | • | R |
| Clear Interrupt Mask | CLI | 0E | 1 | 1 | 0 → I | • | R | • | • | • | • | |
| Clear Overflow | CLV | 0A | 1 | 1 | 0 → V | • | • | • | • | R | • | |
| Set Carry | SEC | 0D | 1 | 1 | 1 → C | • | • | • | • | • | S | |
| Set Interrupt Mask | SEI | 0F | 1 | 1 | 1 → I | • | S | • | • | • | • | |
| Set Overflow | SEV | 0B | 1 | 1 | 1 → V | • | • | • | • | S | • | |
| Accumulator A → CCR | TAP | 06 | 1 | 1 | A → CCR | 10 | | | | | | |
| CCR → Accumulator A | TPA | 07 | 1 | 1 | CCR → A | • | • | • | • | • | • | |

[NOTE 1] Condition Code Register Notes: (Bit set if test is true and cleared otherwise)

- 1: (Bit V) Test: Result = 10000000?
- 2: (Bit C) Test: Result = 00000000?
- 3: (Bit C) Test: BCD Character of high-order byte greater than 9? (Not cleared if previously set)
- 4: (Bit V) Test: Operand = 10000000 prior to execution?
- 5: (Bit V) Test: Operand = 01111111 prior to execution?
- 6: (Bit V) Test: Set equal to N=C=1 after the execution of instructions
- 7: (Bit N) Test: Result less than zero? (Bit 15=1)
- 8: (All Bit) Load Condition Code Register from Stack.
- 9: (Bit I) Set when interrupt occurs. If previously set, a Non-Maskable Interrupt is required to exit the wait state.
- 10: (All Bit) Set according to the contents of Accumulator A.
- 11: (Bit C) Result of Multiplication Bit 7=1 of ACCB?

[NOTE 2] CLI instruction and interrupt.

If interrupt mask-bit is set (I="1") and interrupt is requested ($\overline{IRQ_1}$ = "0" or $\overline{IRQ_2}$ = "0"), and then CLI instruction is executed, the CPU responds as follows.

- ① The next instruction of CLI is one-machine cycle instruction.
Subsequent two instructions are executed before the interrupt is responded.
That is, the next and the next of the next instruction are executed.
- ② The next instruction of CLI is two-machine cycle (or more) instruction.
Only the next instruction is executed and then the CPU jump to the interrupt routine.
Even if TAP instruction is used, instead of CLI, the same thing occurs.

Table 11 OP-Code Map

| OP CODE | | ACC A | | | | | | | | ACCA or SP | | | | ACCB or X | | | | |
|---------|----|-------|------|------|------|------|------|------|------|------------|-----|------|-----|-----------|------|-----|-----|---|
| | HI | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | IMM | DIR | IND | EXT | IMM | DIR | IND | EXT | |
| LO | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F | |
| 0000 | 0 | | SBA | BRA | TSX | | NEG | | | | | | | SUB | | | | 0 |
| 0001 | 1 | NOP | CBA | BRN | INS | | | | AIM | | | | | CMP | | | | 1 |
| 0010 | 2 | | | BHI | PULA | | | | OIM | | | | | SBC | | | | 2 |
| 0011 | 3 | | | | BLS | PULB | | | COM | | | SUBD | | | ADDD | | | 3 |
| 0100 | 4 | LSRD | | | BCC | DES | | | LSR | | | | | AND | | | | 4 |
| 0101 | 5 | ASLD | | | BCS | TXS | | | EIM | | | | | BIT | | | | 5 |
| 0110 | 6 | TAP | TAB | BNE | PSHA | | | | ROR | | | | | LDA | | | | 6 |
| 0111 | 7 | TPA | TBA | BEQ | PSHB | | | ASR | | | STA | | | | STA | | | 7 |
| 1000 | 8 | INX | XGDX | BVC | PULX | | | ASL | | | | | | EOR | | | | 8 |
| 1001 | 9 | DEX | DAA | BVS | RTS | | | | ROL | | | | | ADC | | | | 9 |
| 1010 | A | CLV | SLP | BPL | ABX | | | | DEC | | | | | ORA | | | | A |
| 1011 | B | SEV | ABA | BMI | RTI | | | | TIM | | | | | ADD | | | | B |
| 1100 | C | CLC | | BGE | PSHX | | | | INC | | | CPX | | | LDD | | | C |
| 1101 | D | SEC | | | BLT | MUL | | | TST | | BSR | | JSR | | | STD | | D |
| 1110 | E | CLI | | BGT | WAI | | | | JMP | | | LDS | | | LDX | | | E |
| 1111 | F | SEI | | | BLE | SWI | | | CLR | | | | STS | | | STX | | F |
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F | |

UNDEFINED OP CODE 

* Only for instructions of AIM, OIM, EIM, TIM

● Instruction Execution Cycles

In the HMCS6800 series, the execution cycle of each instruction is the number of cycles between the start of the current instruction fetch and just before the start of the subsequent instruction fetch.

The HD6303R uses a mechanism of the pipeline control for the instruction fetch and the subsequent instruction fetch is performed during the current instruction being executed.

Therefore, the method to count instruction cycles used in the HMCS6800 series cannot be applied to the instruction cycles such as MULT, PULL, DAA and XGDX in the HD6303R.

Table 12 provides the information about the relationship among each data on the Address Bus, Data Bus, and R/W status in cycle-by-cycle basis during the execution of each instruction.

Table 12 Cycle-by-Cycle Operation

| Address Mode & Instructions | | Cycles | Cycle # | Address Bus | R W | Data Bus |
|-----------------------------|------|--------|---------|-----------------------|-----|---------------------------|
| IMMEDIATE | | | | | | |
| ADC | ADD | 2 | 1 | Op Code Address+1 | 1 | Operand Data |
| AND | BIT | | 2 | Op Code Address+2 | 1 | Next Op Code |
| CMP | EOR | | | | | |
| LDA | ORA | | | | | |
| SBC | SUB | | | | | |
| ADDD | CPX | 3 | 1 | Op Code Address+1 | 1 | Operand Data (MSB) |
| LDD | LDS | | 2 | Op Code Address+2 | 1 | Operand Data (LSB) |
| LDX | SUBD | | 3 | Op Code Address+3 | 1 | Next Op Code |
| DIRECT | | | | | | |
| ADC | ADD | 3 | 1 | Op Code Address+1 | 1 | Address of Operand (LSB) |
| AND | BIT | | 2 | Address of Operand | 1 | Operand Data |
| CMP | EOR | | 3 | Op Code Address+2 | 1 | Next Op Code |
| LDA | ORA | | | | | |
| SBC | SUB | | | | | |
| STA | | 3 | 1 | Op Code Address+1 | 1 | Destination Address |
| | | | 2 | Destination Address | 0 | Accumulator Data |
| | | | 3 | Op Code Address+2 | 1 | Next Op Code |
| ADDD | CPX | 4 | 1 | Op Code Address+1 | 1 | Address of Operand (LSB) |
| LDD | LDS | | 2 | Address of Operand | 1 | Operand Data (MSB) |
| LDX | SUBD | | 3 | Address of Operand+1 | 1 | Operand Data (LSB) |
| | | | 4 | Op Code Address+2 | 1 | Next Op Code |
| STD | STS | 4 | 1 | Op Code Address+1 | 1 | Destination Address (LSB) |
| STX | | | 2 | Destination Address | 0 | Register Data (MSB) |
| | | | 3 | Destination Address+1 | 0 | Register Data (LSB) |
| | | | 4 | Op Code Address+2 | 1 | Next Op Code |
| JSR | | 5 | 1 | Op Code Address+1 | 1 | Jump Address (LSB) |
| | | | 2 | FFFF | 1 | Restart Address (LSB) |
| | | | 3 | Stack Pointer | 0 | Return Address (LSB) |
| | | | 4 | Stack Pointer-1 | 0 | Return Address (MSB) |
| | | | 5 | Jump Address | 1 | First Subroutine Op Code |
| TIM | | 4 | 1 | Op Code Address+1 | 1 | Immediate Data |
| | | | 2 | Op Code Address+2 | 1 | Address of Operand (LSB) |
| | | | 3 | Address of Operand | 1 | Operand Data |
| | | | 4 | Op Code Address+3 | 1 | Next Op Code |
| AIM | EIM | 6 | 1 | Op Code Address+1 | 1 | Immediate Data |
| OIM | | | 2 | Op Code Address+2 | 1 | Address of Operand (LSB) |
| | | | 3 | Address of Operand | 1 | Operand Data |
| | | | 4 | FFFF | 1 | Restart Address (LSB) |
| | | | 5 | Address of Operand | 0 | New Operand Data |
| | | | 6 | Op Code Address+3 | 1 | Next Op Code |

— Continued —

Table 12 Cycle-by-Cycle Operation (Continued)

| Address Mode & Instructions | | Cycles | Cycle # | Address Bus | R/W | Data Bus |
|-----------------------------|-----|--------|---------|-------------------|-----|-------------------------------|
| INDEXED | | | | | | |
| JMP | | 3 | 1 | Op Code Address+1 | 1 | Offset |
| | | | 2 | FFFF | 1 | Restart Address (LSB) |
| | | | 3 | Jump Address | 1 | First Op Code of Jump Routine |
| ADC | ADD | 4 | 1 | Op Code Address+1 | 1 | Offset |
| AND | BIT | | 2 | FFFF | 1 | Restart Address (LSB) |
| CMP | EOR | | 3 | IX+Offset | 1 | Operand Data |
| LDA | ORA | | 4 | Op Code Address+2 | 1 | Next Op Code |
| SBC | SUB | | | | | |
| TST | | | | | | |
| STA | | 4 | 1 | Op Code Address+1 | 1 | Offset |
| | | | 2 | FFFF | 1 | Restart Address (LSB) |
| | | | 3 | IX+Offset | 0 | Accumulator Data |
| | | | 4 | Op Code Address+2 | 1 | Next Op Code |
| ADDD | | 5 | 1 | Op Code Address+1 | 1 | Offset |
| CPX | LDD | | 2 | FFFF | 1 | Restart Address (LSB) |
| LDS | LDX | | 3 | IX+Offset | 1 | Operand Data (MSB) |
| SUBD | | | 4 | IX+Offset+1 | 1 | Operand Data (LSB) |
| | | | 5 | Op Code Address+2 | 1 | Next Op Code |
| STD | STS | 5 | 1 | Op Code Address+1 | 1 | Offset |
| STX | | | 2 | FFFF | 1 | Restart Address (LSB) |
| | | | 3 | IX+Offset | 0 | Register Data (MSB) |
| | | | 4 | IX+Offset+1 | 0 | Register Data (LSB) |
| | | | 5 | Op Code Address+2 | 1 | Next Op Code |
| JSR | | 5 | 1 | Op Code Address+1 | 1 | Offset |
| | | | 2 | FFFF | 1 | Restart Address (LSB) |
| | | | 3 | Stack Pointer | 0 | Return Address (LSB) |
| | | | 4 | Stack Pointer-1 | 0 | Return Address (MSB) |
| | | | 5 | IX+Offset | 1 | First Subroutine Op Code |
| ASL | ASR | 6 | 1 | Op Code Address+1 | 1 | Offset |
| COM | DEC | | 2 | FFFF | 1 | Restart Address (LSB) |
| INC | LSR | | 3 | IX+Offset | 1 | Operand Data |
| NEG | ROL | | 4 | FFFF | 1 | Restart Address (LSB) |
| ROR | | | 5 | IX+Offset | 0 | New Operand Data |
| | | | 6 | Op Code Address+1 | 1 | Next Op Code |
| TIM | | 5 | 1 | Op Code Address+1 | 1 | Immediate Data |
| | | | 2 | Op Code Address+2 | 1 | Offset |
| | | | 3 | FFFF | 1 | Restart Address (LSB) |
| | | | 4 | IX+Offset | 1 | Operand Data |
| | | | 5 | Op Code Address+3 | 1 | Next Op Code |
| CLR | | 5 | 1 | Op Code Address+1 | 1 | Offset |
| | | | 2 | FFFF | 1 | Restart Address (LSB) |
| | | | 3 | IX+Offset | 1 | Operand Data |
| | | | 4 | IX+Offset | 0 | 00 |
| | | | 5 | Op Code Address+2 | 1 | Next Op Code |
| AIM | EIM | 7 | 1 | Op Code Address+1 | 1 | Immediate Data |
| OIM | | | 2 | Op Code Address+2 | 1 | Offset |
| | | | 3 | FFFF | 1 | Restart Address (LSB) |
| | | | 4 | IX+Offset | 1 | Operand Data |
| | | | 5 | FFFF | 1 | Restart Address (LSB) |
| | | | 6 | IX+Offset | 0 | New Operand Data |
| | | | 7 | Op Code Address+3 | 1 | Next Op Code |

- Continued -

Table 12 Cycle-by-Cycle Operation (Continued)

| Address Mode & Instructions | Cycles | Cycle # | Address Bus | R/ \overline{W} | Data Bus |
|-----------------------------|--------|---------|-----------------------|-------------------|---------------------------|
| EXTEND | | | | | |
| JMP | 3 | 1 | Op Code Address+1 | 1 | Jump Address (MSB) |
| | | 2 | Op Code Address+2 | 1 | Jump Address (LSB) |
| | | 3 | Jump Address | 1 | Next Op Code |
| ADC ADD TST | 4 | 1 | Op Code Address+1 | 1 | Address of Operand (MSB) |
| AND BIT | | 2 | Op Code Address+2 | 1 | Address of Operand (LSB) |
| CMP EOR | | 3 | Address of Operand | 1 | Operand Data |
| LDA ORA | | 4 | Op Code Address+3 | 1 | Next Op Code |
| SBC SUB | | | | | |
| STA | 4 | 1 | Op Code Address+1 | 1 | Destination Address (MSB) |
| | | 2 | Op Code Address+2 | 1 | Destination Address (LSB) |
| | | 3 | Destination Address | 0 | Accumulator Data |
| | | 4 | Op Code Address+3 | 1 | Next Op Code |
| ADDD | 5 | 1 | Op Code Address+1 | 1 | Address of Operand (MSB) |
| CPX LDD | | 2 | Op Code Address+2 | 1 | Address of Operand (LSB) |
| LDS LDX | | 3 | Address of Operand | 1 | Operand Data (MSB) |
| SUBD | | 4 | Address of Operand+1 | 1 | Operand Data (LSB) |
| | | 5 | Op Code Address+3 | 1 | Next Op Code |
| STD STS | 5 | 1 | Op Code Address+1 | 1 | Destination Address (MSB) |
| STX | | 2 | Op Code Address+2 | 1 | Destination Address (LSB) |
| | | 3 | Destination Address | 0 | Register Data (MSB) |
| | | 4 | Destination Address+1 | 0 | Register Data (LSB) |
| | | 5 | Op Code Address+3 | 1 | Next Op Code |
| JSR | 6 | 1 | Op Code Address+1 | 1 | Jump Address (MSB) |
| | | 2 | Op Code Address+2 | 1 | Jump Address (LSB) |
| | | 3 | FFFF | 1 | Restart Address (LSB) |
| | | 4 | Stack Pointer | 0 | Return Address (LSB) |
| | | 5 | Stack Pointer-1 | 0 | Return Address (MSB) |
| | | 6 | Jump Address | 1 | First Subroutine Op Code |
| ASL ASR | 6 | 1 | Op Code Address+1 | 1 | Address of Operand (MSB) |
| COM DEC | | 2 | Op Code Address+2 | 1 | Address of Operand (LSB) |
| INC LSR | | 3 | Address of Operand | 1 | Operand Data |
| NEG ROL | | 4 | FFFF | 1 | Restart Address (LSB) |
| ROR | | 5 | Address of Operand | 0 | New Operand Data |
| | | 6 | Op Code Address+3 | 1 | Next Op Code |
| CLR | 5 | 1 | Op Code Address+1 | 1 | Address of Operand (MSB) |
| | | 2 | Op Code Address+2 | 1 | Address of Operand (LSB) |
| | | 3 | Address of Operand | 1 | Operand Data |
| | | 4 | Address of Operand | 0 | 00 |
| | | 5 | Op Code Address+3 | 1 | Next Op Code |

- Continued -

Table 12 Cycle-by-Cycle Operation (Continued)

| Address Mode & Instructions | | Cycles | Cycle # | Address Bus | R/W | Data Bus |
|-----------------------------|------|--------|---------------------------------|--|---------------------------------|--|
| IMPLIED | | | | | | |
| ABA | ABX | 1 | 1 | Op Code Address + 1 | 1 | Next Op Code |
| ASL | ASLD | | | | | |
| ASR | CBA | | | | | |
| CLC | CLI | | | | | |
| CLR | CLV | | | | | |
| COM | DEC | | | | | |
| DES | DEX | | | | | |
| INC | INS | | | | | |
| INX | LSR | | | | | |
| LSRD | ROL | | | | | |
| ROR | NOP | | | | | |
| SBA | SEC | | | | | |
| SEI | SEV | | | | | |
| TAB | TAP | | | | | |
| TBA | TPA | | | | | |
| TST | TSX | | | | | |
| TXS | | | | | | |
| DAA | XGDX | 2 | 1 2 | Op Code Address + 1 FFFF | 1 1 | Next Op Code Restart Address (LSB) |
| PULA | PULB | 3 | 1 2 3 | Op Code Address + 1 FFFF Stack Pointer + 1 | 1 1 1 | Next Op Code Restart Address (LSB) Data from Stack |
| PSHA | PSHB | 4 | 1 2 3 4 | Op Code Address + 1 FFFF Stack Pointer Op Code Address + 1 | 1 1 0 1 | Next Op Code Restart Address (LSB) Accumulator Data Next Op Code |
| PULX | | 4 | 1 2 3 4 | Op Code Address + 1 FFFF Stack Pointer + 1 Stack Pointer + 2 | 1 1 1 1 | Next Op Code Restart Address (LSB) Data from Stack (MSB) Data from Stack (LSB) |
| PSHX | | 5 | 1 2 3 4 5 | Op Code Address + 1 FFFF Stack Pointer Stack Pointer - 1 Op Code Address + 1 | 1 1 0 0 1 | Next Op Code Restart Address (LSB) Index Register (LSB) Index Register (MSB) Next Op Code |
| RTS | | 5 | 1 2 3 4 5 | Op Code Address + 1 FFFF Stack Pointer + 1 Stack Pointer + 2 Return Address | 1 1 1 1 1 | Next Op Code Restart Address (LSB) Return Address (MSB) Return Address (LSB) First Op Code of Return Routine |
| MUL | | 7 | 1 2 3 4 5 6 7 | Op Code Address + 1 FFFF FFFF FFFF FFFF FFFF FFFF | 1 1 1 1 1 1 1 | Next Op Code Restart Address (LSB) Restart Address (LSB) Restart Address (LSB) Restart Address (LSB) Restart Address (LSB) Restart Address (LSB) |

- Continued -

Table 12 Cycle-by-Cycle Operation (Continued)

| Address Mode & Instructions | Cycles | Cycle # | Address Bus | R/ \overline{W} | Data Bus |
|-----------------------------|--------|---------|------------------------|-------------------|---------------------------------|
| IMPLIED | | | | | |
| WAI | 9 | 1 | Op Code Address+1 | 1 | Next Op Code |
| | | 2 | FFFF | 1 | Restart Address (LSB) |
| | | 3 | Stack Pointer | 0 | Return Address (LSB) |
| | | 4 | Stack Pointer-1 | 0 | Return Address (MSB) |
| | | 5 | Stack Pointer-2 | 0 | Index Register (LSB) |
| | | 6 | Stack Pointer-3 | 0 | Index Register (MSB) |
| | | 7 | Stack Pointer-4 | 0 | Accumulator A |
| | | 8 | Stack Pointer-5 | 0 | Accumulator B |
| | | 9 | Stack Pointer-6 | 0 | Conditional Code Register |
| RTI | 10 | 1 | Op Code Address+1 | 1 | Next Op Code |
| | | 2 | FFFF | 1 | Restart Address (LSB) |
| | | 3 | Stack Pointer | 1 | Conditional Code Register |
| | | 4 | Stack Pointer+1 | 1 | Accumulator B |
| | | 5 | Stack Pointer+2 | 1 | Accumulator A |
| | | 6 | Stack Pointer+3 | 1 | Index Register (MSB) |
| | | 7 | Stack Pointer+4 | 1 | Index Register (LSB) |
| | | 8 | Stack Pointer+5 | 1 | Return Address (MSB) |
| | | 9 | Stack Pointer+6 | 1 | Return Address (LSB) |
| | | 10 | Return Address | 1 | First Op Code of Return Routine |
| SWI | 12 | 1 | Op Code Address+1 | 1 | Next Op Code |
| | | 2 | FFFF | 1 | Restart Address (LSB) |
| | | 3 | Stack Pointer | 0 | Return Address (LSB) |
| | | 4 | Stack Pointer - 1 | 0 | Return Address (MSB) |
| | | 5 | Stack Pointer - 2 | 0 | Index Register (LSB) |
| | | 6 | Stack Pointer - 3 | 0 | Index Register (MSB) |
| | | 7 | Stack Pointer - 4 | 0 | Accumulator A |
| | | 8 | Stack Pointer - 5 | 0 | Accumulator B |
| | | 9 | Stack Pointer - 6 | 0 | Conditional Code Register |
| | | 10 | Vector Address FFFA | 1 | Address of SWI Routine (MSB) |
| | | 11 | Vector Address FFFB | 1 | Address of SWI Routine (LSB) |
| | | 12 | Address of SWI Routine | 1 | First Op Code of SWI Routine |
| SLP | 4 | 1 | Op Code Address+1 | 1 | Next Op Code |
| | | 2 | FFFF | 1 | Restart Address (LSB) |
| | | 3 | FFFF | | High Impedance-Non MPX Mode |
| | | 4 | Op Code Address+1 | | Address Bus -MPX Mode |
| | | 3 | FFFF | | Restart Address (LSB) |
| | | 4 | Op Code Address+1 | | Next Op Code |

- Continued -

Table 12 Cycle-by-Cycle Operation (Continued)

| Address Mode & Instructions | | Cycles | Cycle # | Address Bus | R W | Data Bus |
|-----------------------------|-----|--------|---------|--|-----|---|
| RELATIVE | | | | | | |
| BCC | BCS | 3 | 1 | Op Code Address + 1 | 1 | Branch Offset |
| BEQ | BGE | | 2 | FFFF | 1 | Restart Address (LSB) |
| BGT | BHI | | 3 | Branch Address.....Test="1" Op Code Address+1....Test="0" | 1 | First Op Code of Branch Routine Next Op Code |
| BLE | BLS | 5 | 1 | Op Code Address + 1 | 1 | Offset |
| BLT | BMT | | 2 | FFFF | 1 | Restart Address (LSB) |
| BNE | BPL | | 3 | Stack Pointer | 0 | Return Address (LSB) |
| BRA | BRN | | 4 | Stack Pointer - 1 | 0 | Return Address (MSB) |
| BVC | BVS | | 5 | Branch Address | 1 | First Op Code of Subroutine |
| BSR | | 5 | 1 | Op Code Address + 1 | 1 | Offset |
| | | | 2 | FFFF | 1 | Restart Address (LSB) |
| | | | 3 | Stack Pointer | 0 | Return Address (LSB) |
| | | | 4 | Stack Pointer - 1 | 0 | Return Address (MSB) |
| | | | 5 | Branch Address | 1 | First Op Code of Subroutine |

■ LOW POWER CONSUMPTION MODE

The HD6303R has two low power consumption modes; sleep and standby mode.

● Sleep Mode

On execution of SLP instruction, the MPU is brought to the sleep mode. In the sleep mode, the CPU sleeps (the CPU clock becomes inactive), but the contents of the registers in the CPU are retained. In this mode, the peripherals of CPU will remain active. So the operations such as transmit and receive of the SCI data and counter may keep in operation. In this mode, the power consumption is reduced to about 1/6 the value of a normal operation.

The escape from this mode can be done by interrupt, RES, STBY. The RES resets the MPU and the STBY brings it into the standby mode (This will be mentioned later). When interrupt is requested to the CPU and accepted, the sleep mode is released, then the CPU is brought in the operation mode and jumps to the interrupt routine. When the CPU has masked the interrupt, after recovering from the sleep mode, the next instruction of SLP starts to execute. However, in such a case that the timer interrupt is inhibited on the timer side, the sleep mode cannot be released due to the absence of the interrupt request to the CPU.

This sleep mode is available to reduce an average power consumption in the applications of the HD6303R which may not be always running.

● Standby Mode

Bringing STBY "Low", the CPU becomes reset and all clocks of the HD6303R become inactive. It goes into the standby mode. This mode remarkably reduces the power consumptions of the HD6303R.

In the standby mode, if the HD6303R is continuously supplied with power, the contents of RAM is retained. The standby mode should escape by the reset start. The following is the typical application of this mode.

First, NMI routine stacks the CPU's internal information and the contents of SP in RAM, disables RAME bit of RAM control register, sets the standby bit, and then goes into the standby mode. If the standby bit keeps set on reset start, it means that the power has been kept during stand-by mode and the contents of RAM is normally guaranteed. The system recovery may be possible by returning SP and bringing into the condition before the standby mode has started. The timing relation for each line in this application is shown in Figure 21.

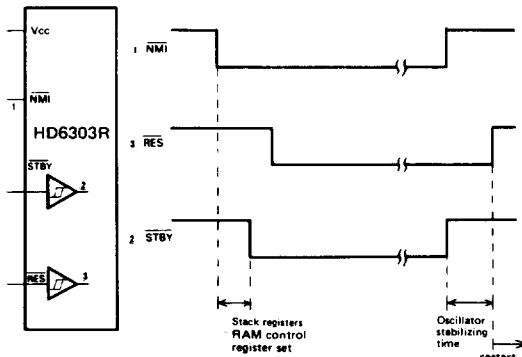


Figure 21 Standby Mode Timing

Table 12 Cycle-by-Cycle Operation (Continued)

| Address Mode & Instructions | Cycles | Cycle # | Address Bus | R/ \bar{W} | Data Bus |
|-----------------------------|--------|---------|------------------------|--------------|--|
| IMPLIED | | | | | |
| WAI | 9 | 1 | Op Code Address + 1 | 1 | Next Op Code |
| | | 2 | FFFF | 1 | Restart Address (LSB) |
| | | 3 | Stack Pointer | 0 | Return Address (LSB) |
| | | 4 | Stack Pointer - 1 | 0 | Return Address (MSB) |
| | | 5 | Stack Pointer - 2 | 0 | Index Register (LSB) |
| | | 6 | Stack Pointer - 3 | 0 | Index Register (MSB) |
| | | 7 | Stack Pointer - 4 | 0 | Accumulator A |
| | | 8 | Stack Pointer - 5 | 0 | Accumulator B |
| | | 9 | Stack Pointer - 6 | 0 | Conditional Code Register |
| RTI | 10 | 1 | Op Code Address + 1 | 1 | Next Op Code |
| | | 2 | FFFF | 1 | Restart Address (LSB) |
| | | 3 | Stack Pointer | 1 | Conditional Code Register |
| | | 4 | Stack Pointer + 1 | 1 | Accumulator B |
| | | 5 | Stack Pointer + 2 | 1 | Accumulator A |
| | | 6 | Stack Pointer + 3 | 1 | Index Register (MSB) |
| | | 7 | Stack Pointer + 4 | 1 | Index Register (LSB) |
| | | 8 | Stack Pointer + 5 | 1 | Return Address (MSB) |
| | | 9 | Stack Pointer + 6 | 1 | Return Address (LSB) |
| | | 10 | Return Address | 1 | First Op Code of Return Routine |
| SWI | 12 | 1 | Op Code Address + 1 | 1 | Next Op Code |
| | | 2 | FFFF | 1 | Restart Address (LSB) |
| | | 3 | Stack Pointer | 0 | Return Address (LSB) |
| | | 4 | Stack Pointer - 1 | 0 | Return Address (MSB) |
| | | 5 | Stack Pointer - 2 | 0 | Index Register (LSB) |
| | | 6 | Stack Pointer - 3 | 0 | Index Register (MSB) |
| | | 7 | Stack Pointer - 4 | 0 | Accumulator A |
| | | 8 | Stack Pointer - 5 | 0 | Accumulator B |
| | | 9 | Stack Pointer - 6 | 0 | Conditional Code Register |
| | | 10 | Vector Address FFFA | 1 | Address of SWI Routine (MSB) |
| | | 11 | Vector Address FFFB | 1 | Address of SWI Routine (LSB) |
| | | 12 | Address of SWI Routine | 1 | First Op Code of SWI Routine |
| SLP | 4 | 1 | Op Code Address + 1 | 1 | Next Op Code |
| | | 2 | FFFF | 1 | Restart Address (LSB) |
| | | Sleep | FFFF | | High Impedance-Non MPX Mode Address Bus -MPX Mode |
| | | 3 | FFFF | | Restart Address (LSB) |
| | | 4 | Op Code Address + 1 | | Next Op Code |

- Continued -

Table 12 Cycle-by-Cycle Operation (Continued)

| Address Mode & Instructions | | Cycles | Cycle # | Address Bus | R \overline{W} | Data Bus |
|-----------------------------|-----|--------|---------|---|------------------|---------------------------------|
| RELATIVE | | | | | | |
| BCC | BCS | 3 | 1 | Op Code Address+1 | 1 | Branch Offset |
| BEQ | BGE | | 2 | FFFF | 1 | Restart Address (LSB) |
| BGT | BHI | | 3 | Branch Address.....Test="1" Op Code Address+1...Test="0" | 1 | First Op Code of Branch Routine |
| BLE | BLS | | | | | Next Op Code |
| BLT | BMT | | | | | |
| BNE | BPL | | | | | |
| BRA | BRN | | | | | |
| BVC | BVS | | | | | |
| BSR | | 5 | 1 | Op Code Address+1 | 1 | Offset |
| | | | 2 | FFFF | 1 | Restart Address (LSB) |
| | | | 3 | Stack Pointer | 0 | Return Address (LSB) |
| | | | 4 | Stack Pointer-1 | 0 | Return Address (MSB) |
| | | | 5 | Branch Address | 1 | First Op Code of Subroutine |

■ **LOW POWER CONSUMPTION MODE**

The HD6303R has two low power consumption modes; sleep and standby mode.

● **Sleep Mode**

On execution of SLP instruction, the MPU is brought to the sleep mode. In the sleep mode, the CPU sleeps (the CPU clock becomes inactive), but the contents of the registers in the CPU are retained. In this mode, the peripherals of CPU will remain active. So the operations such as transmit and receive of the SCI data and counter may keep in operation. In this mode, the power consumption is reduced to about 1/6 the value of a normal operation.

The escape from this mode can be done by interrupt, \overline{RES} , \overline{STBY} . The \overline{RES} resets the MPU 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 sleep mode is released, then the CPU is brought in the operation mode and jumps to the interrupt routine. When the CPU has masked the interrupt, after recovering from the sleep mode, the next instruction of SLP starts to execute. However, in such a case that the timer interrupt is inhibited on the timer side, the sleep mode cannot be released due to the absence of the interrupt request to the CPU.

This sleep mode is available to reduce an average power consumption in the applications of the HD6303R which may not be always running.

● **Standby Mode**

Bringing \overline{STBY} "Low", the CPU becomes reset and all clocks of the HD6303R become inactive. It goes into the standby mode. This mode remarkably reduces the power consumptions of the HD6303R.

In the standby mode, if the HD6303R is continuously supplied with power, the contents of RAM is retained. The standby mode should escape by the reset start. The following is the typical application of this mode.

First, \overline{NMI} routine stacks the CPU's internal information and the contents of SP in RAM, disables RAME bit of RAM control register, sets the standby bit, and then goes into the standby mode. If the standby bit keeps set on reset start, it means that the power has been kept during stand-by mode and the contents of RAM is normally guaranteed. The system recovery may be possible by returning SP and bringing into the condition before the standby mode has started. The timing relation for each line in this application is shown in Figure 21.

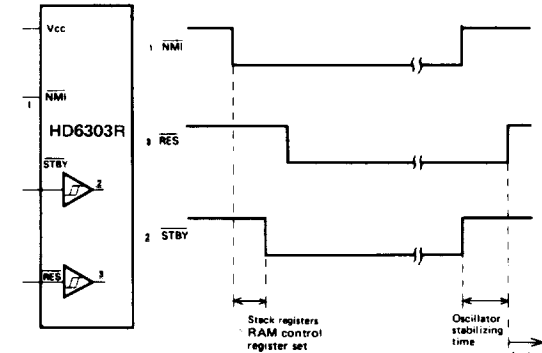


Figure 21 Standby Mode Timing

■ ERROR PROCESSING

When the HD6303R fetches an undefined instruction or fetches an instruction from unusable memory area, it generates the highest priority internal interrupt, that may protect from system upset due to noise or a program error.

● Op-Code Error

Fetching an undefined op-code, the HD6303R will stack the CPU register as in the case of a normal interrupt and vector to the TRAP (\$FFEE, \$FFEF), that has a second highest priority (RES is the highest).

● Address Error

When an instruction is fetched from other than a resident RAM, or an external memory area, the CPU starts the same interrupt as op-code error. In the case which the instruction is fetched from external memory area and that area is not usable, the address error can not be detected.

The address which cause address error are shown in Table 13.

This feature is applicable only to the instruction fetch, not to normal read/write of data accessing.

Transitions among the active mode, sleep mode, standby mode and reset are shown in Figure 22.

Figures 23, 24 show a system configuration.

The system flow chart of HD6303R is shown in Figure 25.

Table 13 Address Error

| Address Error |
|-----------------|
| \$0000 ~ \$001F |

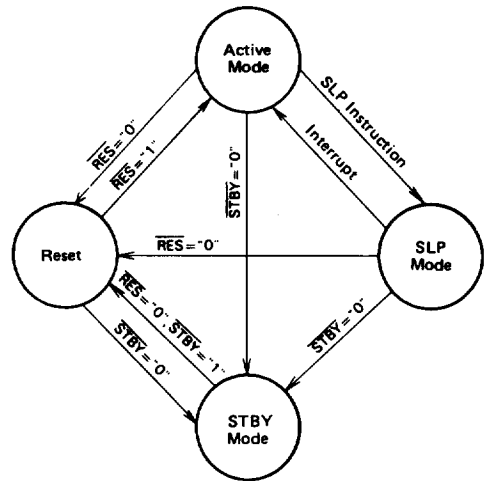


Figure 22 Transitions among Active Mode, Standby Mode, Sleep Mode, and Reset

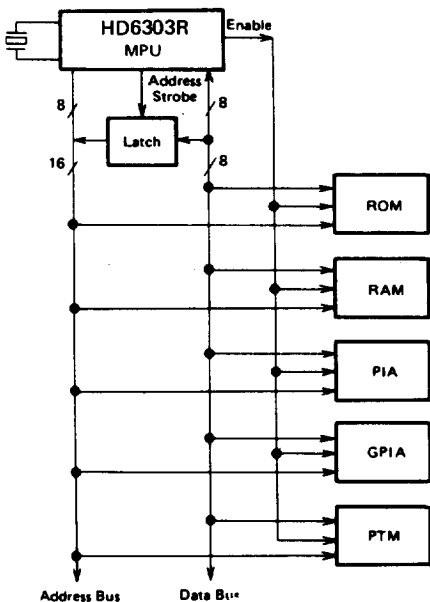


Figure 23 HD6303R MPU Multiplexed Mode

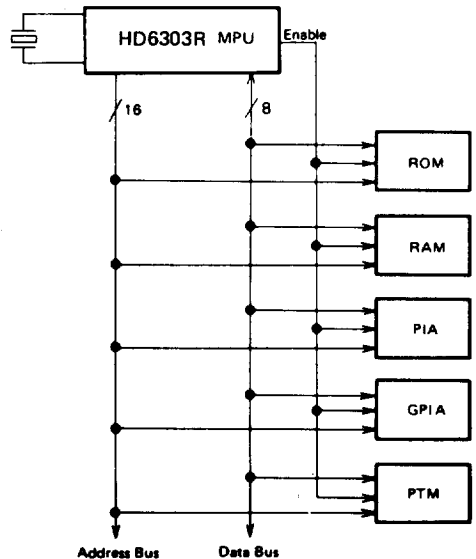


Figure 24 HD6303R MPU Non-Multiplexed Mode

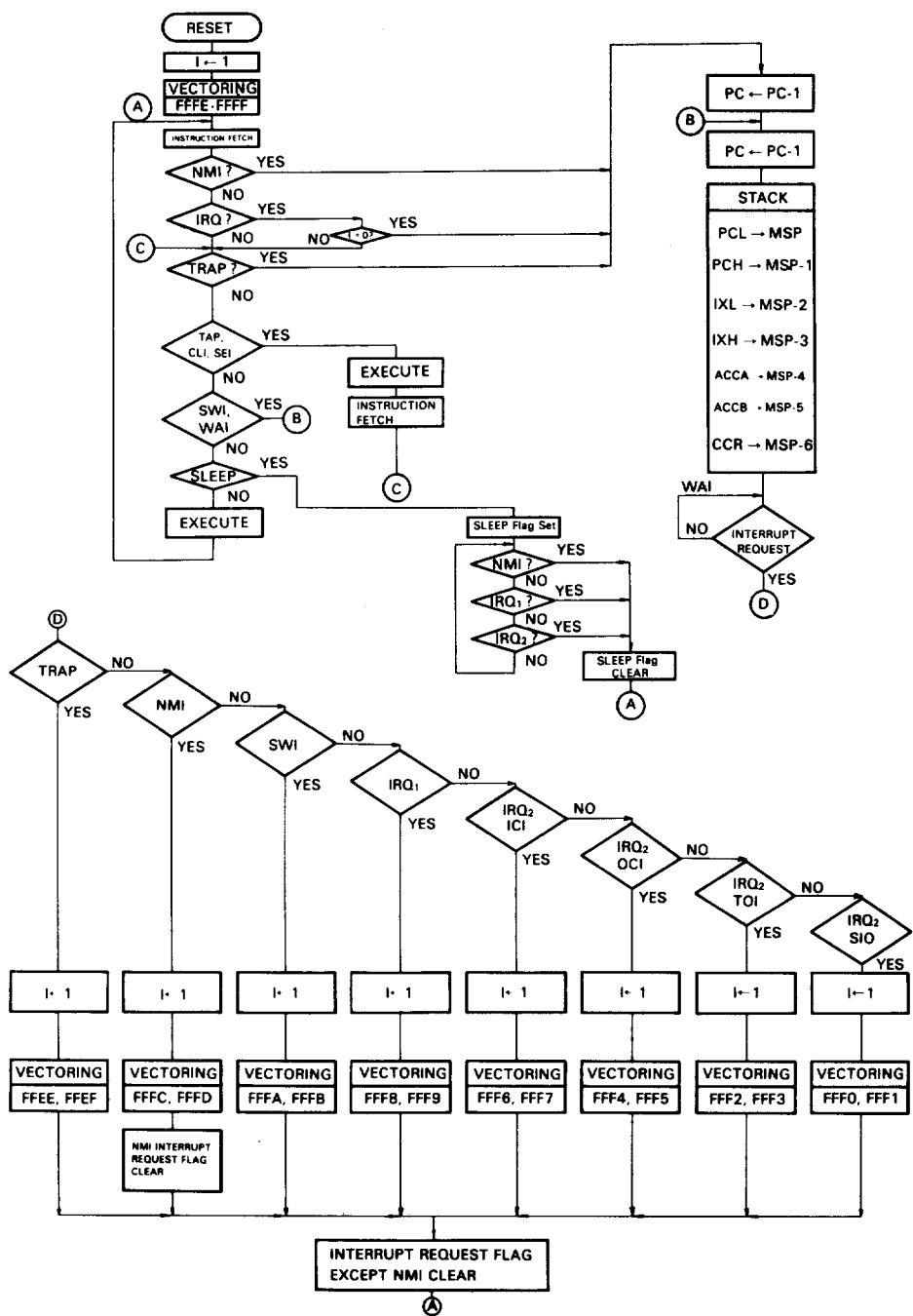
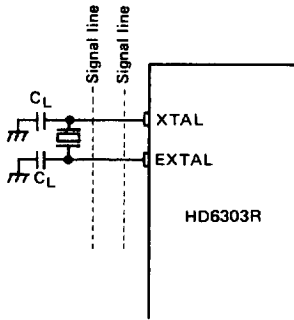


Figure 25 HD6303R System Flow Chart

■ PRECAUTION TO THE BOARD DESIGN OF OSCILLATION CIRCUIT

As shown in Fig. 26, there is a case that the cross talk disturbs the normal oscillation if signal lines are put near the oscillation circuit. When designing a board, pay attention to this. Crystal and C_L must be put as near the HD6303R as possible.



Do not use this kind of print board design.

Figure 26 Precaution to the board design of oscillation circuit

■ PIN CONDITIONS AT SLEEP AND STANDBY STATE

● Sleep State

The conditions of power supply pins, clock pins, input pins and E clock pin are the same as those of operation. Refer to Table 14 for the other pin conditions.

● Standby State

Only power supply pins and \overline{STBY} are active. As for the clock pin EXTAL, its input is fixed internally so the MPU is not influenced by the pin conditions. XTAL is in "1" output. All the other pins are in high impedance.

Table 14 Pin Condition in Sleep State

| Pin \ Mode | | Non Multiplexed Mode | Multiplexed Mode |
|---|-----------|---|--|
| P ₂₀ ~ P ₂₄ | Function | I/O Port | I/O Port |
| | Condition | Keep the condition just before sleep | ← |
| A ₀ /P ₁₀ ~ A ₇ /P ₁₇ | Function | Address Bus (A ₀ ~ A ₇) | I/O Port |
| | Condition | Output "1" | Keep the condition just before sleep |
| A ₈ ~ A ₁₅ | Function | Address Bus (A ₈ ~ A ₁₅) | Address Bus (A ₈ ~ A ₁₅) |
| | Condition | Output "1" | ← |
| D ₀ /A ₀ ~ D ₇ /A ₇ | Function | Data Bus (D ₀ ~ D ₇) | \overline{E} : Address Bus (A ₀ ~ A ₇), E: Data Bus |
| | Condition | High Impedance | \overline{E} : Output "1", E: High Impedance |
| R/W | Function | R/W Signal | R/W Signal |
| | Condition | Output "1" | ← |
| AS | | — | Output AS |

Table 15 Pin Condition during RESET

| Pin \ Mode | | Non-Multiplexed Mode | Multiplexed Mode |
|---|--|--------------------------------|------------------------------------|
| P ₂₀ ~ P ₂₄ | | High Impedance | ← |
| A ₀ /P ₁₀ ~ A ₇ /P ₁₇ | | High Impedance | ← |
| A ₈ ~ A ₁₅ | | High Impedance | ← |
| D ₀ /A ₀ ~ D ₇ /A ₇ | | High Impedance | E: "1" Output E: High Impedance |
| R/W | | "1" Output | ← |
| AS | | E: "1" Output E: "0" Output | ← |

■ DIFFERENCE BETWEEN HD6303 AND HD6303R

The HD6303R is an upgraded version of the HD6303. The difference between HD6303 and HD6303R is shown in Table 16.

Table 16 Difference between HD6303 and HD6303R

| Item | HD6303 | HD6303R |
|----------------------------|---|---|
| Operating Mode | Mode 2: Not defined | Mode 2: Multiplexed Mode (Equivalent to Mode 4) |
| Electrical Characteristics | The electrical characteristics of 2MHz version (B version) are not specified. | Some characteristics are improved. The 2MHz version is guaranteed. |
| Timer | Has problem in output compare function. (Can be avoided by software.) | The problem is solved. |