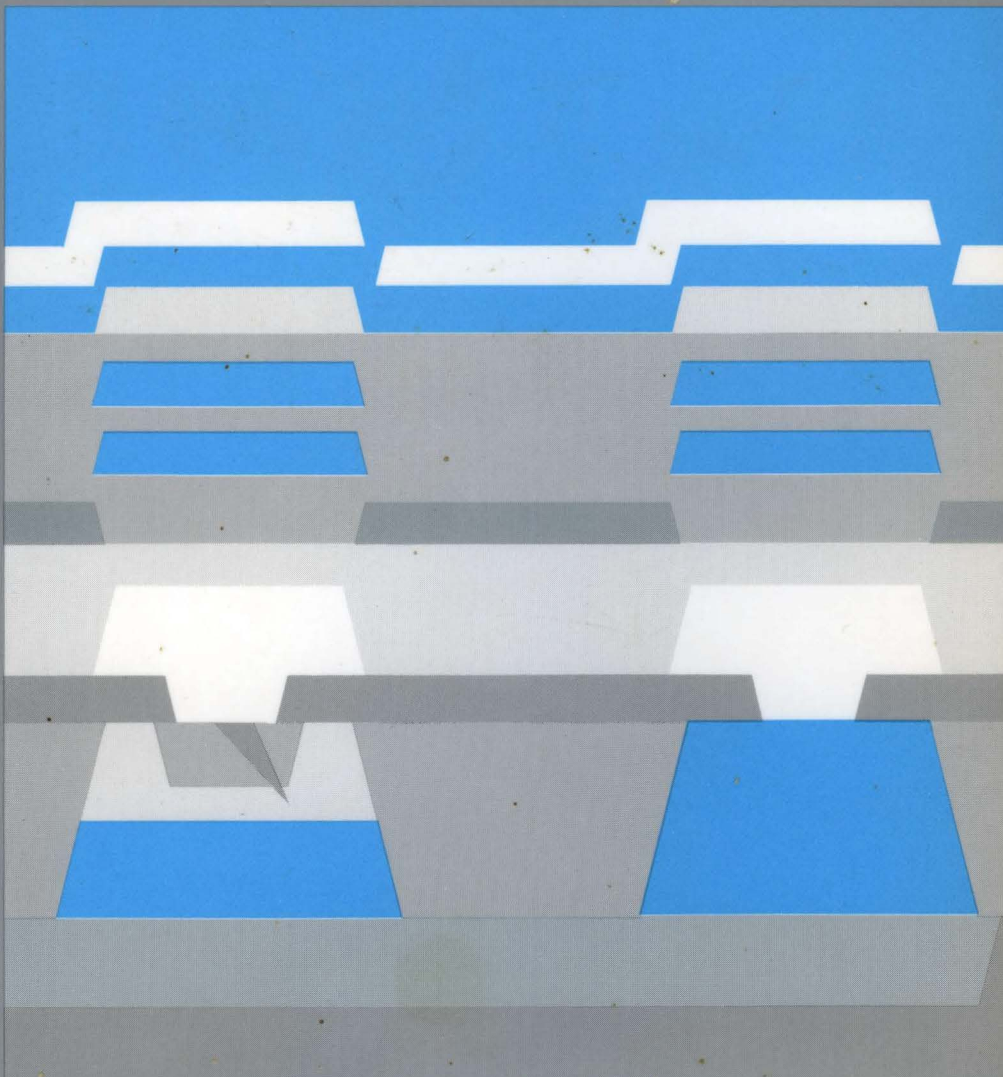
January, 1988

HITACHI®

HD647180X
8-BIT MICROCONTROLLER
HARDWARE MANUAL



#U94

High-Integration 180 Family

HD647180X
8-BIT MICROCONTROLLER
HARDWARE MANUAL

HITACHI®

#U94

ADE-602-005

MEDICAL APPLICATIONS

PREFACE

The HD647180X is a ZTAT microcontroller incorporating the following on a single chip: an instruction set compatible with the HD64180, 16-kbyte of programmable ROM, 512-byte of RAM, memory management unit (MMU), DMA controller, timer, asynchronous serial communication interface (ASCI), clock synchronous serial I/O port (CSI/O), analog comparator, and parallel I/O pins.

TABLE OF CONTENTS

Figures

Tables

SECTION 1. HD647180X OVERVIEW

The HD647180X provides instruction compatibility with the HD64180 and incorporates a 16-kbyte PROM, 512-byte RAM, memory management unit (MMU), DMA controller, timer, asynchronous serial communications interface (ASCI), clocked serial I/O ports (CSI/O), analog comparator and parallel I/O pins on a single chip.

The internal PROM can be programmed and verified under the same specifications as the 27256 series using a general-purpose PROM writer.

1.1 Features

1.1.1 Software

- Instruction set compatible with the HD64180

1.1.2 Hardware

- 16-kbyte PROM and 512-byte RAM
- Timer
  - One-channel 16-bit timer with input capture, output compare, and timer overflow functions
  - Two-channel 16-bit reload timer
- Six-channel analog comparator
- 54 parallel I/O pins
  - Includes eight high current pins I_{OL} = 10 mA)
- MMU with 1-Mbyte memory physical address space
- Two-channel DMA controller
- Two-channel ASCI
- One-channel CSI/O
- Four external and eight internal interrupts
- DRAM refresh controller and low speed memory, I/O interface
- Operating frequency up to 8 MHz (ϕ clock)
- Low power operation
- Four operation modes
  - Mode 0: single-chip mode
  - Mode 1: expanded mode (internal ROM disabled)
  - Mode 2: expanded mode (internal ROM enabled)
  - Mode 3: PROM programming mode
- Internal ROM data protect function
- Packages
  - 80-pin quad flat package
  - 84-pin plastic leaded chip carrier

1.2 Block Diagram

The HD647180X combines a high-performance CPU core with many of the systems and I/O resources required by a broad range of applications (figure 1-1).

The CPU core consists of five functional blocks:

- Clock generator
- Bus state controller
- Interrupt controller
- Memory management unit (MMU)
- Central processing unit (CPU)

The integrated I/O resources comprise the remaining four functional blocks:

- DMA controller (DMAC: two channels)
- Asynchronous serial communication interface (ASCI: two channels)
- Clocked serial I/O port (CSI/O: one channel)
- Programmable reload timer (PRT: two channels)
- Programmable timer 2 (PT2: one channel)
- Analog comparator (six channels)
- I/O ports

The memory consists of:

- RAM (512 bytes)
- PROM (16 kbyte)

HD647180X Speed and Packages

Type No.	Clock Frequency	Package
HD647180XF-4	4 MHz	FP-80
HD647180XF-6	6 MHz	
HD647180XF-8	8 MHz	
HD647180XCP-4	4 MHz	CP-84
HD647180XCP-6	6 MHz	
HD647180XCP-8	8 MHz	

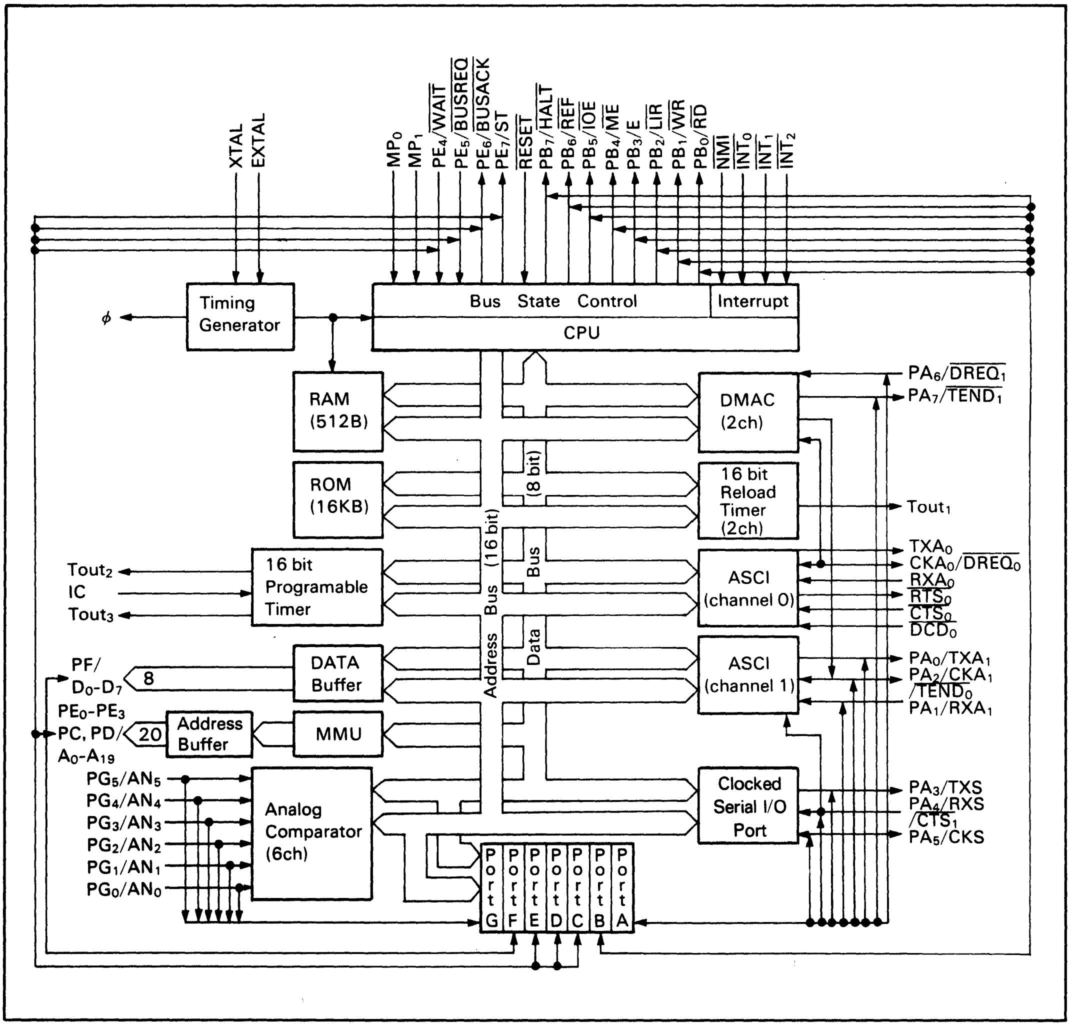


Figure 1-1. Block Diagram

1.3 Pin Assignment

Figure 1-2. shows a top view of the HD647180X packages. Table 1-1. shows the pin functions in the four modes.

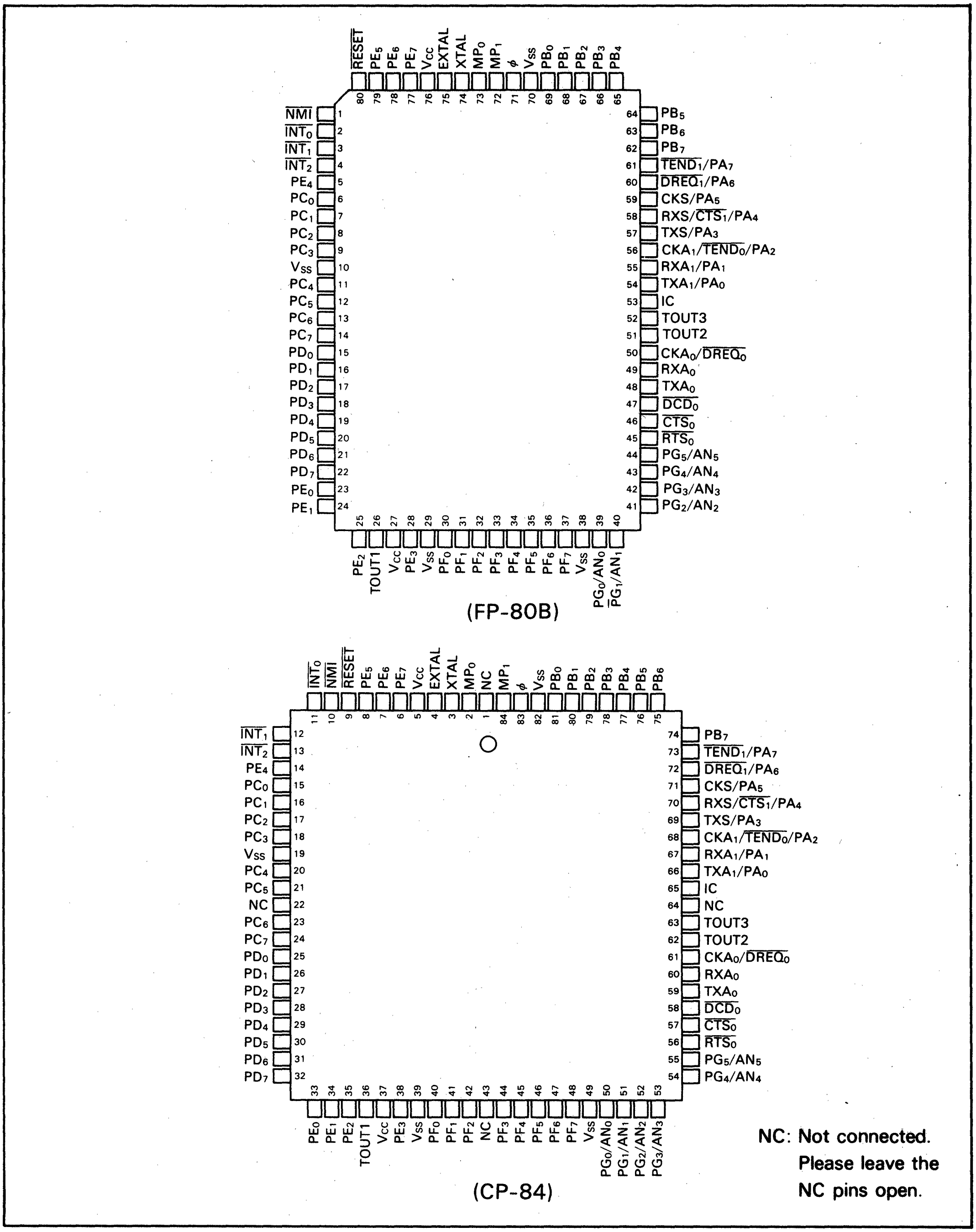


Figure 1-2. Pin Assignment

Table 1-1. Pin Function

Pin No.		Mode 0	Mode 1	Mode 2	Mode 3
FP-80B	CP-84				
1	10	$\overline{NMI}$	←	←	A_9
2	11	$\overline{INT_0}$	←	←	—
3	12	$\overline{INT_1}$	←	←	—
4	13	$\overline{INT_2}$	←	←	—
5	14	PE_4	ST	←	—
6	15	PC_0	A_0	←	←
7	16	PC_1	A_1	←	←
8	17	PC_2	A_2	←	←
9	18	PC_3	A_3	←	←
10	19	V_{SS}	←	←	←
11	20	PC_4	A_4	←	←
12	21	PC_5	A_5	←	←
13	23	PC_6	A_6	←	←
14	24	PC_7	A_7	←	←
15	25	PD_0	A_8	A_8/PD_0	A_8
16	26	PD_1	A_9	A_9/PD_1	—
17	27	PD_2	A_{10}	A_{10}/PD_2	A_{10}
18	28	PD_3	A_{11}	A_{11}/PD_3	A_{11}
19	29	PD_4	A_{12}	A_{12}/PD_4	A_{12}
20	30	PD_5	A_{13}	A_{13}/PD_5	A_{13}
21	31	PD_6	A_{14}	A_{14}/PD_6	A_{14}
22	32	PD_7	A_{15}	A_{15}/PD_7	$\overline{OE}$
23	33	PE_0	A_{16}	A_{16}/PE_0	$\overline{CE}$
24	34	PE_1	A_{17}	A_{17}/PE_1	—
25	35	PE_2	A_{18}	A_{18}/PE_2	—
26	36	TOUT1	←	←	—
27	37	V_{CC}	←	←	←
28	38	PE_3	A_{19}	A_{19}/PE_3	—
29	39	V_{SS}	←	←	←
30	40	PF_0	D_0	←	O_0
31	41	PF_1	D_1	←	O_1
32	42	PF_2	D_2	←	O_2
33	44	PF_3	D_3	←	O_3
34	45	PF_4	D_4	←	O_4
35	46	PF_5	D_5	←	O_5
36	47	PF_6	D_6	←	O_6
37	48	PF_7	D_7	←	O_7
38	49	V_{SS}	←	←	←
39	50	PG_0/AN_0	←	←	—
40	51	PG_1/AN_1	←	←	—

Notes: ← Same as previous column
— No function

Table 1-1. Pin Function (cont.)

Pin No.		Mode 0	Mode 1	Mode 2	Mode 3
FP-80B	CP-84				
41	52	PG_2/AN_2	←	←	–
42	53	PG_3/AN_3	←	←	–
43	54	PG_4/AN_4	←	←	–
44	55	PG_5/AN_5	←	←	–
45	56	$\overline{RTS_0}$	←	←	–
46	57	$\overline{CTS_0}$	←	←	–
47	58	$\overline{DCD_0}$	←	←	–
48	59	TXA_0	←	←	–
49	60	RXA_0	←	←	–
50	61	$CKA_0/\overline{DREQ_0}$	←	←	–
51	62	TOUT2	←	←	–
52	63	TOUT3	←	←	–
53	65	IC	←	←	–
54	66	TXA_1/PA_0	←	←	–
55	67	RXA_1/PA_1	←	←	–
56	68	$CKA_1/\overline{TEND_0}/PA_2$	←	←	–
57	69	TXS/PA_3	←	←	–
58	70	$RXS/\overline{CTS_1}/PA_4$	←	←	–
59	71	CKS/PA_5	←	←	–
60	72	$\overline{DREQ_1}/PA_6$	←	←	–
61	73	$\overline{TEND_1}/PA_7$	←	←	–
62	74	PB_7	$\overline{HALT}$	←	–
63	75	PB_6	$\overline{REF}$	←	–
64	76	PB_5	$\overline{IOE}$	←	–
65	77	PB_4	$\overline{ME}$	←	–
66	78	PB_3	E	←	–
67	79	PB_2	$\overline{LIR}$	←	–
68	80	PB_1	$\overline{WR}$	←	–
69	81	PB_0	$\overline{RD}$	←	–
70	82	V_{SS}	←	←	←
71	83	ϕ	←	←	–
72	84	MP_1	←	←	←
73	2	MP_0	←	←	←
74	3	XTAL	←	←	←
75	4	EXTAL	←	←	←
76	5	V_{CC}	←	←	←
77	6	PE_7	$\overline{WAIT}$	←	–
78	7	PE_6	$\overline{BUSACK}$	←	–
79	8	PE_5	$\overline{BUSREQ}$	←	–
80	9	$\overline{RESET}$	←	←	V_{PP}

1.4 CPU Architecture

The five CPU core functional blocks are described in this section.

1.4.1 Clock Generator

The clock generator generates the system clock (ϕ) from an external crystal or external clock input. Also, the system clock is programmably prescaled to generate timing for the on-chip I/O and system support devices.

1.4.2 Bus State Controller

The bus state controller performs all status/control bus activity. This includes external bus cycle wait state timing, $\overline{\text{RESET}}$, DRAM refresh, and master DMA bus exchange. Generates 'dual-bus' control signals for compatibility with peripheral devices.

1.4.3 Interrupt Controller

The interrupts controller monitors and prioritizes the four external and eight internal interrupt sources. A variety of interrupt response modes are programmable.

1.4.4 Memory Management Unit (MMU)

Maps the CPU 64-kbyte logical memory address space into a 1-Mbyte physical memory address space. The MMU organization preserves software object code compatibility while providing extended memory access and uses an efficient 'common area—bank area' scheme. I/O accesses (64-kbyte I/O address space) bypass the MMU.

1.4.5 Central Processing Unit (CPU)

The CPU is microcoded to implement an upward-compatible superset of the 8-bit standard software instruction set. Many instructions require fewer clock cycles for execution and seven new instructions are added.

1.4.6 Mode Selection

Mode program pins, MP_0 and MP_1 determine the operation mode of the HD647180X (table 2-3).

1.5 I/O Resources

1.5.1 DMA Controller (DMAC)

The two channel DMAC provides high speed memory to/from memory, memory to/from I/O, and memory to/from memory-mapped I/O transfers. The DMAC features edge or level sense request input, address increment/decrement/no-change and (for memory to/from memory transfers) programmable burst or cycle steal transfer. In addition, the DMAC can directly access the full 1-Mbyte of physical memory address space (the MMU is bypassed during DMA) and transfers (up to 64-kbyte in length) can cross 64-kbyte boundaries. See figure 10-1. for further details.

1.5.2 Asynchronous Serial Communication Interface (ASCI)

The ASCI provides two separate full-duplex UARTs and includes a programmable baud rate generator, modem control signals, and a multiprocessor communication format. The ASCI can use the DMAC for high-speed serial data transfer, reducing CPU overhead. See figure 11-1. for further details.

1.5.3 Clocked Serial I/O Port (CSI/O)

The CSI/O half-duplex clocked serial transmitter and receiver can be used for simple, high-speed connection to another microprocessor or microcomputer. See figure 12-1. for further details.

1.5.4 Programmable Reload Timer (PRT)

The PRT contains two separate channels, each consisting of 16-bit timer data and 16-bit timer reload registers. The time base is the system clock divided by 20 (fixed) and PRT channel 1 has an optional output allowing waveform generation. See figure 13-1. for further details.

1.5.5 Programmable Timer 2 (PT2)

The PT2 16-bit programmable timer can measure an input waveform and generate two independent output waveforms. The pulse widths of both input/output waveforms vary from microseconds to seconds (figure 15-1.).

1.5.6 Analog Comparator

The HD647180X provides an analog comparator with 6 channels. Each channel can be programmed as a reference voltage (V_{ref}) input pin or a compared voltage (V_{in}) input pin. See figure 16-1. for further details.

1.5.7 Input Output Port (I/O Port)

The HD647180X provides seven I/O ports. (port A–G). Each port consists of a data direction register (DDR) to determine the directions of the individual pins, an output data register (ODR) to hold output data and an input data register (IDR) to latch input data. However, Port G does not have a DDR or ODR since it is an input-only port.

SECTION 2. HD647180X PINS

2.1 Signal Description

2.1.1 XTAL, EXTAL: Crystal (Input)

XTAL and EXTAL are the crystal oscillator connections. An external TTL clock can be input on EXTAL. XTAL should be left open if an external TTL clock is used. Note that XTAL. XTAL is schmitt triggered. See Section 23 DC characteristics.

2.1.2 ϕ (OUT)

ϕ is the system clock output. Its frequency is equal to one-half of the crystal oscillator's.

2.1.3 $\overline{\text{RESET}}$: CPU Reset (Input)

When $\overline{\text{RESET}}$ is low, it initializes the HD647180X CPU. All output signals are held inactive during reset.

2.1.4 A_0-A_{19}: Address Bus (Output, Three-State)

The address bus enters the high-impedance state during reset and when another device acquires the bus as indicated by $\overline{\text{BUSREQ}}$ and $\overline{\text{BUSACK}}$ low. During reset, the address function is selected.

2.1.5 D_0-D_7: Data Bus (Input/Output, Three-State)

The bidirectional 8-bit data bus enters the high-impedance state during reset and when another device acquires the bus as indicated by $\overline{\text{BUSREQ}}$ and $\overline{\text{BUSACK}}$ low.

2.1.6 $\overline{\text{RD}}$: Read (Output, Three-State)

During a CPU read cycle, $\overline{\text{RD}}$ enables transfer from the external memory or I/O device to the CPU data bus.

2.1.7 $\overline{\text{WR}}$: Write (Output, Three-State)

During a CPU write cycle, $\overline{\text{WR}}$ enables transfer from the CPU data bus to the external memory or I/O device.

2.1.8 $\overline{\text{ME}}$: Memory Enable (Output, Three-State)

$\overline{\text{ME}}$ indicates memory read or write operations. The HD647180X asserts $\overline{\text{ME}}$ low in the following cases.

- When fetching instructions and operands
- When reading or writing memory data
- During DMA memory access cycles
- During dynamic RAM refresh cycles

2.1.9 $\overline{\text{IOE}}$: I/O Enable (Output, Three-State)

$\overline{\text{IOE}}$ indicates I/O read or write operations. The HD647180X asserts $\overline{\text{IOE}}$ low in the following cases:

- When reading or writing I/O data
- During DMA I/O access cycles
- During INT$_0$ acknowledge cycle

2.1.10 $\overline{\text{WAIT}}$: Bus Cycle Wait (Input)

$\overline{\text{WAIT}}$ introduces wait states to extend memory and I/O cycles. If low at the falling edge of T$_2$, a wait state (Tw) is inserted. Wait states will continue to be inserted until the $\overline{\text{WAIT}}$ input is sampled high at the falling edge of Tw, at which time the bus cycle will proceed to completion.

2.1.11 E: Enable (Output)

E is a synchronous clock for connection to HD63×× series and other 6800/6500 series compatible peripheral LSIs.

2.1.12 $\overline{\text{BUSREQ}}$: Bus Request (Input)

Another device may request use of the bus by asserting $\overline{\text{BUSREQ}}$ low. The CPU will stop executing instructions and place the address bus, data bus, $\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{ME}}$, and $\overline{\text{IOE}}$ in the high-impedance state.

2.1.13 $\overline{\text{BUSACK}}$: Bus Acknowledge (Output)

When the CPU completes bus release (in response to $\overline{\text{BUSREQ}}$ low), it will assert $\overline{\text{BUSACK}}$ low. This acknowledges that the bus is free for use by the requesting device.

2.1.14 $\overline{\text{HALT}}$: Halt/Sleep Status (Output)

$\overline{\text{HALT}}$ is asserted low after execution of the HALT or SLP instructions. Used with $\overline{\text{LIR}}$ and ST output pins to encode CPU status (table 2-1.).

2.1.15 $\overline{\text{LIR}}$: Load Instruction Register (Output)

$\overline{\text{LIR}}$ is asserted low when the current cycle is an opcode fetch cycle. Used with $\overline{\text{HALT}}$ and ST output pins to encode CPU status (table 2-1.).

2.1.16 ST: Status (Output)

ST is used with the $\overline{\text{HALT}}$ and $\overline{\text{LIR}}$ output pins to encode CPU status (table 2-1.).

Table 2-1. Status Summary

ST	$\overline{\text{HALT}}$	$\overline{\text{LIR}}$	Operation
0	1	0	CPU operation (1st opcode fetch)
1	1	0	CPU operation (2nd opcode and 3rd opcode fetch)
1	1	1	CPU operation (MC except for opcode fetch)
0	X	1	DMA operation
0	0	0	Halt mode
1	0	1	Sleep mode (including System stop mode)

Note X: Don't care
MC: Machine cycle

2.1.17 $\overline{\text{REF}}$: Refresh (Output)

When low, $\overline{\text{REF}}$ indicates that the CPU is in a dynamic RAM refresh cycle and the low-order 8 bits (A_0-A_7) of the address bus contain the refresh address.

2.1.18 $\overline{\text{NMI}}$: Non-Maskable Interrupt (Input)

When high to low is detected, it forces the CPU to save certain state information and vector to an interrupt service routine at address 0066H. The saved state information is restored by executing the RETN (return from non-maskable interrupt) instruction.

2.1.19 $\overline{\text{INT}_0}$: Maskable Interrupt Level 0 (Input)

When low, $\overline{\text{INT}_0}$ requests a CPU interrupt (unless masked) and saves certain state information unless masked by software. $\overline{\text{INT}_0}$ requests service using one of three software programmable interrupt modes (table 2-2.).

Table 2-2. Interrupt Modes

Mode	Operation
0	Instruction fetched and executed from data bus
1	Instruction fetched and executed from address 0038H
2	Vector system: Low-order 8 bits of vector table address fetched from data bus

In all modes, the saved state information is restored by executing the RETI (return from interrupt) instruction.

2.1.20 $\overline{\text{INT}_1}$, $\overline{\text{INT}_2}$: Maskable Interrupt Levels 1, 2 (Input)

When low, $\overline{\text{INT}_1}$ and $\overline{\text{INT}_2}$ request a CPU interrupt (unless masked) and save certain state information unless masked by software. $\overline{\text{INT}_1}$ and $\overline{\text{INT}_2}$ (and internally generated interrupts) request interrupt service using a vector system similar to mode 2 of $\overline{\text{INT}_0}$.

2.1.21 $\overline{\text{DREQ}_0}$: DMA Request—Channel 0 (Input)

$\overline{\text{DREQ}_0}$ low (programmable edge or level sense) requests DMA transfer service from channel 0 of the HD647180X DMAC. $\overline{\text{DREQ}_0}$ is used for channel 0 memory to/from I/O and memory to/from memory-mapped I/O transfers. $\overline{\text{DREQ}_0}$ is not used for memory to/from memory transfers. This pin is multiplexed with CKA_0.

2.1.22 $\overline{\text{TEND}_0}$: Transfer End—Channel 0 (Output)

$\overline{\text{TEND}_0}$ is asserted low synchronous with the last write cycle of channel 0 DMA transfer to indicate DMA completion to an external device. This pin is multiplexed with CKA_1.

2.1.23 $\overline{\text{DREQ}_1}$: DMA Request—Channel 1 (Input)

$\overline{\text{DREQ}_1}$ low (programmable edge or level sense) requests DMA transfer service from channel 1 of the HD647180X DMAC. Channel 1 supports memory to/from I/O transfers.

2.1.24 $\overline{\text{TEND}_1}$: Transfer End—Channel 1 (Output)

$\overline{\text{TEND}_1}$ is asserted low synchronous with the last write cycle of channel 1 DMA transfer to indicate DMA completion to an external device.

2.1.25 TXA_0: Asynchronous Transmit Data—Channel 0 (Output)

TXA_0 is the asynchronous transmit data from channel 0 of the asynchronous serial communication interface (ASCI).

2.1.26 RXA_0: Asynchronous Receive Data—Channel 0 (Input)

RXA_0 is the asynchronous receive data to channel 0 of the ASCI.

2.1.27 CKA_0: Asynchronous Clock—Channel 0 (Input/Output)

CKA_0 is the clock input/output for channel 0 of the ASCI. This pin is multiplexed (software selectable) with $\overline{DREQ_0}$.

2.1.28 $\overline{RTS_0}$: Request to Send—Channel 0 (Output)

$\overline{RTS_0}$ is the programmable modem control output signal for channel 0 of the ASCI.

2.1.29 $\overline{CTS_0}$: Clear to Send—Channel 0 (Input)

$\overline{CTS_0}$ is the modem control input signal for channel 0 of the ASCI.

2.1.30 $\overline{DCD_0}$: Data Carrier Detect—Channel 0 (Input)

$\overline{DCD_0}$ is the modem control input signal for channel 0 of the ASCI.

2.1.31 TXA_1: Asynchronous Transmit Data—Channel 1 (Output)

TXA_1 is the asynchronous transmit data from channel 1 of the ASCI.

2.1.32 RXA_1: Asynchronous Receive Data—Channel 1 (Input)

RXA_1 is the asynchronous receive data to channel 1 of the ASCI.

2.1.33 CKA_1: Asynchronous Clock—Channel 1 (Input/Output)

CKA_1 is the clock input/output for channel 1 of the ASCI. This pin is multiplexed (software selectable) with $\overline{TEND_0}$.

2.1.34 $\overline{CTS_1}$: Clear to Send—Channel 1 (Input)

$\overline{CTS_1}$ is the modem control input signal for channel 1 of the ASCI. This pin is multiplexed (software selectable) with RXS.

2.1.35 TXS: Clocked Serial Transmit Data (Output)

Clocked serial transmit data from the Clocked Serial I/O Port (CSI/O).

2.1.36 RXS: Clocked Serial Receive Data (Input)

Clocked serial receive data to the CSI/O. This pin is multiplexed (software selectable) with ASCI channel 1 $\overline{CTS_1}$ modem control input.

2.1.37 CKS: Serial Clock (Input/Output)

Input or output clock for the CSI/O.

2.1.38 TOUT1: Timer Output (Output)

Pulse output from Programmable Reload Timer channel 1.

2.1.39 AN_0-AN_5: Comparator (Input)

AN_0-AN_5 input data to the analog comparator. Select two of these pins and apply the reference voltage (V_{ref}) and the voltage to be compared (V_{in}) to them.

2.1.40 PA_0-PA_7, PB_0-PB_7, PC_0-PC_7, PD_0-PD_7, PE_0-PE_7, PF_0-PF_7: Parallel Ports A-F (Input/Output)

Ports A-F are 8-bit I/O ports. Each pin of each port can be individually configured as an input or output depending on the port data direction register. At reset, each port is initialized as an input port.

2.1.41 PG_0-PG_5: Parallel Port G (Input)

Port G is a 6-bit input port.

2.1.42 IC: Input Capture (Input)

IC inputs the input capture signal for timer 2.

2.1.43 TOUT2, TOUT3: Timer Output 2, 3 (Output)

TOUT2 and TOUT3 are timer 2's outputs.

2.1.44 MP_0, MP_1: Mode Program 0,1 (Input)

The mode program pins, MP_0 and MP_1, determine the operation mode of the MPU as shown in table 2-3.

Table 2-3. Operating Mode Selection

Mode	MP_1	MP_0	ROM	RAM	Operating Mode
0	0	0	I	I	Single chip mode
1	0	1	E	I	Expanded mode 1
2	1	0	I	I	Expanded mode 2
3	1	1	I	–	PROM programming mode

I: Internal E: External

2.1.45 VCC, VSS: Power

VCC is the HD647180X power supply. VSS is the ground.

2.2 Multiplexed Pins

2.2.1 PA_0/TXA_1, PA_1/RXA_1, PA_3/TXS, PA_5/CKS, $PA_6/\overline{DREQ_1}$, $PA_7/\overline{TEND_1}$

At reset, PA_0/TXA_1, PA_1/RXA_1, PA_3/TXS, PA_5/CKS, $PA_6/\overline{DREQ_1}$, and $PA_7/\overline{TEND_1}$ are configured as port A input. They can be used as TXA_1, RXA_1, TXS, CKS, $\overline{DREQ_1}$, and $\overline{TEND_1}$ by setting the corresponding bit in the port A disable register to 1.

2.2.2 $PA_2/CKA_1/\overline{TEND_0}$

At reset, $PA_2/CKA_1/\overline{TEND_0}$ is configured as a port A input. The function of this pin depends on the combination of bit 2 in the port A disable register (DERA2) and the CKA1D bit in the ASCI control register channel 1 (table 2-4.).

Table 2-4. $PA_2/CKA_1/\overline{TEND_1}$ State

DERA2	CKA1D	Pin Function
0	0, 1	PA_2
1	0	CKA_1
	1	$\overline{TEND_0}$

2.2.3 $PA_4/RXS/\overline{CTS_1}$

At reset, $PA_4/RXS/\overline{CTS_1}$ is configured as a port A input. The function of this pin depends on the combination of bit 4 in the port A disable register (DERA4) and the CKA1D bit in the ASCI control register channel 1 (table 2-5.).

Table 2-5. $PA_4/RXS/\overline{CTS_1}$ State

DERA4	CKA1D	Pin Function
0	0, 1	PA_4
1	0	RXS
	1	$\overline{CTS_1}$

2.2.4 $CKA_0/\overline{DREQ_0}$

$CKA_0/\overline{DREQ_0}$ is configured as the CKA_0 at reset. When either the DM1 or SM1 bit of the DMA mode registers 1, this bit is forcibly configured as the $\overline{DREQ_0}$ input, even if it has been configured as an output pin.

SECTION 3. CPU BUS TIMING

This section explains the HD647180X CPU timing for the following operations:

1. Instruction (opcode) fetch timing
2. Operand and data read/write timing
3. I/O read/write timing
4. Basic instruction (fetch and execute) timing
5. Reset timing
6. $\overline{BUSREQ}/\overline{BUSACK}$ bus exchange timing

The basic CPU operation consists of one or more machine cycles (MC). A machine cycle consists of three system clocks, T_1, T_2, and T_3 during memory or I/O access, or it consists of one system clock, Ti during CPU internal operation. The system clock (ϕ) is half the frequency of crystal oscillation (for example 8 MHz crystal $\longrightarrow$ ϕ of 4 MHz, 250 nsec). To interface to slow memory or peripherals, optional wait states (Tw) may be inserted between T_2 and T_3.

3.1 Instruction (Opcode) Fetch Timing

Figure 3-1. shows the instruction (opcode) fetch timing with no wait states.
An opcode fetch cycle is externally indicated when the $\overline{LIR}$ (load instruction register) output pin is low.

In the first half of T_1, the address bus is driven with the contents of the program counter (PC). Note that this is the translated address output of the HD647180X on-chip MMU.

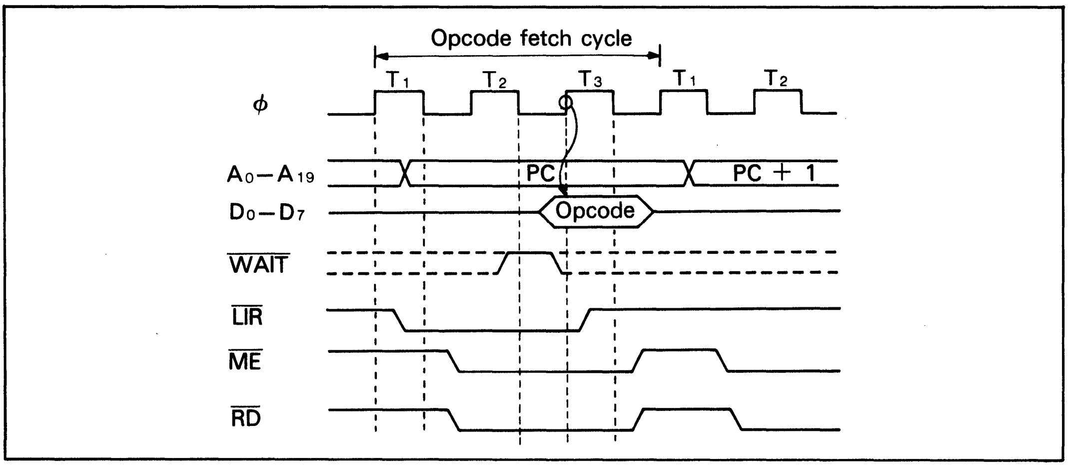


Figure 3-1. Opcode Fetch Timing

In the second half of T_1, the $\overline{\text{ME}}$ (memory enable) and $\overline{\text{RD}}$ (read) signals are asserted low, enabling the memory.

The opcode on the data bus is latched at the rising edge of T_3 and the bus cycle terminates at the end of T_3.

Figure 3-2. illustrates the insertion of wait states (Tw) into the opcode fetch cycle. Wait states (Tw) are controlled by the external $\overline{\text{WAIT}}$ input combined with an on-chip programmable wait state generator.

At the falling edge of T_2 the combined $\overline{\text{WAIT}}$ input is sampled. If $\overline{\text{WAIT}}$ input is asserted low, a wait state (Tw) is inserted. The address bus, $\overline{\text{ME}}$, $\overline{\text{RD}}$, and $\overline{\text{LIR}}$ are held stable during wait states. When the $\overline{\text{WAIT}}$ is sampled inactive high at the falling edge of Tw, the bus cycle enters T_3 and completes at the end of T_3.

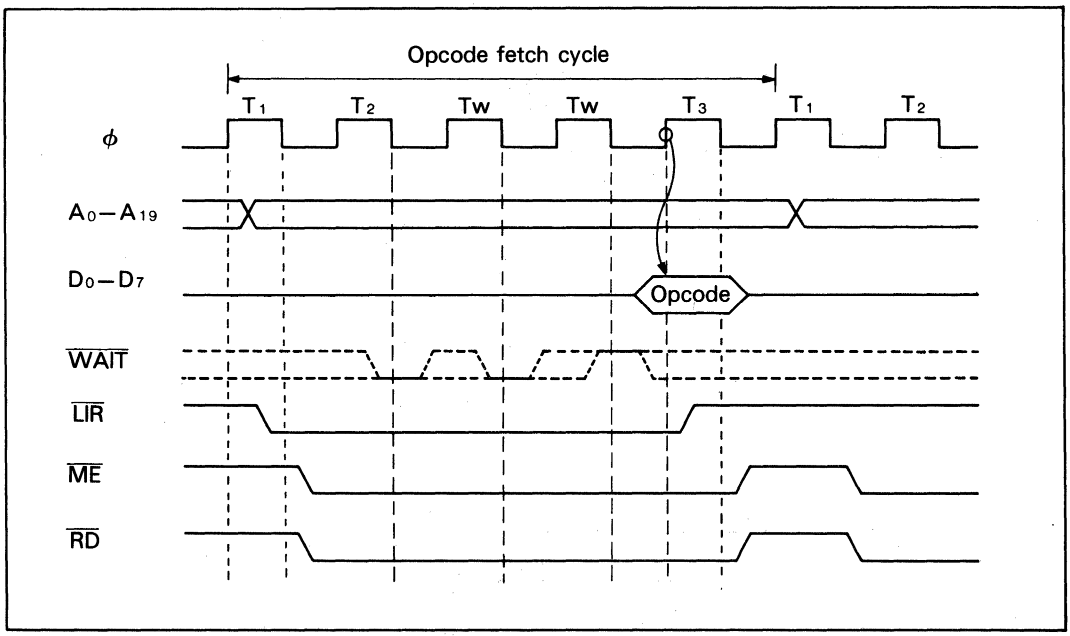


Figure 3-2. Opcode Fetch Timing (with Wait State)

3.2 Operand and Data Read/Write Timing

The instruction operand and data read/write timing differs from opcode fetch timing in two ways. First, the $\overline{LIR}$ output is held inactive. Second, the read cycle timing is relaxed by one-half clock cycle since data is latched at the falling edge of T_3.

Instruction operands include immediate data, displacement, and extended addresses and have the same timing as memory data reads.

During memory write cycles the $\overline{ME}$ signal goes active in the second half of T_1. At the end of T_1, the data bus is driven with the write data.

At the start of T_2, the $\overline{WR}$ signal is asserted low, enabling the memory. $\overline{ME}$ and $\overline{WR}$ go inactive in the second half of T_3, followed by deactivation of the write data on the data bus.

Wait states (Tw) are inserted as previously described for opcode fetch cycles. Figure 3-3. illustrates the read/write timing without wait states (Tw), while figure 3-4. illustrates read/write timing with wait states (Tw).

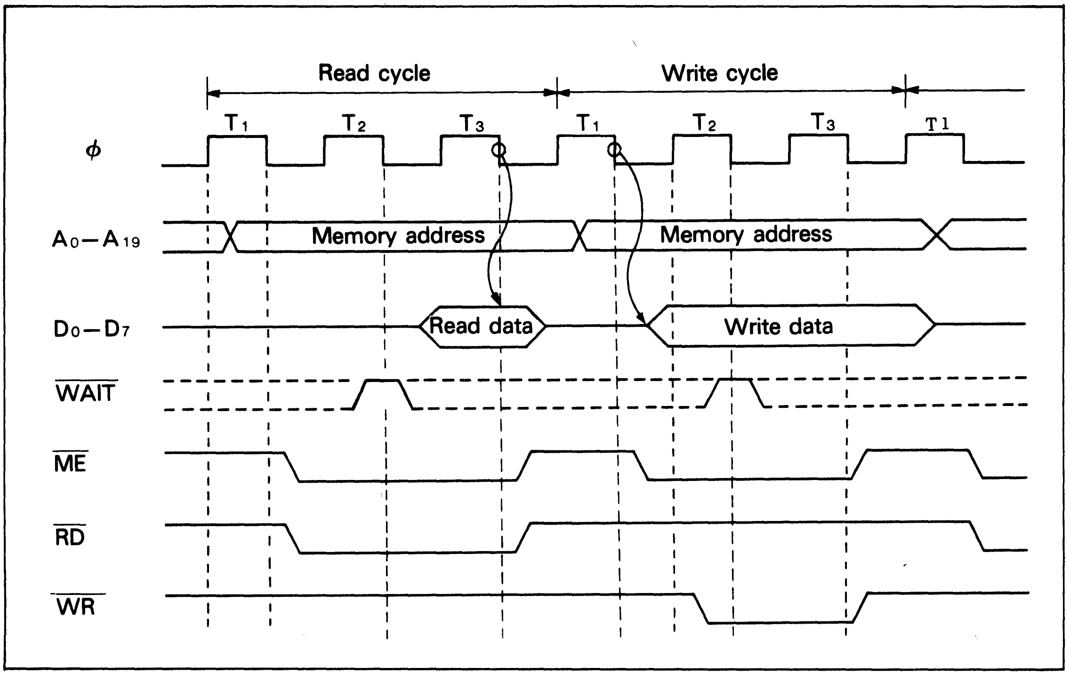


Figure 3-3. Memory Read/Write Timing (without Wait State)

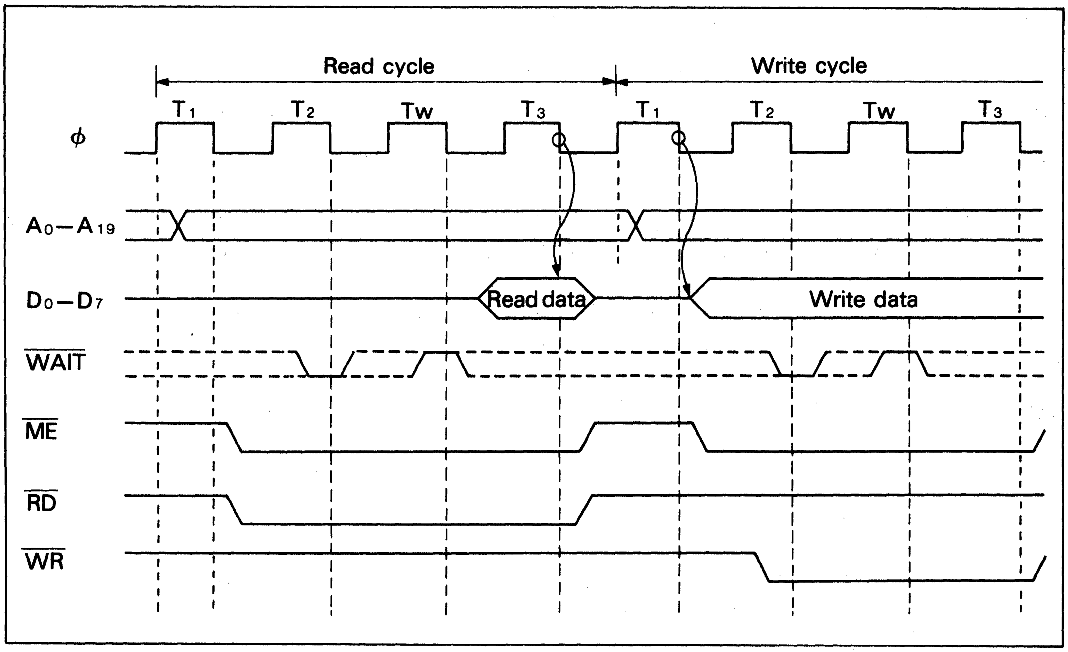
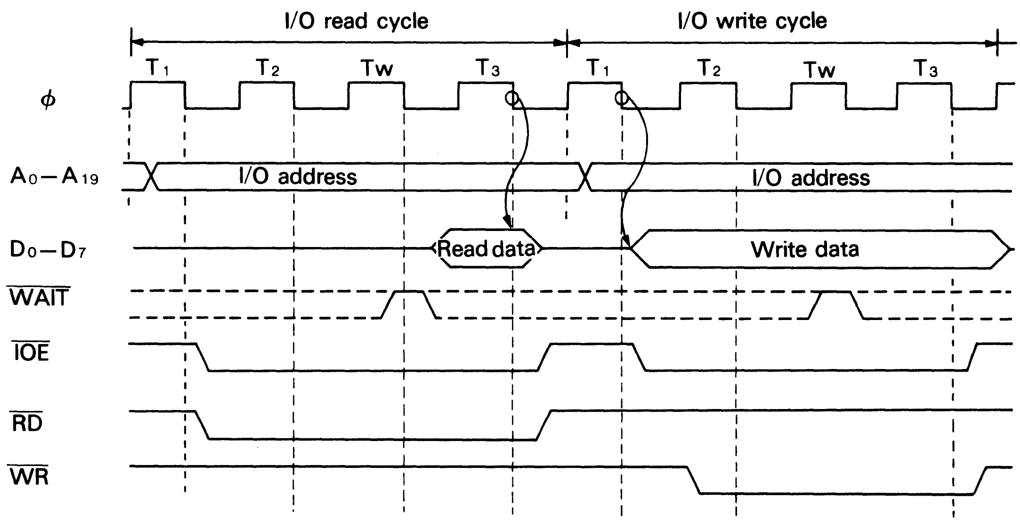


Figure 3-4. Memory Read/Write Timing (with Wait State)

3.3 I/O Read/Write Timing

I/O instructions cause data read/write transfers which differ from memory data transfers in the following three ways. The $\overline{IOE}$ (I/O Enable) signal is asserted low instead of the $\overline{ME}$ signal. The 16-bit I/O address is not translated by the MMU and A_{16}-A_{19} are held low. At least one wait state (Tw) is always inserted for I/O read and write cycles (except internal I/O cycles).

Figure 2-5. shows I/O read/write timing with the automatically inserted wait state (Tw).



NOTE: $A_{16}-A_{19} = 0$ for I/O cycles

Figure 3-5. I/O Read/Write Timing

3.4 Basic Instruction Timing

An instruction may consist of a number of machine cycles including opcode fetch, operand fetch, and data read/write cycles. An instruction may also include cycles for internal processing during which the bus is idle.

The example in figure 3-6. illustrates the bus timing for the data transfer instruction LD (IX+d), g. This instruction moves the contents of a CPU register (g) to the memory location with address computed by adding an signed 8-bit displacement (d) to the contents of an index register (IX).

The instruction cycle starts with the two machine cycles to read the two bytes instruction opcode as indicated by $\overline{\text{LIR}}$ low. Next, the instruction operand (d) is fetched.

The external bus is idle while the CPU computes the effective address. Finally, the contents of the CPU register (g) are written into the computed memory location.

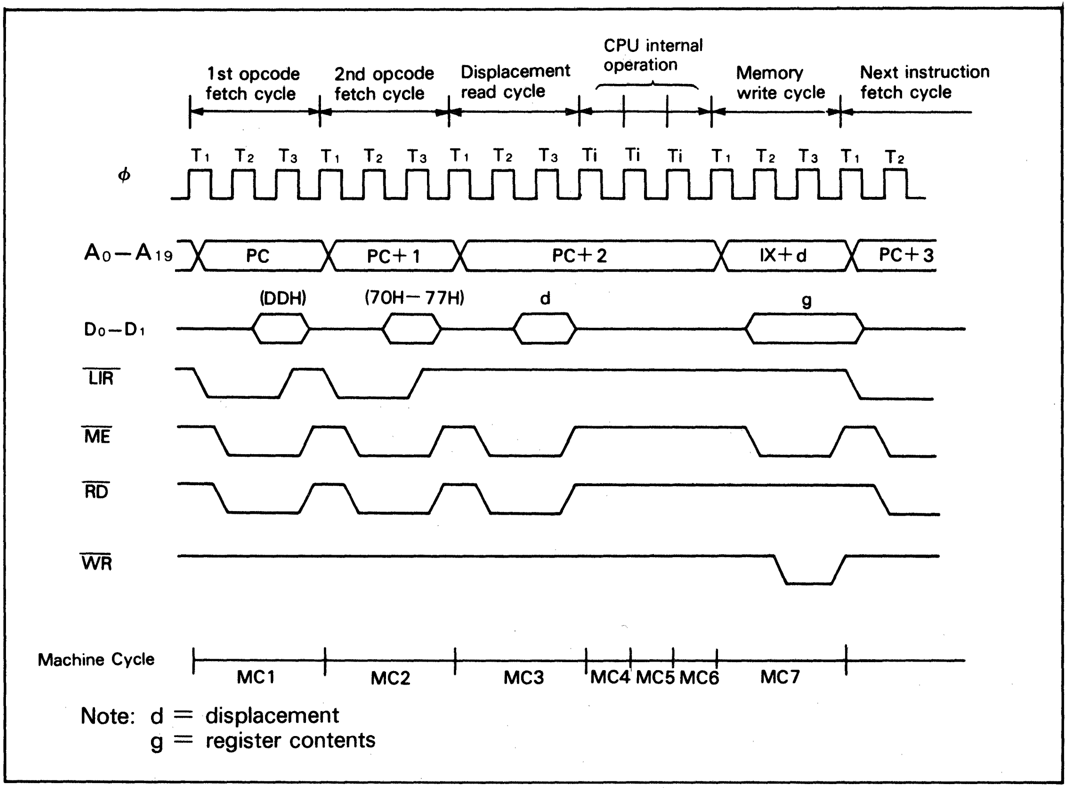


Figure 3-6. LD (IX+d), g Instruction Timing

3.5 Reset Timing

Figure 3-7. shows the HD647180X hardware reset timing. If the $\overline{RESET}$ pin is low for six or more clock cycles, processing is terminated and the HD647180X restarts execution from (logical and physical) address 00000H.

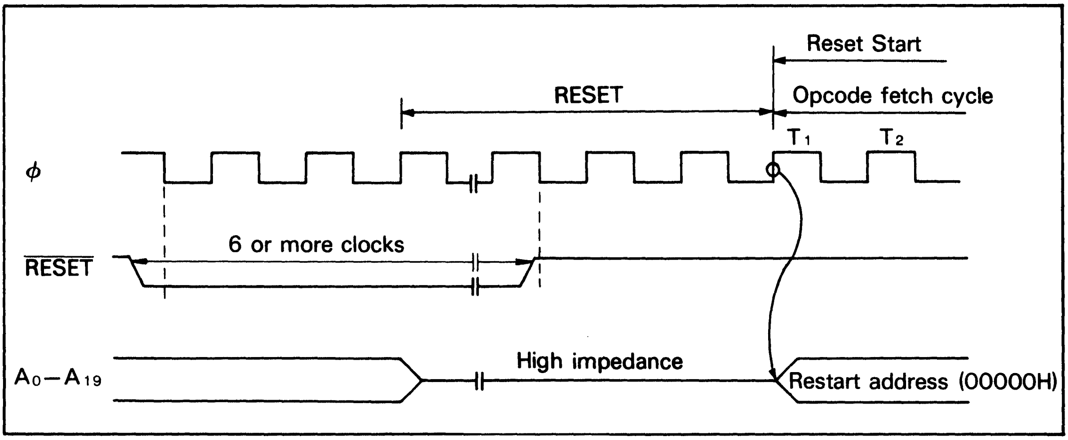


Figure 3-7. Reset Timing

3.6 $\overline{\text{BUSREQ}}/\overline{\text{BUSACK}}$ Bus Exchange Timing

The HD647180X can coordinate the exchange of control, address, and data bus ownership with another bus master. The alternate bus master can request the bus release by asserting the $\overline{\text{BUSREQ}}$ (bus request) input low. After the HD647180X releases the bus, it relinquishes control to the alternate bus master by asserting the $\overline{\text{BUSACK}}$ (bus acknowledge) output low.

The bus may be released by the HD647180X at the end of each machine cycle. In this context a machine cycle consists of a minimum of 3 clock cycles (more if wait states are inserted) for opcode fetch, memory read/write, and I/O read/write cycles. Except for these cases, a machine cycle corresponds to one clock cycle.

When the bus is released, the address (A_0-A_{19}), data (D_0-D_7), and control ($\overline{\text{ME}}$, $\overline{\text{IOE}}$, $\overline{\text{RD}}$, and $\overline{\text{WR}}$) signals are placed in the high-impedance state.

Note that dynamic RAM refresh is not performed when the HD647180X has released the bus. The alternate bus master must provide dynamic memory refresh if the bus is released for long periods of time.

Figure 3-8. illustrates $\overline{\text{BUSREQ}}/\overline{\text{BUSACK}}$ bus exchange during a memory read cycle. Figure 3-9. illustrates bus exchange when the bus release is requested during an HD647180X CPU internal operation. $\overline{\text{BUSREQ}}$ is sampled at the falling edge of the system clock prior to T_3, Ti, and Tx (bus release state). If $\overline{\text{BUSREQ}}$ is asserted low at the falling edge of the clock state prior to Tx, another Tx is executed.

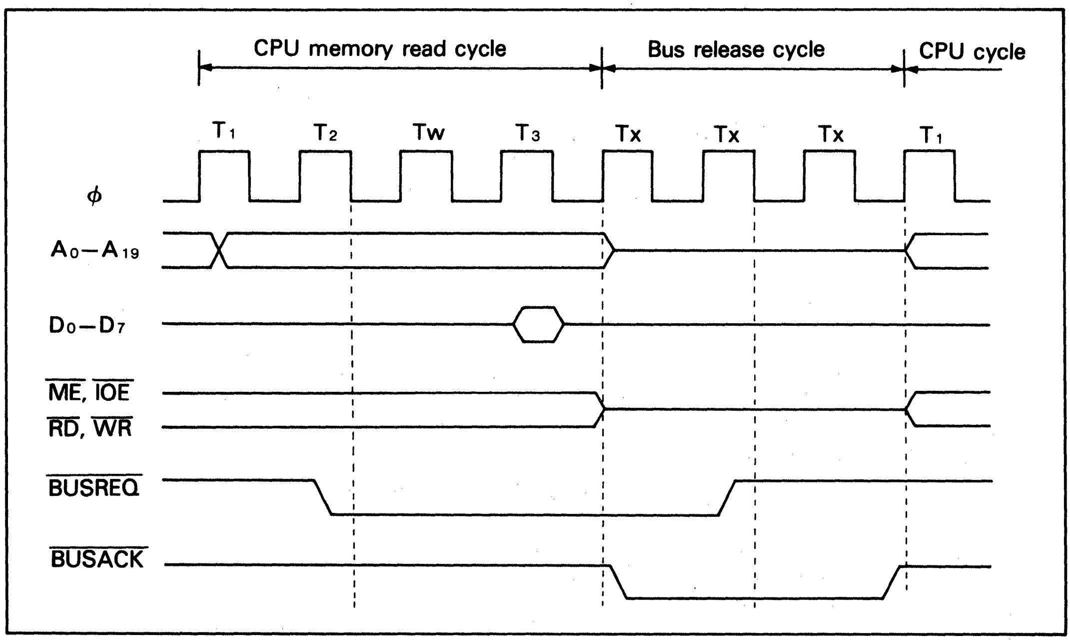


Figure 3-8. Bus Exchange Timing (1)

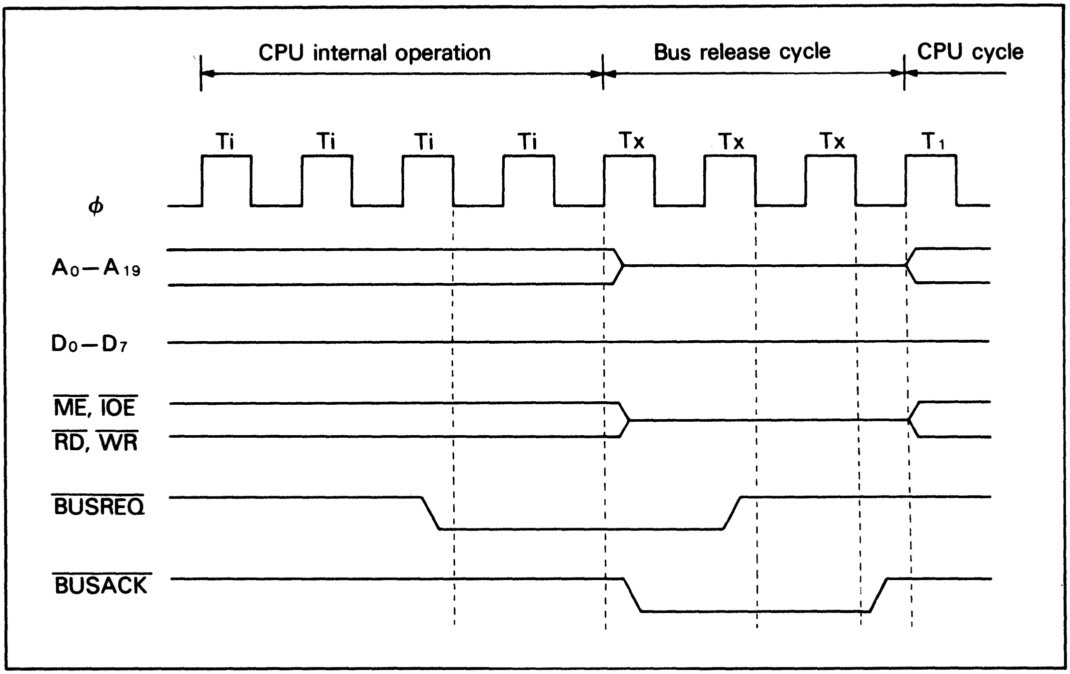


Figure 3-9. Bus Exchange Timing (2)

3.7 Z80-Type Bus Interface

3.7.1. $\overline{LIR}$, $\overline{IOE}$, and $\overline{RD}$ Signal Control

The $\overline{LIR}$, $\overline{IOE}$, and $\overline{RD}$ signals are controlled through the operation mode control register figure 3-10.

Operation Mode Control Register (OMCR: I/O address=3EH)

bit	7	6	5	4	3	2	1	0
	LIRE	$\overline{LIRTE}$	$\overline{IOC}$	—	—	—	—	—
	R/W	W	R/W					

Figure 3-10. Operation Mode Control Register

LIRE: LIR Enable (Bit 7): LIRE controls the $\overline{\text{LIR}}$ output and is set to 1 during RESET.

When LIRE $=1$, the $\overline{\text{LIR}}$ output is asserted low in the following cases:

- Opcode fetch cycles
- The acknowledge cycle of $\overline{\text{INT}_0}$
- The first machine cycle of the $\overline{\text{NMI}}$ acknowledge cycle

When LIRE $=0$, the $\overline{\text{LIR}}$ output is normally inactive (high). The $\overline{\text{LIR}}$ is asserted low only in the following cases:

- The second opcode fetch cycle of RETI (Please see 3.7.2 RETI Instruction)
- The acknowledge cycle of $\overline{\text{INT}_0}$

This mode is used to interface with Z80 peripheral LSIs using daisy chain interrupt.

$\overline{\text{LIRTE}}$: $\overline{\text{LIR}}$ Temporary Enable (Bit 6): $\overline{\text{LIRTE}}$ activates the $\overline{\text{LIR}}$ output temporarily. $\overline{\text{LIRTE}}$ is always read as 1 and is set to 1 during RESET. This bit resets Z80's PIO internal states after internal control register is set when daisy chain interrupt is used.

When $\overline{\text{LIRTE}}$ set to 1, there is no effect and the $\overline{\text{LIR}}$ output is subject to the LIRE bit.

When $\overline{\text{LIRTE}}$ set to 0,

- When the LIRE bit is 1, the $\overline{\text{LIR}}$ output is not affected by this write operation.
- When the LIRE bit is 0, the $\overline{\text{LIR}}$ output is temporarily asserted low in one opcode fetch cycle just after 0 is written to $\overline{\text{LIRTE}}$. The timing is shown in figure 3-11.

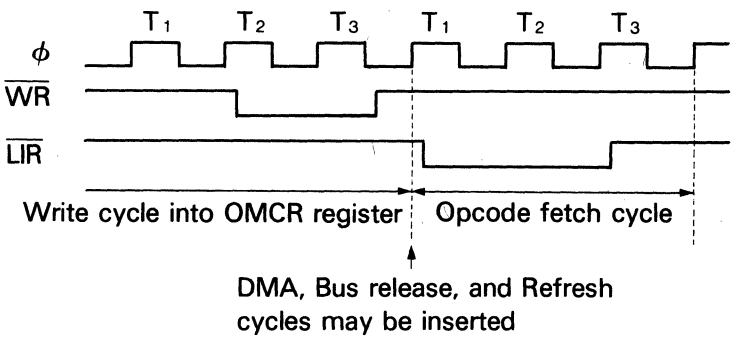


Figure 3-11. Writing 0 TO $\overline{\text{LIRTE}}$ When LIRE $=$ 0

$\overline{IOC}$: I/O Compatibility (Bit 5): $\overline{IOC}$ controls $\overline{IOE}$ and $\overline{RD}$ output and is set to 1 during reset.
When $\overline{IOC}=1$
In an I/O read cycle, $\overline{IOE}$ and $\overline{RD}$ signals go to low at a falling edge of T_1. The timing is shown in figure 3-12.

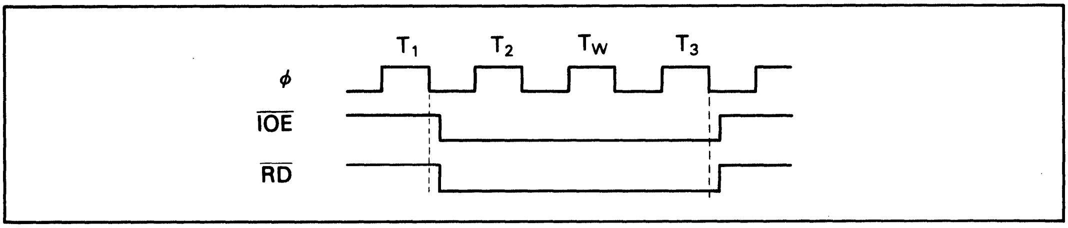


Figure 3-12. I/O Read Cycle When $\overline{IOC}$ = 1

In an I/O write cycle, $\overline{IOE}$ signal goes to low at a falling edge of T_1. The timing is shown in figure 3-13.

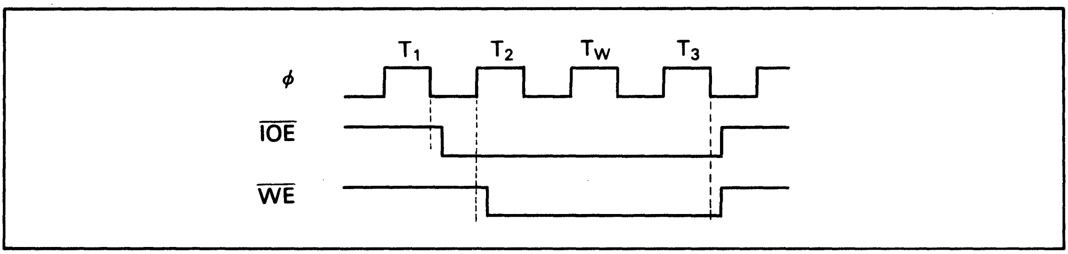


Figure 3-13. I/O Write Cycle When $\overline{IOC}$ = 1

When $\overline{IOC}=0$, the $\overline{IOE}$ and $\overline{RD}$ outputs are compatible with the Z80's peripheral LSIs. In an I/O read cycle, $\overline{IOE}$ and $\overline{RD}$ signals go to low at a rising edge of T_2. The timing is shown in figure 3-14.

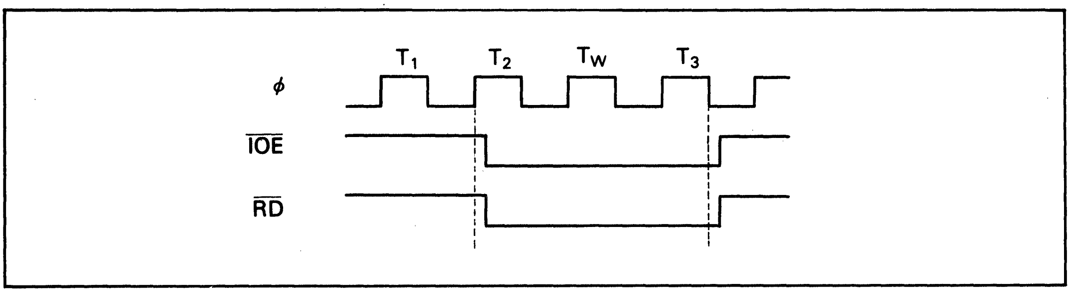


Figure 3-14. I/O Read Cycle When $\overline{IOC}$ = 0

In an I/O write cycle, $\overline{IOE}$ signal goes to low at a rising edge of T_2. The timing is shown in figure 3-15.

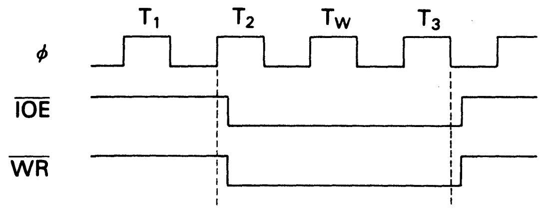


Figure 3-15. I/O Write Cycle When $\overline{IOC}$ = 0

3.7.2 RETI Instruction

The CPU reads the opcode, EDH and 4DH, twice as shown in figure 3-16.

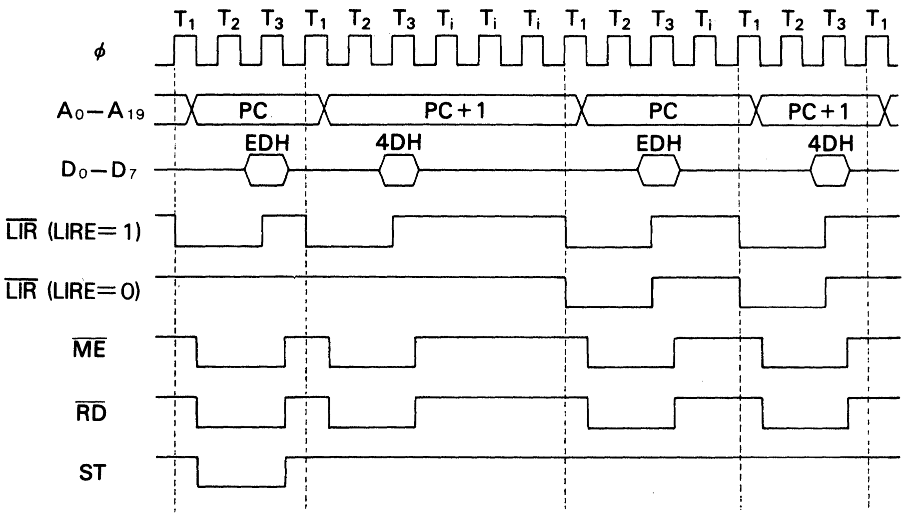


Figure 3-16. Operation of RETI Instruction

The number of states and machine cycles is shown in table 3-1.

Table 3-1. Number of States and Machine Cycles

Number of states	Number of machine cycles
22	10

Note: Interrupt request during RETI instruction: The CPU can't be interrupted between the first and the second read of the opcode. The CPU can be interrupted after it completes the unstack operation.

Please set the bits in OMCR according to table 3-2.

Table 3-2. Setting the Operation Mode Control Register

			OMCR		
Daisy Chain	CTC	PIO	LIRE	$\overline{\text{LIRTE}}$	$\overline{\text{IOC}}$
Yes	Yes	Yes	0	write 0	0
		No	0	No operation	0
	No	Yes	0	write 0	0/1
		No	0	No operation	0/1
No	Yes		1	No operation	0
	No		1	No operation	0/1

SECTION 4. WAIT STATE GENERATOR

4.1 Wait State Timing

To ease interfacing with slow memory and I/O devices, the HD647180X uses wait states (Tw) to extend bus cycle timing. A wait state is inserted based on the combined (logical OR) state of the external $\overline{\text{WAIT}}$ input and an internal programmable wait state (Tw) generator. Wait states (Tw) can be inserted in both CPU execution and DMA transfer cycles.

4.2 $\overline{\text{WAIT}}$ Input

When the external $\overline{\text{WAIT}}$ input is asserted low, a wait state (Tw) is inserted between T_2 and T_3 to extend the bus cycle duration. The $\overline{\text{WAIT}}$ input is sampled at the falling edge of the system clock in T_2 or Tw. If the $\overline{\text{WAIT}}$ input is asserted low at the falling edge of the system clock in Tw, another Tw is inserted into the bus cycle. Note that $\overline{\text{WAIT}}$ input transitions must meet specified set-up and hold times. This can easily be accomplished by externally synchronizing $\overline{\text{WAIT}}$ input transitions with the rising edge of the system clock. Figure 4-1. shows $\overline{\text{WAIT}}$ timing.

Dynamic RAM refresh is not performed during wait states (Tw) and thus systems designs which uses the automatic refresh function must consider the affects of the occurrence and duration of wait states (Tw).

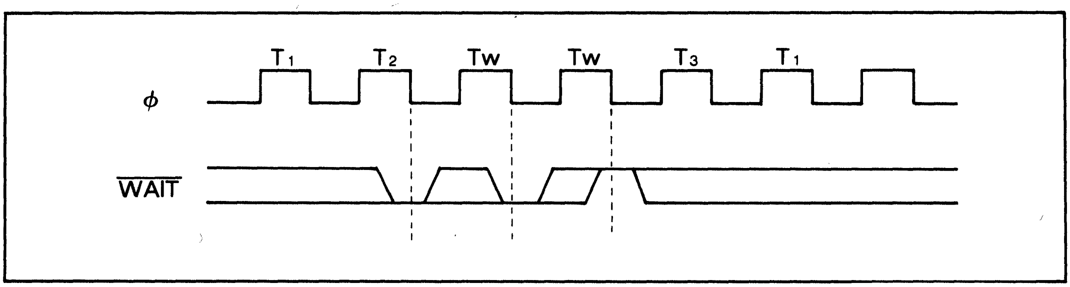


Figure 4-1. $\overline{\text{WAIT}}$ Timing

4.3 Programmable Wait State Insertion

In addition to the $\overline{\text{WAIT}}$ input, wait states (Tw) can also be programmably inserted using the HD647180X on-chip wait state generator. Wait state (Tw) timing applies for both CPU execution and on-chip DMAC cycles.

By programming the 4 significant bits of the DMA/wait control register (DCNTL), the number of wait states (Tw) automatically inserted in memory and I/O cycles can be separately specified. Bits 4, 5 specify the number of wait states (Tw) inserted for I/O access and bits 6, 7 specify the number of wait states (Tw) inserted for memory access (figure 4-2.).

DMA/wait Control Register (DCNTL: I/O Address = 32H)

bit	7	6	5	4
	MWI1	MWI0	IWI1	IWI0
	R/W	R/W	R/W	R/W

Figure 4-2. DMA/Wait Control Register

The number of wait states (Tw) inserted in a specific cycle is the maximum of the number requested by the $\overline{\text{WAIT}}$ input, and the number automatically generated by the on-chip wait state generator.

MWI1, MWI0: Memory Wait Insertion (Bit 7,6): For CPU and DMAC cycles which access memory (including memory-mapped I/O), 0 to 3 wait states may be automatically inserted depending on the programmed value in MWI1 and MWI0 (table 4-1.).

Table 4-1. Memory Wait Insertion

MWI1	MWI0	Number of Wait States
0	0	0
0	1	1
1	0	2
1	1	3

IWI1, IWI0: I/O Wait Insertion (Bit 5,4): For CPU and DMA cycles which access external I/O (and interrupt acknowledge cycles), 1 to 6 wait states (Tw) may be automatically inserted depending on the programmed value in IWI1 and IWI0 (table 4-2.).

Table 4-2. I/O Wait Insertion

		Number of Wait States				
IWI1	IWI0	External I/O Register Access	Internal I/O Registers Access	$\overline{\text{INT}}_0$ Interrupt Acknowledge Cycles ($\overline{\text{LIR}}$ is low)	$\overline{\text{INT}}_1$, $\overline{\text{INT}}_2$ and Internal Interrupts Acknowledge Cycles (Note 2)	$\overline{\text{NMI}}$ Interrupt Acknowledge Cycles ($\overline{\text{LIR}}$ is low) (Note 2)
0	0	1	0	2	2	0
0	1	2	(Note 1)	4		
1	0	3		5		
1	1	4		6		

Notes: 1. 0−4 wait states are always inserted, regardless of the IWI0-IWI1 bits' value, when accessing the following: the ASCI receive data register, ASCI transmit data register, CSI/O transmit/receive data register, timer 1 data register, timer 1 reload timer, timer 2 input capture register, timer 2 free running counter, timer 2 control status register 1, timer 2 control status register 2, and timer 2 output compare register.
These 0−4 wait cycles are inserted to synchronize the CPU and I/O functions depending on the CPU and I/O status.
2. For interrupt acknowledge cycles in which $\overline{\text{LIR}}$ is high, such as interrupt vector table read and PC stacking cycles, memory access timing applies.

4.4 $\overline{\text{WAIT}}$ Input and Reset

During reset, MWI1, MWI0, IWI1, and IWI0 are all set to 1, selecting the maximum number of wait states (Tw) (3 for memory accesses, 4 for external I/O accesses).

4.5 $\overline{\text{WAIT}}$ State Generator Note

$\overline{\text{WAIT}}$ states are automatically inserted in mode 0 (single-chip mode). Therefore, the MWI0 and MWI1 bits should be cleared to 0 in mode 0.

SECTION 5. HALT AND LOW POWER OPERATION MODES

The HD647180X can operate in 4 different modes. Halt mode, I/O stop mode and two low power operation modes: Sleep and System stop. Note that in all operating modes, the basic CPU clock (XTAL, EXTAL) must remain active.

5.1 Halt mode

Halt mode is entered by execution of the HALT instruction (opcode = 76H) and has the following characteristics:

- The internal CPU clock remains active
- All internal and external interrupts can be received
- Bus exchange ($\overline{BUSREQ}$ and $\overline{BUSACK}$) can occur
- Dynamic RAM refresh cycle ($\overline{REF}$) insertion continues at the programmed interval
- I/O operations (ASCI, CSI/O and PRT) continue
- The DMAC can operate
- The $\overline{HALT}$ output pin is asserted LOW
- The external bus activity consists of repeated 'dummy' fetches of the opcode following the HALT instruction

Essentially, the HD647180X operates normally in halt mode, except that instruction execution is stopped.

Halt mode can be exited in the following two ways.

5.1.1 Reset Exit from Halt Mode

If the $\overline{RESET}$ input is asserted low for at least six clock cycles, the HD647180X exits halt mode and the normal Reset sequence (restart at address 00000H) is initiated.

5.1.2 Interrupt Exit from Halt Mode

When an internal or external interrupt is generated, the HD647180X exits halt mode and the normal interrupt response sequence is initiated.

If the interrupt source is masked (individually by enable bit, or globally by IEF_1 state), the HD647180X remains in halt mode. However, $\overline{NMI}$ interrupt will initiate the normal $\overline{NMI}$ interrupt response sequence independent of the state of IEF_1.

Halt timing is shown in figure 5-1.

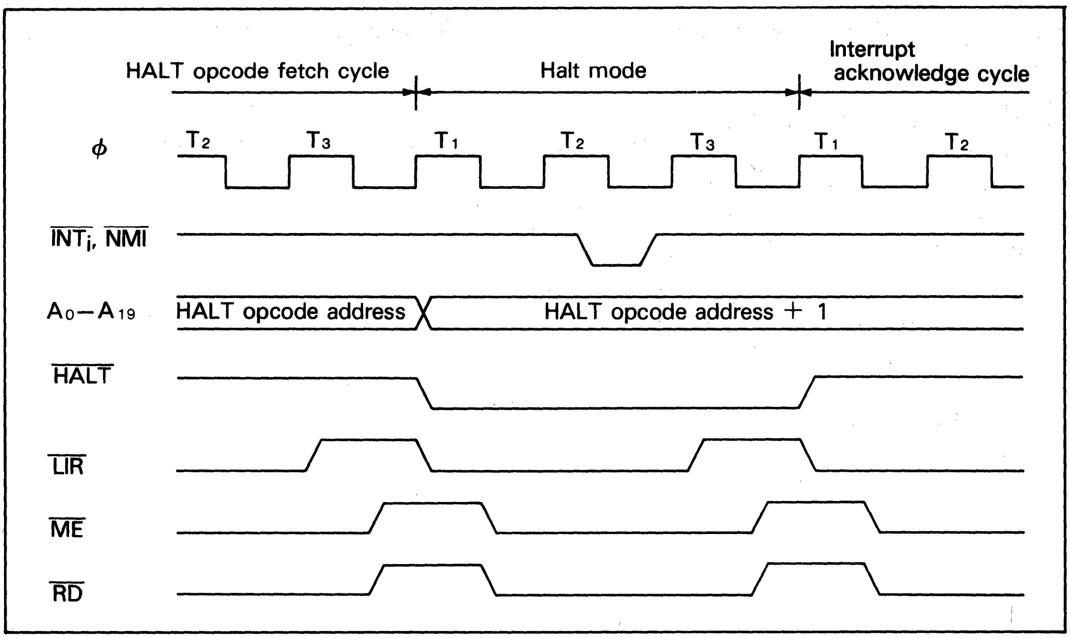


Figure 5-1. Halt Timing

5.2 Sleep Mode

The HD647180X enters sleep mode by executing of the 2-byte SLP instruction. Sleep mode has the following characteristics.

- The internal CPU clock stops, reducing power consumption
- The internal crystal oscillator does not stop
- Internal and external interrupt inputs can be received
- DRAM refresh cycles stop
- I/O operations using on-chip peripherals continue
- The internal DMAC stops
- $\overline{\text{BUSREQ}}$ can be received and acknowledged.
- Address outputs go high and all other control signal output become inactive high
- Data bus, goes high impendace

Sleep mode is exited in one of following two ways.

5.2.1 Reset Exit from Sleep Mode

If the $\overline{\text{RESET}}$ input is held low for at least six clock cycles, the HD647180X will exit sleep mode and begin the normal reset sequence with execution starting at address (logical and physical) 00000H.

5.2.2 Interrupt Exit from Sleep Mode

The HD647180X exits sleep mode by detecting an external ($\overline{NMI}$, $\overline{INT_0}$, $\overline{INT_1}$, $\overline{INT_2}$) or internal (ASCI, CSI/O, PRT) interrupt.

In the case of $\overline{NMI}$, the CPU exits sleep mode and begins the normal $\overline{NMI}$ interrupt response sequence.

In the case of all other interrupts, the interrupt response depends on the state of the global interrupt enable flag (IEF_1) and the individual interrupt source enable bit.

If the individual interrupt condition is disabled by the corresponding enable bit, that interrupt is ignored and the CPU remains in the sleep state.

If the individual interrupt condition is enabled, the response to that interrupt depends on the global interrupt enable flag (IEF_1). If interrupts are globally enabled ($IEF_1=1$) and an individually enabled interrupt occurs, the CPU exits sleep mode and executes the appropriate normal interrupt response sequence.

If interrupts are globally disabled ($IEF_1=0$) and an individually enabled interrupt occurs, the CPU exits sleep mode and instruction execution begins with the instruction following the SLP instruction. Note that this provides a technique for synchronization with high-speed external events without incurring the latency imposed by an interrupt response sequence.

Figure 5-2. shows sleep timing.

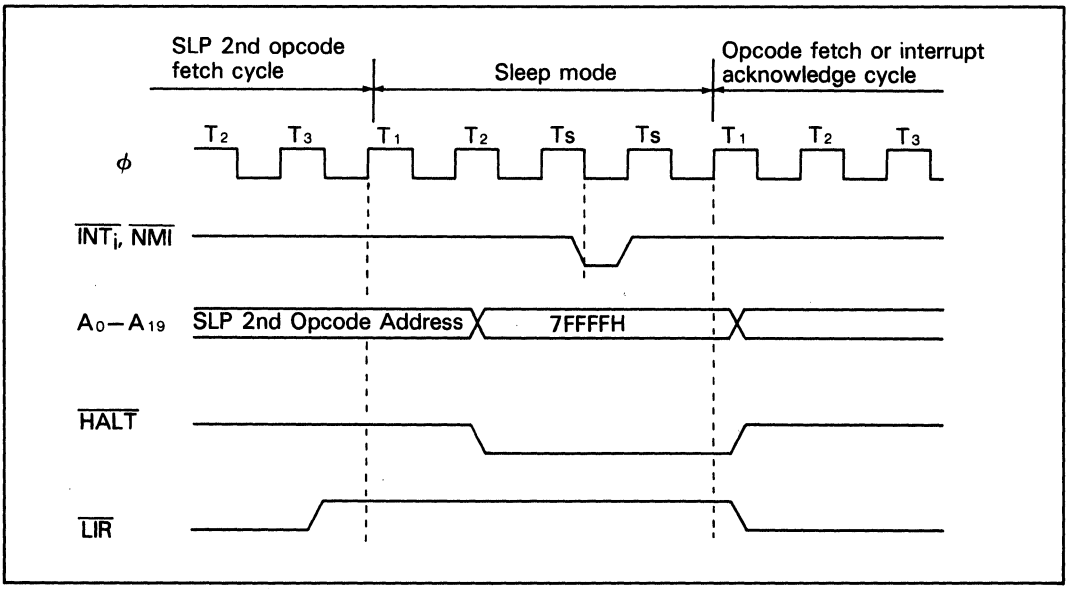


Figure 5-2. Sleep Timing

If interrupt requests occur while the CPU fetches a SLP instruction, $\overline{HALT}$ output goes low for only 1 state in sleep mode HD647180X as shown in figure 5-3.

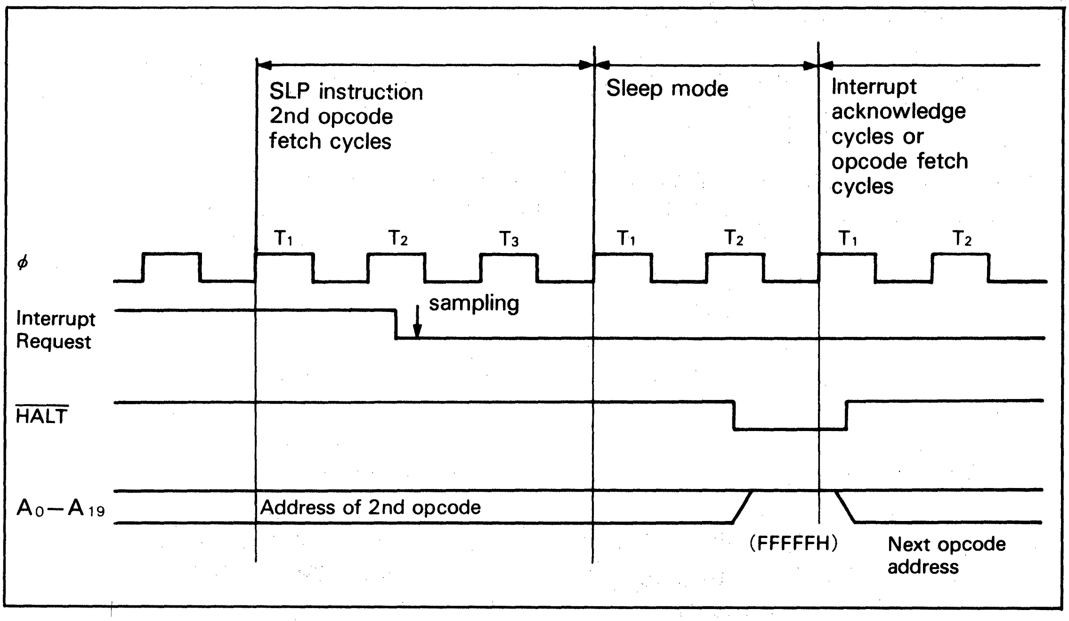


Figure 5-3. $\overline{HALT}$ Output

5.3 I/O Stop Mode

I/O stop mode is selected by setting the IOSTP bit of the I/O control register (ICR) to 1. In this case, on-chip I/O (ASCI, CSI/O, PRT) stops operating. However, the CPU continues to operate. Recovery from I/O stop mode is caused by resetting the IOSTP bit in ICR to 0.

5.4 System Stop Mode

System stop mode is the combination of sleep and I/O stop modes. System stop mode is selected by setting the IOSTP bit in ICR to 1 followed by executing the SLP instruction. In this mode, on-chip I/O and CPU stop operating, reducing power consumption. Recovery from system stop mode is the same as recovery from sleep mode, noting that internal I/O sources (disabled by I/O stop) cannot generate a recovery interrupt.

SECTION 6. INTERNAL I/O REGISTERS

The HD647180X internal I/O registers occupy 128 I/O addresses (including reserved addresses). These registers access the internal I/O modules (ASCI, CSI/O, PRT, PT2, I/O port, Analog comparator) and control functions (DMAC, DRAM refresh, interrupts, wait state generator, MMU, and I/O relocation).

To avoid address conflicts with external I/O, the HD647180X internal I/O addresses can be relocated on 128 bytes boundaries within the bottom 256 bytes of the 64-kbyte I/O address space.

6.1 I/O Control Register (ICR)

ICR allows relocation of the internal I/O addresses (figure 6-1.). ICR also controls enabling/disabling of the I/O stop mode.

I/O Control Register (ICR: I/O Address = 3FH)

bit	7	6	5	4	3	2	1	0
	IOA7	–	IOSTP	–	–	–	–	–
	R/W		R/W					

Figure 6-1. I/O Control Register

6.1.1 IOA7: I/O Address Relocation (Bits 7)

IOA7 relocates internal I/O as shown in figure 6-2. Note that the high-order 8 bits of 16-bit internal I/O addresses are always 0. IOA7 is cleared to 0 during reset.

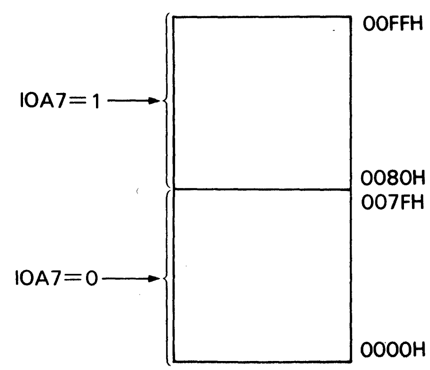


Figure 6-2. Internal I/O Address Relocation

6.1.2 IOSTP: I/O Stop Mode (Bit 5)

I/O stop mode is enabled when IOSTP is set to 1. Normal I/O operation resumes when IOSTP is reset to 0. IOSTP is cleared to 0 during reset.

6.2 Internal I/O Register Address Map

The internal I/O register addresses are shown in Table 6-1. These addresses are relative to the 128-byte boundary base address specified in ICR.

Table 6-1. Internal I/O Register Address Map

	Register	Address Binary	Address Hexadecimal
ASCI	ASCI Control Register A Ch 0	X0000000	00H
	ASCI Control Register A Ch 1	X0000001	01H
	ASCI Control Register B Ch 0	X0000010	02H
	ASCI Control Register B Ch 1	X0000011	03H
	ASCI Status Register Ch 0	X0000100	04H
	ASCI Status Register Ch 1	X0000101	05H
	ASCI Transmit Data Register Ch 0	X0000110	06H
	ASCI Transmit Data Register Ch 1	X0000111	07H
	ASCI Receive Data Register Ch 0	X0001000	08H
	ASCI Receive Data Register Ch 1	X0001001	09H
CSI/O	CSI/O Control Register	X0001010	0AH
	CSI/O Transmit/Receive Data Register	X0001011	0BH
Timer	Timer Data Register Ch 0L	X0001100	0CH
	Timer Data Register Ch 0H	X0001101	0DH
	Reload Register Ch 0L	X0001110	0EH
	Reload Register Ch 0H	X0001111	0FH
	Timer Control Register	X0010000	10H
	Reserved	X0010001-X0010011	11H-13H
	Timer Data Register Ch 1L	X0010100	14H
	Timer Data Register Ch 1H	X0010101	15H
	Reload Register Ch 1L	X0010110	16H
	Reload Register Ch 1H	X0010111	17H
Others	Free Running Counter	X0011000	18H
	Reserved	X0011001-X0011111	19H-1FH

Table 6-1. Internal I/O Register Address Map (cont)

	Register	Address Binary	Hexadecimal
DMA	DMA Source Address Register Ch 0L	X0100000	20H
	DMA Source Address Register Ch 0H	X0100001	21H
	DMA Source Address Register Ch 0B	X0100010	22H
	DMA Destination Address Register Ch 0L	X0100011	23H
	DMA Destination Address Register Ch 0H	X0100100	24H
	DMA Destination Address Register Ch 0B	X0100101	25H
	DMA Byte Count Register Ch 0L	X0100110	26H
	DMA Byte Count Register Ch 0H	X0100111	27H
	DMA Memory Address Register Ch 1L	X0101000	28H
	DMA Memory Address Register Ch 1H	X0101001	29H
	DMA Memory Address Register Ch 1B	X0101010	2AH
	DMA I/O Address Register Ch 1L	X0101011	2BH
	DMA I/O Address Register Ch 1H	X0101100	2CH
	Reserved	X0101101	2DH
	DMA Byte Count Register Ch 1L	X0101110	2EH
	DMA Byte Count Register Ch 1H	X0101111	2FH
	DMA Status Register	X0110000	30H
	DMA Mode Register	X0110001	31H
	DMA/Wait Control Register	X0110010	32H
INT	IL Register (Interrupt Vector Low Register)	X0110011	33H
	INT/Trap Control Register	X0110100	34H
	Reserved	X0110101	35H
Refresh	Refresh Control Register	X0110110	36H
	Reserved	X0110111	37H
MMU	MMU Common Base Register	X0111000	38H
	MMU Bank Base Register	X0111001	39H
	MMU Common/Bank Area Register	X0111010	3AH
I/O	Reserved	X0111011- X0111101	3BH-3DH
	Operation Mode Control Register	X0111110	3EH
	I/O Control Register	X0111111	3FH

Table 6-1. Internal I/O Register Address Map (cont)

	Register	Address Binary	Hexadecimal
Timer 2	Timer 2 Free Running Counter L	X1000000	40H
	Timer 2 Free Running Counter H	X1000001	41H
	Timer 2 Output Compare Register 1L	X1000010	42H
	Timer 2 Output Compare Register 1H	X1000011	43H
	Timer 2 Output Compare Register 2L	X1000100	44H
	Timer 2 Output Compare Register 2H	X1000101	45H
	Timer 2 Input Capture Register L	X1000110	46H
	Timer 2 Input Capture Register H	X1000111	47H
	Timer 2 Control/Status Register 1	X1001000	48H
	Timer 2 Control/Status Register 2	X1001001	49H
	Reserved	X1001010-X1001111	4AH-4FH
Others	Comparator Control/Status Register	X1010000	50H
	RAM Control Register	X1010001	51H
	Reserved	X1010010	52H
	Port A Disable Register	X1010011	53H
	Reserved	X1010100-X1011111	54H-5FH
I/O Port	Port A Input Data Register	X1100000	60H
	Port A Output Data Register	X1100000	60H
	Port B Input Data Register	X1100001	61H
	Port B Output Data Register	X1100001	61H
	Port C Input Data Register	X1100010	62H
	Port C Output Data Register	X1100010	62H
	Port D Input Data Register	X1100011	63H
	Port D Output Data Register	X1100011	63H
	Port E Input Data Register	X1100100	64H
	Port E Output Data Register	X1100100	64H
	Port F Input Data Register	X1100101	65H
	Port F Output Data Register	X1100101	65H
	Port G Input Data Register	X1100110	66H
	Reserved	X1100111-X1101111	67H-6FH

Table 6-1. Internal I/O Register Address Map (cont)

	Register	Address Binary	Address Hexadecimal
I/O Port (cont)	Data Direction Register A	X1110000	70H
	Data Direction Register B	X1110001	71H
	Data Direction Register C	X1110010	72H
	Data Direction Register D	X1110011	73H
	Data Direction Register E	X1110100	74H
	Data Direction Register F	X1110101	75H
	Reserved	X1110110-X1111111	76H-7FH

6.3 I/O Addressing Notes

The internal I/O register addresses are located in the I/O address space from 0000H to 00FFH (16-bit I/O addresses). Thus, to access the internal I/O registers (using I/O instructions), the high-order 8 bits of the 16-bit I/O address must be 0.

The conventional I/O instructions (OUT (m),A; IN A,(m); OUTI; INI; etc.) place the contents of a CPU register on the high-order 8 bits of the address bus, and thus may be difficult to use for accessing internal I/O registers.

For efficient internal I/O register access, a number of new instructions have been added, which force the high-order 8 bits of the 16-bit I/O address to 0. These instructions are IN0, OUT0, OTIM, OTIMR, OTDM, OTDMR, and TSTIO (see section 22, Instruction Set).

Note that when an internal I/O register is written to, the same I/O write occurs on the external bus. However, the duplicate external I/O write cycle will exhibit internal I/O write cycle timing. For example, the $\overline{\text{WAIT}}$ input and programmable wait state generator are ignored. Similarly, internal I/O read cycles also cause a duplicate external I/O read cycle. However, the external read data is ignored by the HD647180X.

Normally, external I/O addresses should be chosen to avoid overlap with internal I/O addresses, causing duplicate I/O accesses.

SECTION 7. MEMORY MANAGEMENT UNIT (MMU)

The HD647180X contains an on-chip MMU which translates the CPU 64-kbyte (16-bit addresses: 0000H to FFFFH) logical memory address space into a 1-Mbyte (20-bit addresses: 00000H to FFFFFH) physical memory address space. Address translation occurs internally in parallel with other CPU operations.

7.1 Logical Address Spaces

The 64-kbyte CPU logical address space is interpreted by the MMU as consisting of up to three separate logical address areas, common area 0, bank area, and common area 1.

As shown in figure 7-1., a variety of logical memory configurations are possible. The boundaries between the common and bank areas can be programmed with 4-kbyte resolution.

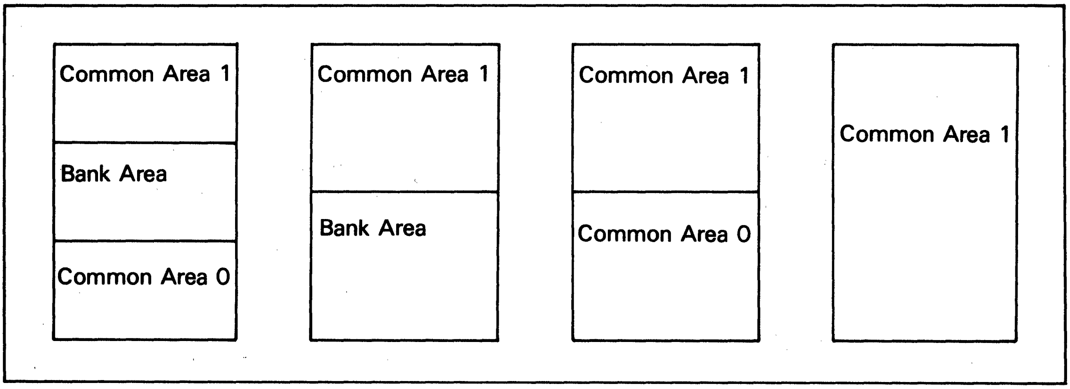


Figure 7-1. Logical Address Mapping Examples

7.2 Logical to Physical Address Translation

Figure 2-2. shows an example in which the three logical address space portions are mapped into a 1-Mbyte physical address space. The important points to note are that common and bank areas can overlap and that common area 1 and bank area can be freely relocated (on 4-kbyte physical address boundaries). common area 0 (if it exists) is always based at physical address 00000H.

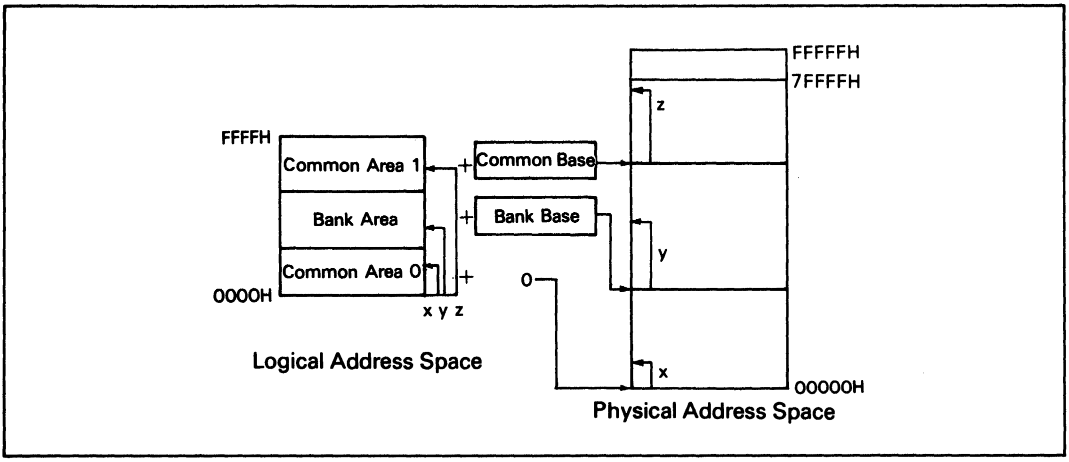


Figure 7-2. Logical to Physical Memory Mapping Example

7.3 MMU Block Diagram

The MMU figure 7-3. translates internal 16-bit logical addresses to external 20-bit physical addresses.

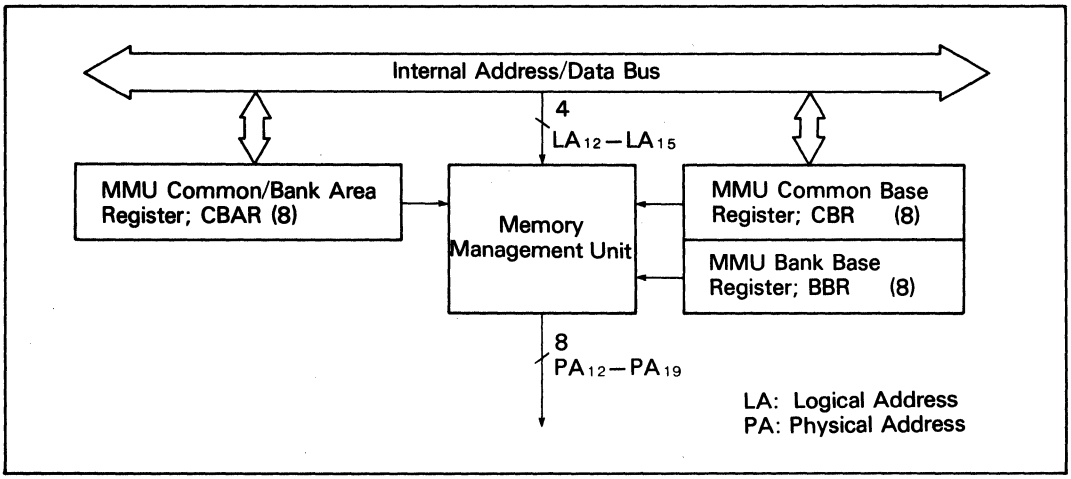


Figure 7-3. MMU Block Diagram

Whether address translation takes place depends on the type of CPU cycle.

7.3.1 Memory Cycles

Address translation occurs for all memory access cycles including instruction and operand fetches, memory data reads and writes, hardware interrupt vector fetch, and software interrupt restarts.

7.3.2 I/O Cycles

The MMU is logically bypassed for I/O cycles. The 16-bit logical I/O address space corresponds directly with the 16-bit physical I/O address space. The four high order bits (A_{16}-A_{19}) of the physical address are always 0 during I/O cycles (figure 7-4.).

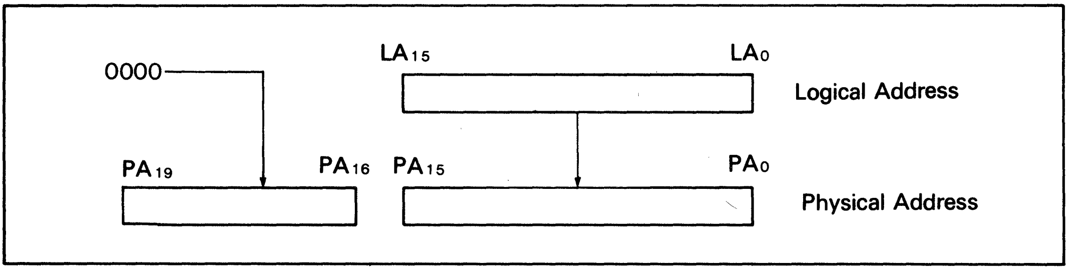


Figure 7-4. I/O Address Translation

7.3.3 DMA Cycles

When the HD647180X on-chip DMAC uses the external bus, the MMU is physically bypassed. The 20-bit source and destination registers in the DMAC are directly output on the physical address bus (A_0-A_{19}).

7.4 MMU Registers

Three MMU registers program a specific configuration of logical and physical memory.

1. MMU common/bank area register (CBAR)
2. MMU common base register (CBR)
3. MMU bank base register (BBR)

CBAR defines the logical memory organization, while CBR and BBR relocate logical areas within the 1-Mbyte physical address space. The resolution for both boundaries within the logical space and relocation within the physical space is 4-kbyte.

The CAR field of CBAR determines the start address of common area 1 (upper common) and by default, the end address of the bank area. The BAR field determines the start address of the bank area and by default, the end address of common area 0 (lower common).

The CA and BA fields of CBAR may be freely programmed subject only to the restriction that CA may never be less than BA. Figure 7-5. and figure 7-6. show examples of logical memory organizations associated with different values of CA and BA.

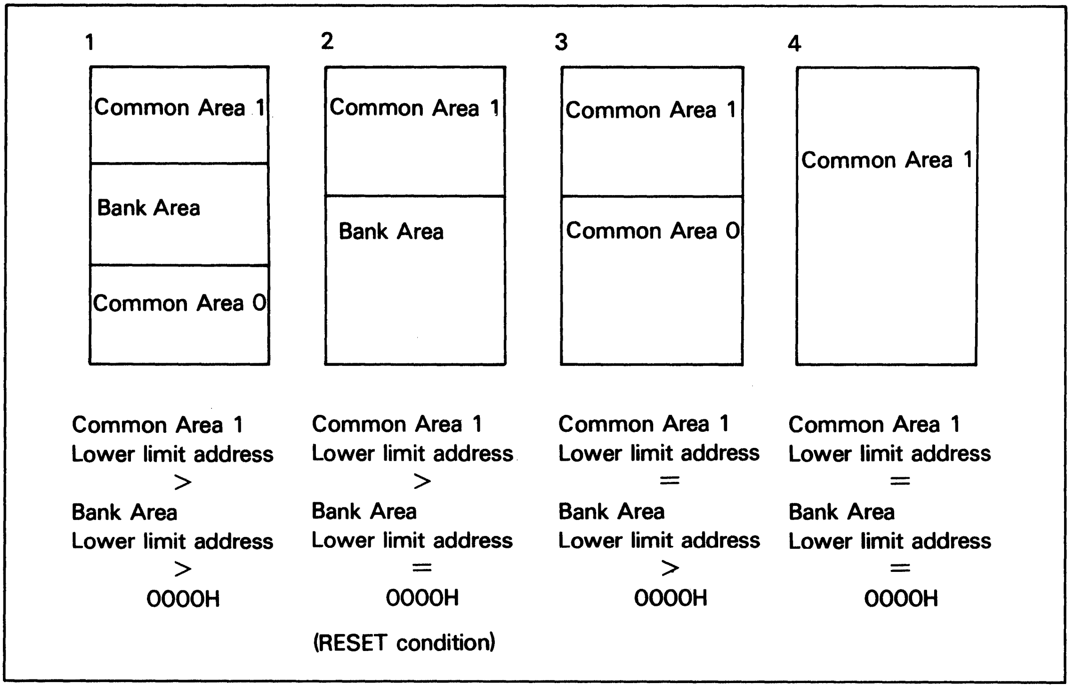


Figure 7-5. Logical Memory Organization

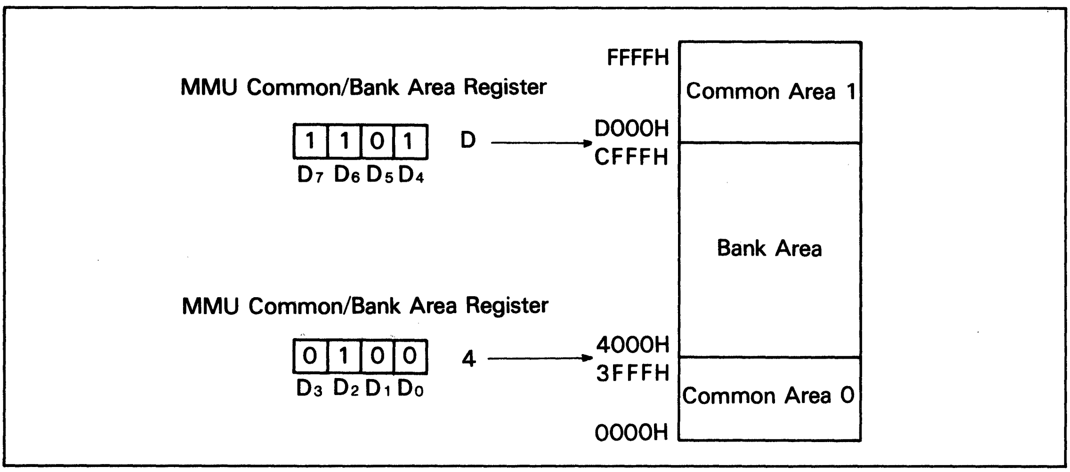


Figure 7-6. Logical Space Configuration (Example)

7.4.1 MMU Common/Bank Area Register (CBAR)

CBAR figure 7-7. specifies boundaries within the HD647180X 64-kbyte logical address space for up to three areas, common area 0, bank area and common area 1.

MMU Common/Bank Area Register (CBAR : I/O Address = 3AH)

bit	7	6	5	4	3	2	1	0
	CA3	CA2	CA1	CA0	BA3	BA2	BA1	BA0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Figure 7-7. MMU Common/Bank Area Register

CA3-CA0 (Bits 7-4): CA specifies the start (low) address (on 4-kbyte boundaries) for common area 1. This also determines the last address of the bank area. All bits of CA are set to 1 during reset.

BA3-BA0 (Bits 3-0): BA specifies the start (low) address (on 4-kbyte boundaries) for the bank area. This also determines the last address of common area 0. All bits of BA are reset to 0 during reset.

7.4.2 MMU Common Base Register (CBR)

CBR (figure 7-8.) specifies the base address (on 4-kbyte boundaries) used to generate a 20-bit physical address for common area 1 accesses. All bits of CBR are reset to 0 during reset.

MMU Common Base Register (CBR : I/O Address = 38H)

bit	7	6	5	4	3	2	1	0
	CB7	CB6	CB5	CB4	CB3	CB2	CB1	CB0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Figure 7-8. MMU Common Base Register

7.4.3 MMU Bank Base Register (BBR)

BBR (figure 7-9.) specifies the base address (on 4-kbyte boundaries) used to generate a 20-bit physical address for bank area accesses. All bits of BBR are reset to 0 during reset.

MMU Bank Base Register (BBR : I/O Address = 39H)

bit	7	6	5	4	3	2	1	0
	BB7	BB6	BB5	BB4	BB3	BB2	BB1	BB0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Figure 7-9. MMU Bank Base Register

7.5 Physical Address Translation

Figure 7-10. shows the way in which physical addresses are generated based on the contents of CBAR, CBR and BBR. MMU comparators classify an access by logical area as defined by CBAR. Depending on which of the three potential logical areas (common area 1, bank area, or common area 0) is being accessed, the appropriate 8-bit base address is added to the high-order 4 bits of the logical address, yielding a 20-bit physical address. CBR is associated with common area 1 accesses. Common area 0 accesses use a (non-accessible, internal) base register which contains 0. Thus, common area 0, if defined, is always based at physical address 00000H.

7.6 MMU and Reset

During reset, all bits of the CA field of CBAR are set to 1 while all bits of the BA field of CBAR, CBR and BBR are reset to 0. The logical 64-kbyte address space corresponds directly with the first 64-kbyte (0000H to FFFFH) of the 512-kbyte (00000H to 7FFFFH) physical address space. Thus, after reset, the HD647180X will begin execution at logical and physical address 0.

7.7 MMU Register Access Timing

When data is written into CBAR, CBR or BBR, the value will be effective from the cycle immediately following the I/O write cycle which updates these registers.

Care must be taken during MMU programming to insure that CPU program execution is not disrupted. Observe that the next cycle following MMU register programming will normally be an opcode fetch from the newly translated address. One simple technique is to localize all MMU programming routines in a common area that is always enabled.

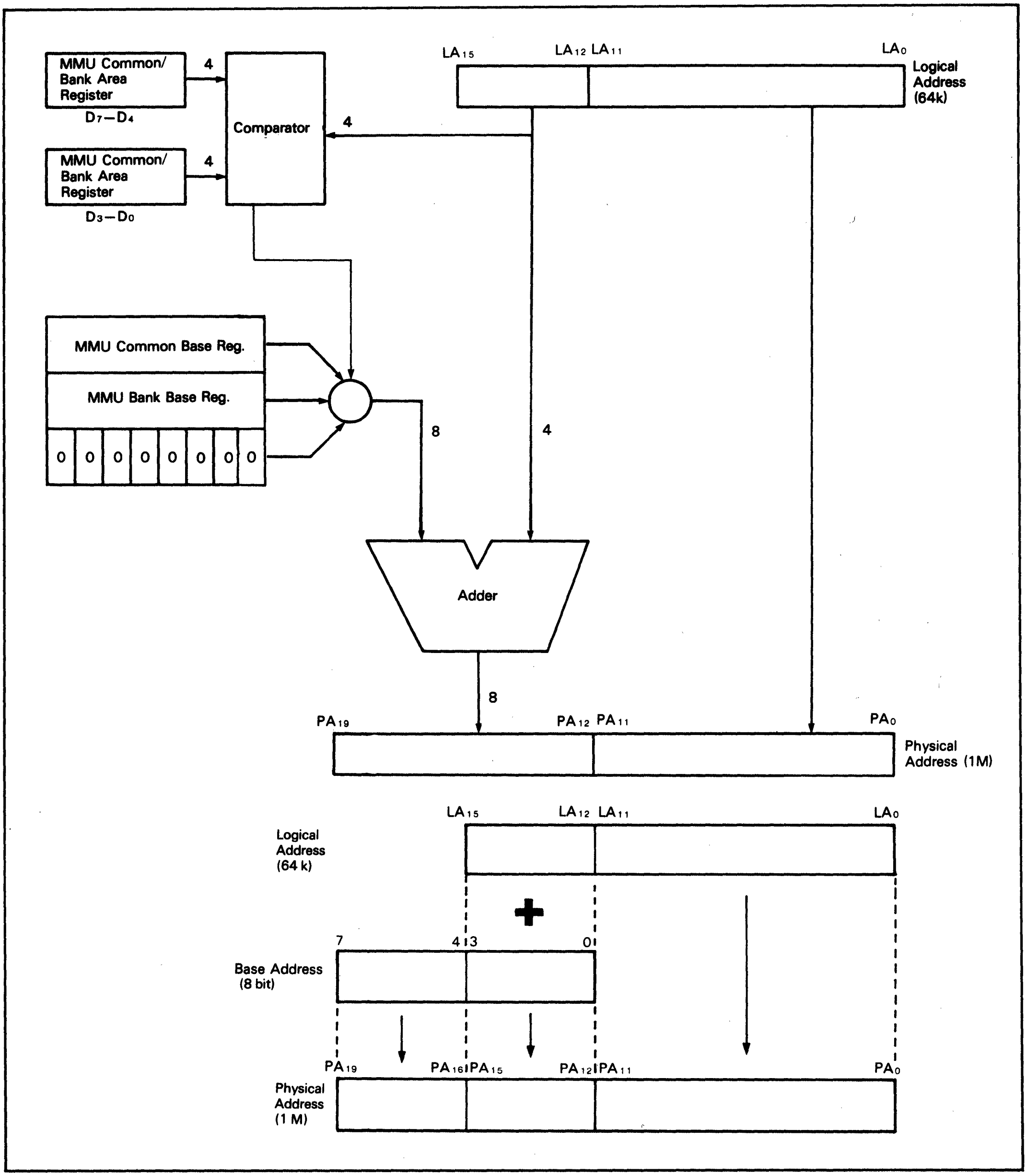


Figure 7-10. Physical Address Generation

SECTION 8. INTERRUPTS

The HD647180X CPU has fifteen interrupt sources, four external and eleven internal, with fixed priority.

Higher Priority	1.	Trap (Undefined Opcode Trap)	Internal Interrupt
	2.	$\overline{NMI}$ (Non Maskable Interrupt)	External Interrupt
	3.	$\overline{INT_0}$ (Maskable Interrupt Level 0)	External Interrupt
	4.	$\overline{INT_1}$ (Maskable Interrupt Level 1)	External Interrupt
	5.	$\overline{INT_2}$ (Maskable Interrupt Level 2)	External Interrupt
	6.	Input Capture	Internal Interrupt
	7.	Output Compare	Internal Interrupt
	8.	Timer Overflow	Internal Interrupt
	9.	Timer 0	Internal Interrupt
	10.	Timer 1	Internal Interrupt
	11.	DMA channel 0	Internal Interrupt
	12.	DMA channel 1	Internal Interrupt
	13.	Clocked Serial I/O Port	Internal Interrupt
Lower	14.	Asynchronous SCI channel 0	Internal Interrupt
Priority	15.	Asynchronous SCI channel 1	Internal Interrupt

Figure 8-1. Interrupt Sources

This section explains the CPU registers associated with interrupt processing, the TRAP interrupt, interrupt response modes, and the external interrupts. Detailed discussions of internal interrupt generation (except Trap) are presented in the appropriate hardware sections (that is PRT, DMAC, ASCI, and CSI/O).

8.1 Interrupt Control Registers and Flags

The HD647180X contains three registers and two flags which are associated with interrupt processing (figure 8-1.).

Table 8-1. Interrupt Registers

Function	Name	Access
Interrupt Vector High	I	LD A, I and LD I, A instructions
Interrupt Vector Low	IL	I/O instructions (addr = 33H)
Interrupt/Trap Control	ITC	I/O instruction (addr = 34H)
Interrupt Enable Flag 1,2	IEF1, IEF2	EI, DI, LD A, I, and LD A, I instructions

8.1.1 Interrupt Vector Register (I)

$\overline{INT_0}$ external interrupt mode 2, $\overline{INT_1}$ and $\overline{INT_2}$ external interrupts, and all internal interrupts (except Trap) use a programmable vectored technique to determine the address at which interrupt processing starts. In response to the interrupt a 16-bit address is generated. This address accesses a vector table in memory to obtain the address at which execution restarts.

While the method for generation of the least significant byte of the table address differs, all vectored interrupts use the contents of I as the most significant byte of the table address. By programming the contents of I, vector tables can be relocated on 256 bytes boundaries throughout the 64-kbyte logical address space.

Note that I is read/written with the LD A, I and LD I, A instructions rather than I/O (IN, OUT) instructions.

I is initialized to 00H during reset.

8.1.2 Interrupt Vector Low Register (IL)

Interrupt Vector Low Register (IL: I/O Address = 33H)

bit	7	6	5	4	3	2	1	0
	IL7	IL6	IL5	–	–	–	–	–
	R/W	R/W	R/W					
	Programmable			Interrupt Source Dependent				

Figure 8-2. Interrupt Vector Low Register

IL (figure 8-2.) determines the most significant three bits of the low-order byte of the interrupt vector table address for external interrupts $\overline{INT_1}$ and $\overline{INT_2}$ and all internal interrupts (except Trap). The five least significant bits are fixed for each specific interrupt source. By programming IL the vector table can be relocated on 32-byte boundaries.

IL is initialized to 00H during Reset.

8.1.3 INT/Trap Control Register (ITC)

INT/Trap Control Register (ITC: I/O Address = 34H)

bit	7	6	5	4	3	2	1	0
	TRAP	UFO	–	–	–	ITE2	ITE1	ITE0
	R/W	R				R/W	R/W	R/W

Figure 8-3. INT/Trap Control Register

ITC (figure 8-3.) is used to handle Trap interrupts and to enable or disable the external maskable interrupt inputs $\overline{INT_0}$, $\overline{INT_1}$ and $\overline{INT_2}$.

TRAP (Bit 7): This bit is set to 1 when an undefined opcode is fetched. TRAP can be reset under program control by writing 0 to it. However 1 cannot be written to it under program control. TRAP is reset to 0 during reset.

UFO: Undefined Fetch Object (Bit 6): When a trap interrupt occurs (TRAP bit is set to 1), the contents of UFO determine the starting address of the undefined instruction. This is necessary since the trap may occur on either the second or third byte of the opcode. UFO allows the stacked PC value (stacked in response to trap) to be correctly adjusted. If UFO = 0, the first opcode should be interpreted as the stacked PC − 1. If UFO = 1, the first opcode address is the stacked PC − 2. UFO is read-only.

ITE2,1,0: Interrupt Enable 2,1,0 (Bits 2-0): ITE2, ITE1, and ITE0 enable and disable the external interrupt inputs $\overline{INT_2}$, $\overline{INT_1}$ and $\overline{INT_0}$ respectively. If reset to 0, the interrupt is masked. During reset, ITE0 is initialized to 1 while ITE1 and ITE2 are initialized to 0.

8.1.4 Interrupt Enable Flag 1,2 (IEF_1, IEF_2)

IEF_1 controls the overall enabling and disabling of all internal and external maskable interrupts (that is, all interrupts except $\overline{NMI}$ and trap). If $IEF_1 = 0$, all maskable interrupts are disabled. IEF_1 can be reset to 0 by the DI (disable interrupts) instruction and set to 1 by the EI (enable interrupts) instruction.

The purpose of IEF_2 is to correctly manage the occurrence of $\overline{NMI}$. During $\overline{NMI}$, the prior interrupt reception state is saved and all maskable interrupts are automatically disabled (IEF_1 copied to IEF_2 and then IEF_1 cleared to 0). At the end of the $\overline{NMI}$ interrupt service routine, execution of the RETN (return from non-maskable interrupt) will automatically restore the interrupt state prior to the occurrence of $\overline{NMI}$ (by copying IEF_2 to IEF_1).

IEF_2 state can be reflected in the P/V bit of the CPU status register by executing LD A, I or LD A, R instructions.

Table 8-2. shows the state of IEF_1 and IEF_2.

8.1.5 Interrupt Requests during LD A, I or LD A, R Instruction

No interrupt requests including $\overline{NMI}$ can be sampled during execution of LD A, I or LD A, R instructions like EI and DI instruction.

Therefore, the correct value of IEF_2 is transfered to P/V flag after completion of LD A, I or LD A, R.

Table 8-2. State of IEF_1 and IEF_2

CPU Operation	IEF_1	IEF_2	Remarks
RESET	0	0	Inhibits interrupts except $\overline{NMI}$ and TRAP
$\overline{NMI}$	0	IEF_1	Copies the contents of IEF_1 to IEF_2
RETN	IEF_2	Not affected	Returns from the $\overline{NMI}$ service routine
Interrupt except $\overline{NMI}$ and TRAP	0	0	Inhibits interrupts except $\overline{NMI}$ and TRAP.
RETI	Not affected	Not affected	
TRAP	Not affected	Not affected	
EI	1	1	Interrupts are not sampled
DI	0	0	
LD A, I	Not affected	Not affected	Transfers the contents of IEF_2 to P/V flag Interrupts are not sampled
LD A, R	Not affected	Not affected	

8.2 Trap Interrupt

The HD647180X generates a non-maskable (not affected by the state of IEF_1) trap interrupt when an undefined opcode fetch occurs. This feature can be used to increase software reliability, implement an 'extended' instruction set, or both. Trap may occur during opcode fetch cycles and also if an undefined opcode is fetched during the interrupt acknowledge cycle for $\overline{INT_0}$ when mode 0 is used.

When a trap interrupt occurs the HD647180X operates as follows:

1. The TRAP bit in the Interrupt Trap/Control (ITC) register is set to 1.
2. The current PC (program counter) value, reflecting the location of the undefined opcode, is saved on the stack.
3. The HD647180X vectors to logical address 0. Note that if logical address 0000H is mapped to physical address 00000H, the vector is the same as for reset. In this case, testing the TRAP bit in ITC will reveal whether the restart at

physical address 00000H was caused by reset or trap.

The state of the UFO (Undefined Fetch Object) bit in ITC allows trap manipulation software to correctly 'adjust' the stacked PC depending on whether the second or third byte of the opcode generated the TRAP. If UFO $=$ 0, the starting address of the invalid instruction is equal to the stacked PC$-$1. If UFO $=$ 1, the starting address of the invalid instruction is equal to the stacked PC$-$2. Figure 8-4. and 8-5. show trap timing.

Note that bus release cycles, refresh cycles, DMA cycles, and WAIT cycles can't be inserted just after the T_{TP} state which is inserted for trap interrupt sequence.

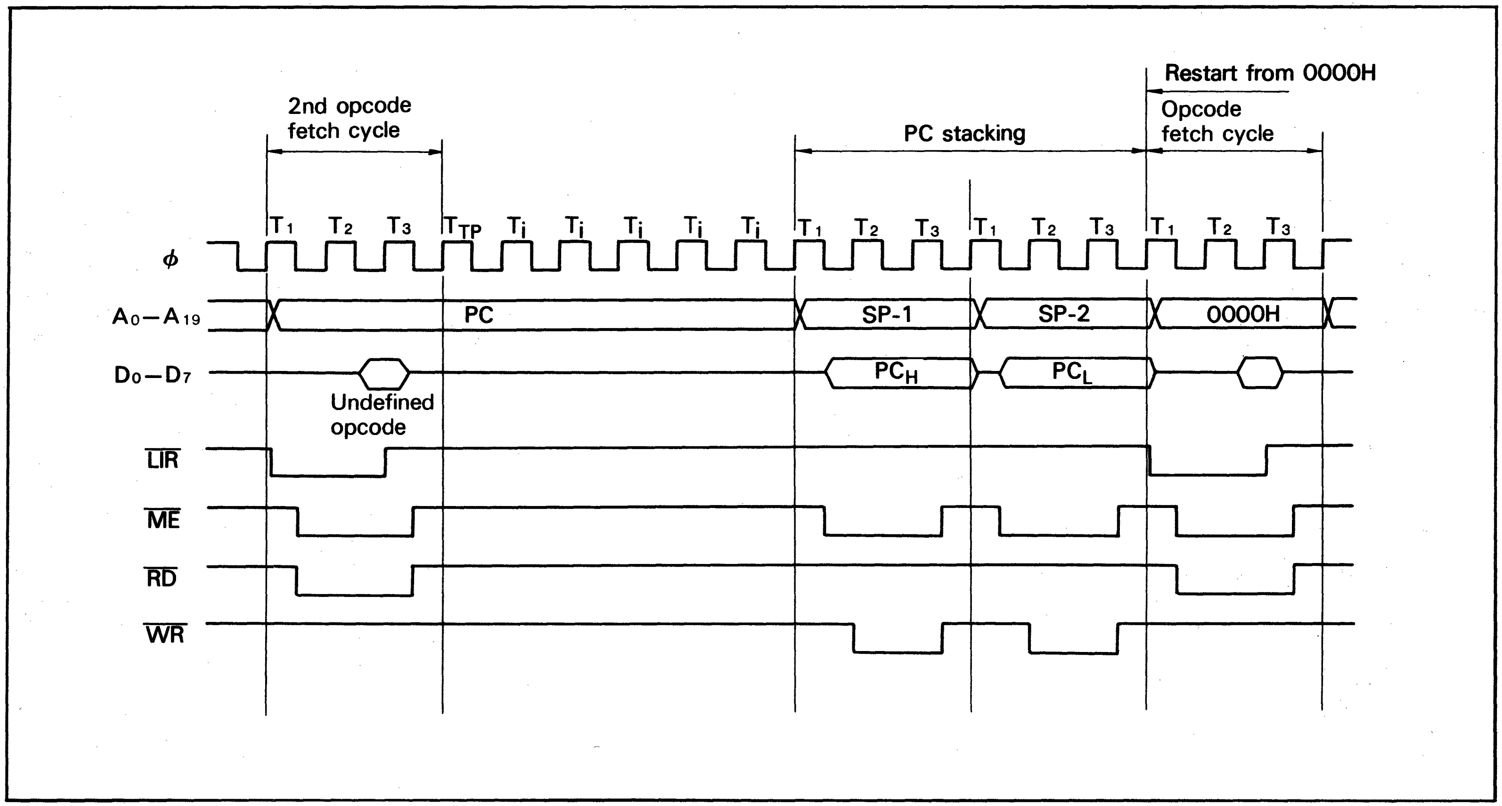


Figure 8-4. Trap Timing—Second Opcode Undefined

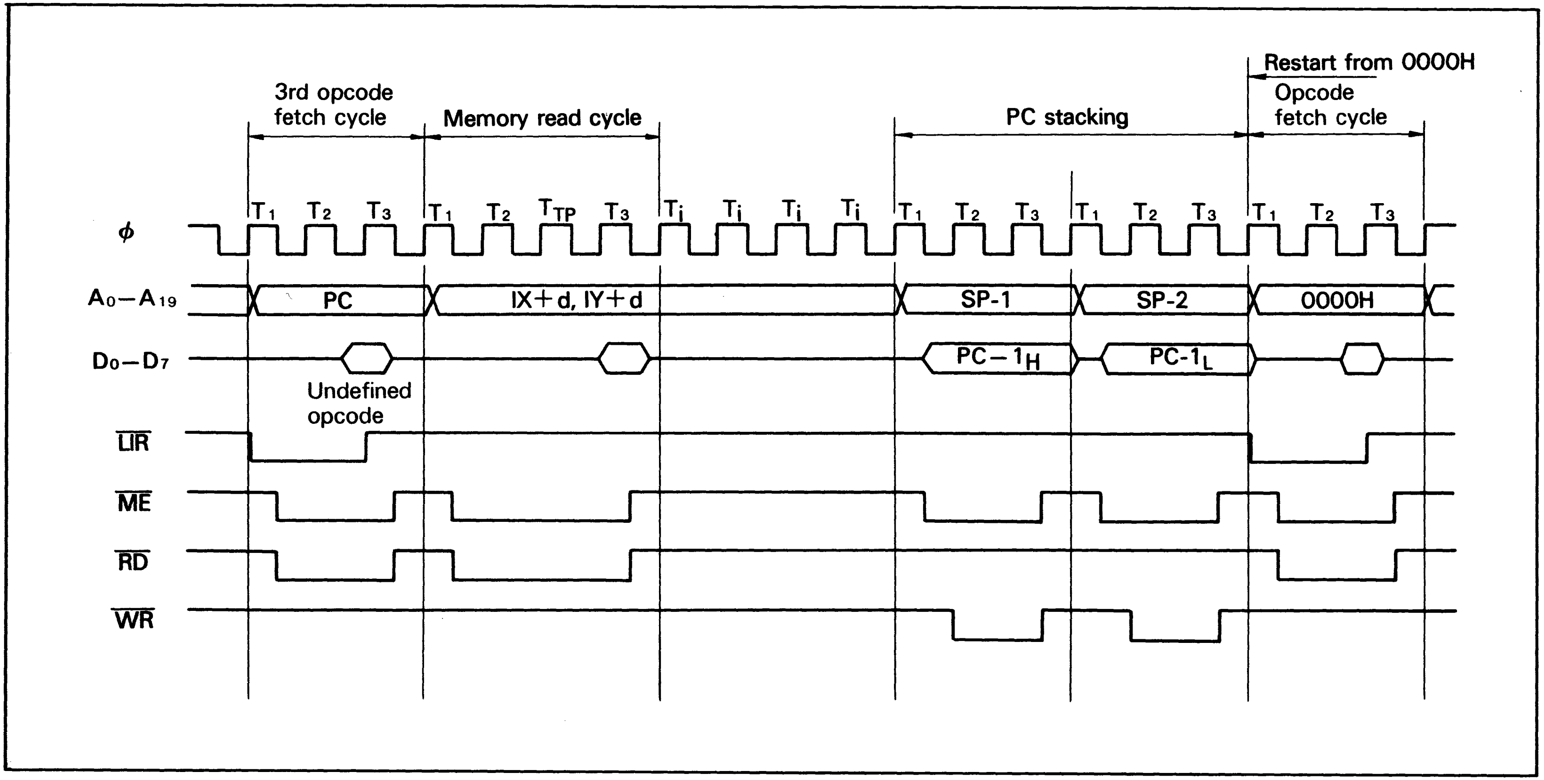


Figure 8-5. Trap Timing—Third Opcode Undefined

8.3 External Interrupts

The HD647180X has four external hardware interrupt inputs:

1. $\overline{\mathrm{NMI}}$ — Non-maskable interrupt
2. $\overline{\mathrm{INT_0}}$ — Maskable interrupt level 0
3. $\overline{\mathrm{INT_1}}$ — Maskable interrupt level 1
4. $\overline{\mathrm{INT_2}}$ — Maskable interrupt level 2

$\overline{\mathrm{NMI}}$, $\overline{\mathrm{INT_1}}$ and $\overline{\mathrm{INT_2}}$ have fixed interrupt response modes. $\overline{\mathrm{INT_0}}$ has three different software programmable interrupt response modes — mode 0, mode 1, and mode 2.

8.3.1 $\overline{\mathrm{NMI}}$ — Non-Maskable Interrupt

The $\overline{\mathrm{NMI}}$ interrupt input is edge sensitive and cannot be masked by software. When $\overline{\mathrm{NMI}}$ is detected, the HD647180X operates as follows:

1. DMAC operation is suspended by the clearing of the DME (DMA Main Enable) bit in DCNTL.
2. The PC is pushed onto the stack.
3. The contents of IEF_1 are copied to IEF_2. This saves the interrupt reception state that existed prior to $\overline{\mathrm{NMI}}$.
4. IEF_1 is cleared to 0. This disables all external and internal maskable interrupts (i.e. all interrupts except $\overline{\mathrm{NMI}}$ and trap).
5. Execution commences at logical address 0066H.

The last instruction of an $\overline{\mathrm{NMI}}$ service routine should be RETN (return from non-maskable interrupt). This restores the stacked PC, allowing the interrupted program to continue. Furthermore, RETN causes IEF_2 to be copied to IEF_1, restoring the interrupt reception state that existed prior to the $\overline{\mathrm{NMI}}$.

Note that $\overline{\mathrm{NMI}}$, since it can be accepted during HD647180X on-chip DMAC operation, can be used to externally interrupt DMA transfer. The $\overline{\mathrm{NMI}}$ service routine can reactivate or abort the DMAC operation as required by the application.

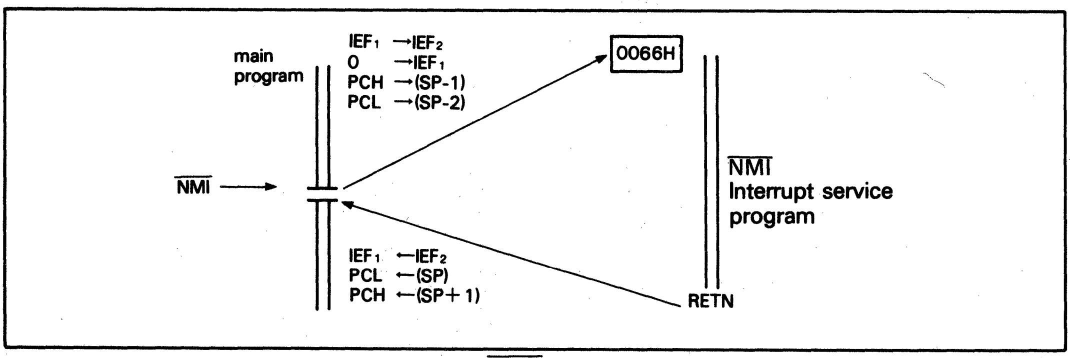


Figure 8-6. $\overline{\mathrm{NMI}}$ Sequence

For $\overline{\text{NMI}}$, special care must be taken to insure that interrupt inputs do not 'overrun' the $\overline{\text{NMI}}$ service routine. Unlimited $\overline{\text{NMI}}$ inputs without a corresponding number of RETN instructions will eventually cause stack overflow.

Figure 8-6. shows the use of $\overline{\text{NMI}}$ and RETN while figure 8-7. details $\overline{\text{NMI}}$ response timing. $\overline{\text{NMI}}$ is edge sensitive and the internally latched $\overline{\text{NMI}}$ falling edge is held until it is sampled. If the falling edge of $\overline{\text{NMI}}$ is latched before the falling edge of the clock state prior to T_3 or Ti in the last machine cycle, the internally latched $\overline{\text{NMI}}$ is sampled at the falling edge of the clock state prior to T_3 or Ti in the last machine cycle and the $\overline{\text{NMI}}$ acknowledge cycle begins at the end of the current machine cycle.

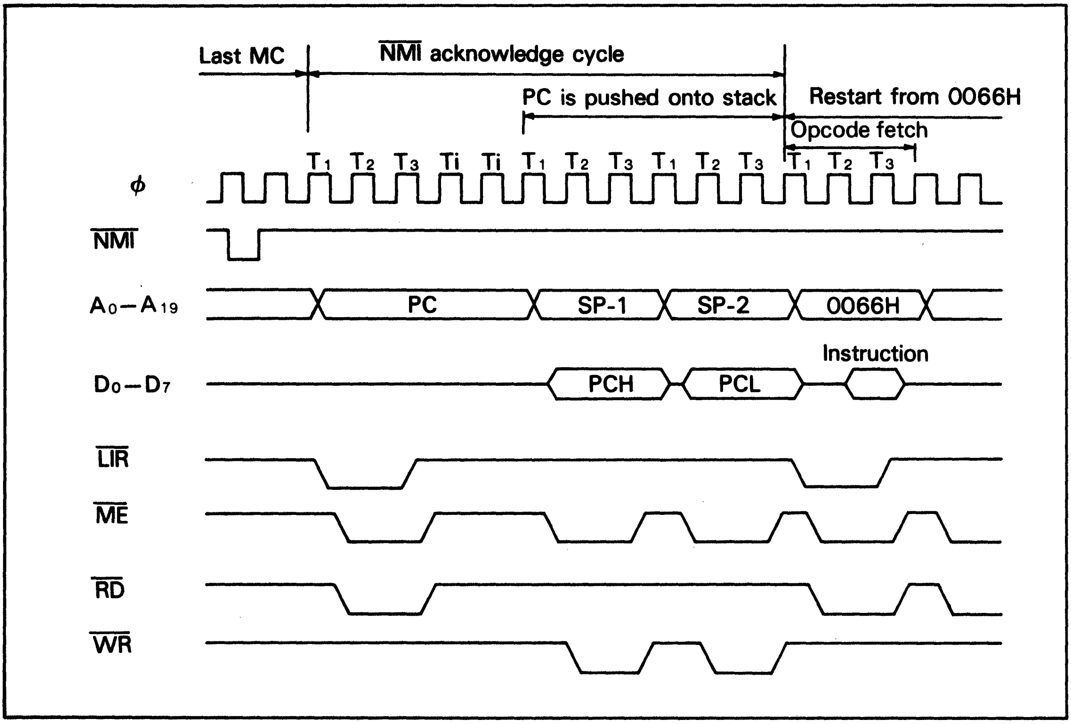


Figure 8-7. $\overline{\text{NMI}}$ Timing

8.3.2 $\overline{INT_0}$ — Maskable Interrupt Level 0

The next highest priority external interrupt after $\overline{NMI}$ is $\overline{INT_0}$. $\overline{INT_0}$ is sampled at the falling edge of the clock state prior to T_3 or Ti in the last machine cycle. If $\overline{INT_0}$ is asserted low at the falling edge of the clock state prior to T_3 or Ti in the last machine cycle, $\overline{INT_0}$ is accepted. The interrupt is masked if either the IEF_1 flag or the ITE0 (interrupt enable 0) bit in ITC are reset to 0. Note that after reset the state is as follows.

- IEF_1 is 0, so $\overline{INT_0}$ is masked
- ITE0 is 1, so $\overline{INT_0}$ is enabled by execution of the EI (enable interrupts) instruction

The $\overline{INT_0}$ interrupt is unique in that three programmable interrupt response modes are available: mode 0, mode 1, and mode 2. The specific mode is selected with the IM 0, IM 1, and IM 2 (set interrupt mode) instructions. However, in single-chip mode, mode 0 or mode 2 operation cannot be guaranteed. In this mode, the IM 1 instruction should be executed after reset. During reset, the HD647180X is initialized to use mode 0 for $\overline{INT_0}$.

The three interrupt response modes for $\overline{INT_0}$ are:

1. Mode 0—Instruction fetch from data bus
2. Mode 1—Restart at logical address 0038H
3. Mode 2—Low-byte vector table address fetch from data bus

$\overline{INT_0}$ Mode 0: During the interrupt acknowledge cycle, an instruction is fetched from the data bus (D_0-D_7) at the rising edge of T_3. Often, this instruction is one of the eight single byte RST (restart) instructions which stack the PC and restart execution at a fixed logical address. However, multibyte instructions can be processed if the interrupt acknowledging device can provide a multibyte response. Unlike all other interrupts, the PC is not automatically stacked.

Note that a trap interrupt will occur if an invalid instruction is fetched during mode 0 interrupt acknowledge.

Figure 8-8. shows $\overline{INT_0}$ Mode 0 Timing.

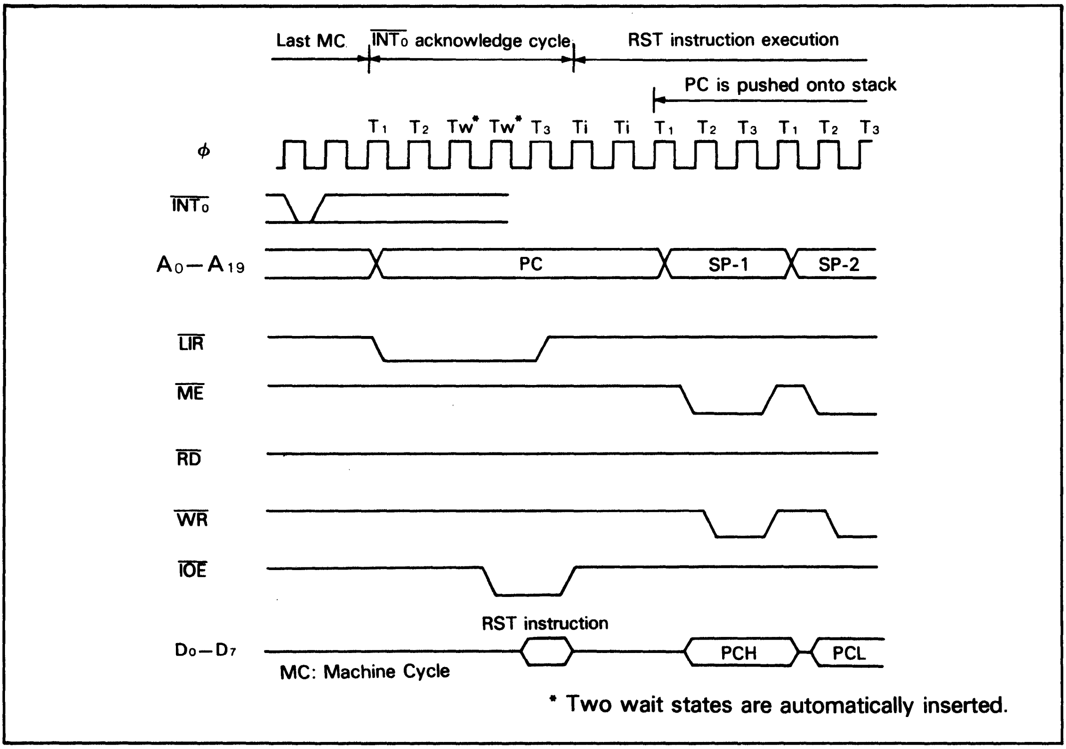


Figure 8-8. $\overline{INT_0}$ Mode 0 Timing (RST Instruction on the Data Bus)

$\overline{INT_0}$ Mode 1: When $\overline{INT_0}$ is received, the PC is stacked and instruction execution restarts at logical address 0038H. Both IEF_1 and IEF_2 flags are reset to 0, disabling all maskable interrupts. The interrupt service routine should normally terminate with the EI (enable interrupts) instruction followed by the RETI (return from interupt) instruction, so that the interrupts are reenabled. Figure 8-9. shows the use of $\overline{INT_0}$ (mode 1) and RETI.

Figure 8-10. shows $\overline{INT_0}$ mode 1 timing.

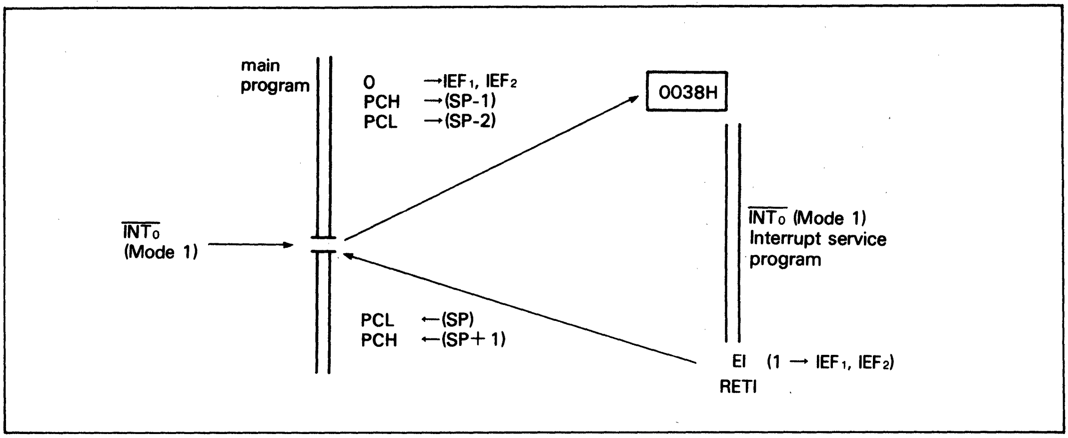


Figure 8-9. $\overline{\mathrm{INT}_0}$ Mode 1 Interrupt Sequence

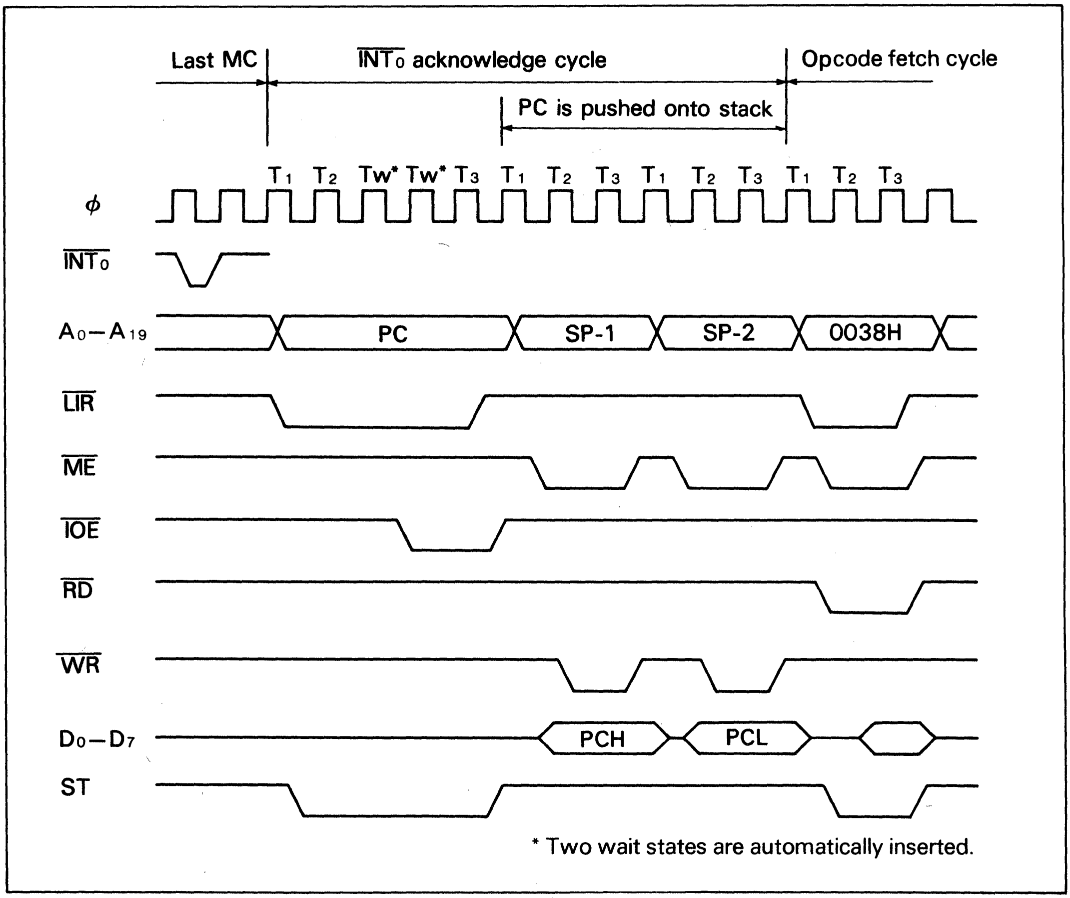


Figure 8-10. $\overline{\mathrm{INT}_0}$ Mode 1 Timing

$\overline{INT_0}$ Mode 2: Mode 2 determines the restart address by reading the contents of a table residing in memory. The vector table consists of up to 128 two-byte restart addresses stored in low byte, high byte order.

The vector table address is located on 256 bytes boundaries in the 64-kbyte logical address space as programmed in the 8-bit interrupt vector register (I). Figure 8-11. shows the $\overline{INT_0}$ mode 2 vector acquisition.

During $\overline{INT_0}$ mode 2 acknowledge cycle, first, the low-order 8 bits of vector is fetched from the data bus at the rising edge of T_3 and CPU acquires the 16-bit vector. Next, the PC is stacked. Finally, the 16-bit restart address is fetched from the vector table and execution commences at that address.

Note that external vector acquisition is indicated by $\overline{LIR}$ and $\overline{IOE}$ both low. Two wait states (Tw) are automatically inserted for external vector fetch cycles.

During reset the interrupt vector register (I) is initialized to 00H and, if necessary, should be set to a different value prior to the occurrence of a mode 2 $\overline{INT_0}$ interrupt. Figure 8-12. shows $\overline{INT_0}$ interrupt mode 2 timing.

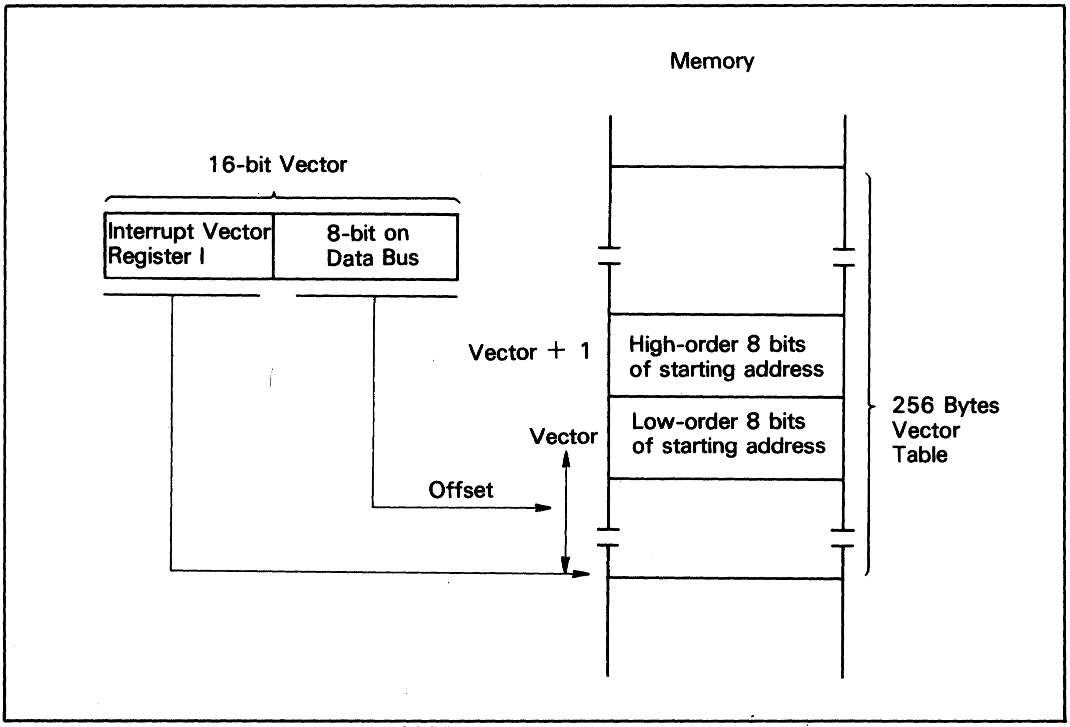


Figure 8-11. $\overline{INT_0}$ Mode 2 Vector Acquisition

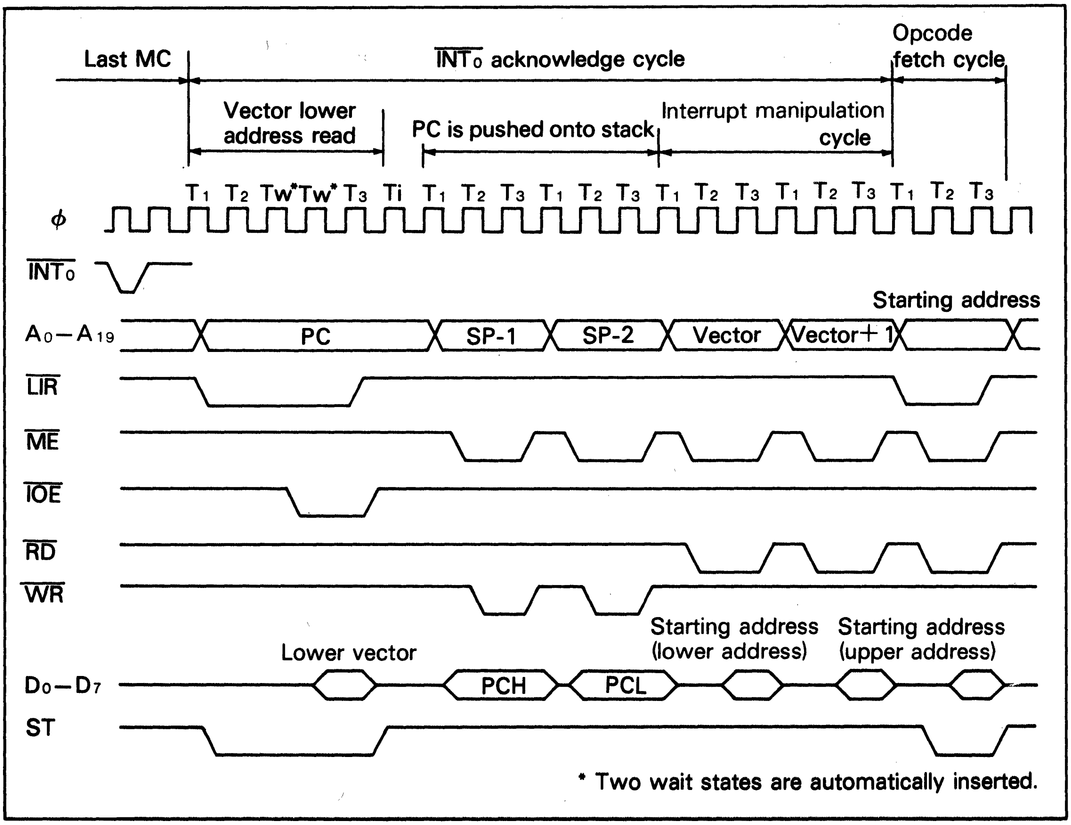


Figure 8-12. $\overline{INT_0}$ Mode 2 Timing

8.3.3 $\overline{INT_1}$, $\overline{INT_2}$

External interrupts $\overline{INT_1}$ and $\overline{INT_2}$ operate in a vector mode similar to $\overline{INT_0}$ mode 2. The difference is that $\overline{INT_1}$ and $\overline{INT_2}$ generate the low-order byte of vector table address using the IL (interrupt vector low) register rather than fetching it from the data bus. This is also the interrupt response sequence used for all internal interrupts (except trap).

As shown in figure 8-13. the low-order byte of vector table address is composed of the most significant three bits of the software programmable IL register while the least significant five bits are a unique fixed value for each interrupt ($\overline{INT_1}$, $\overline{INT_2}$, and internal) source.

$\overline{INT_1}$ and $\overline{INT_2}$ are globally masked by $IEF_1 = 0$. Each is also individually maskable by respectively clearing the ITE1 and ITE2 (bits 1, 2) of the INT/Trap control register to 0.

During RESET, IEF_1, ITE1 and ITE2 bits are reset to 0.

8.3.4 Internal Interrupts

Internal interrupts (except trap) use the same vectored response mode as $\overline{INT_1}$ and $\overline{INT_2}$ (figure 8-13.). Internal interrupts are globally masked by $IEF_1 = 0$. Individual internal interrupts are enabled/disabled by programming each individual I/O (PRT, DMAC, CSI/O, ASCI) control register. The lower vector of $\overline{INT_1}$, $\overline{INT_2}$ and internal interrupt are summarized in table 8-3.

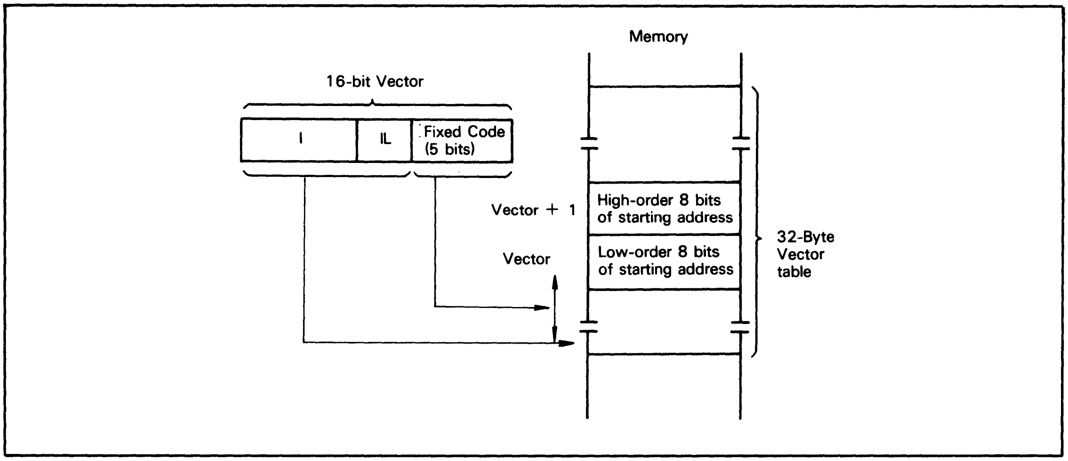


Figure 8-13. $\overline{INT_1}$, $\overline{INT_2}$, and Internal Interrupt Vector Acquisition

Table 8-3. Interrupt Source and Lower Vector

		IL			Fixed Code				
Interrupt Source	**Priority**	**b_7**	**b_6**	**b_5**	**b_4**	**b_3**	**b_2**	**b_1**	**b_0**
$\overline{INT_1}$	Highest	*	*	*	0	0	0	0	0
$\overline{INT_2}$	↑	*	*	*	0	0	0	1	0
Input capture		*	*	*	1	0	0	1	0
Output compare		*	*	*	1	0	1	0	0
Timer overflow		*	*	*	1	0	1	1	0
PRT channel 0		*	*	*	0	0	1	0	0
PRT channel 1		*	*	*	0	0	1	1	0
DMA channel 0		*	*	*	0	1	0	0	0
DMA channel 1		*	*	*	0	1	0	1	0
CSI/O		*	*	*	0	1	1	0	0
ASCI channel 0	↓	*	*	*	0	1	1	1	0
ASCI channel 1	Lowest	*	*	*	1	0	0	0	0

Note: * = Programmable

8.4 Interrupt Acknowledge Cycle Timing

Figure 8-14. shows interrupt acknowledge cycle timing for internal interrupts, $\overline{INT_1}$ and $\overline{INT_2}$. $\overline{INT_1}$ and $\overline{INT_2}$ are sampled at the falling edge of clock state prior to T_3 or Ti in the last machine cycle. If $\overline{INT_1}$ or $\overline{INT_2}$ is asserted LOW at the falling edge of clock state prior to T_3 or Ti in the last machine cycle, the interrupt request is accepted.

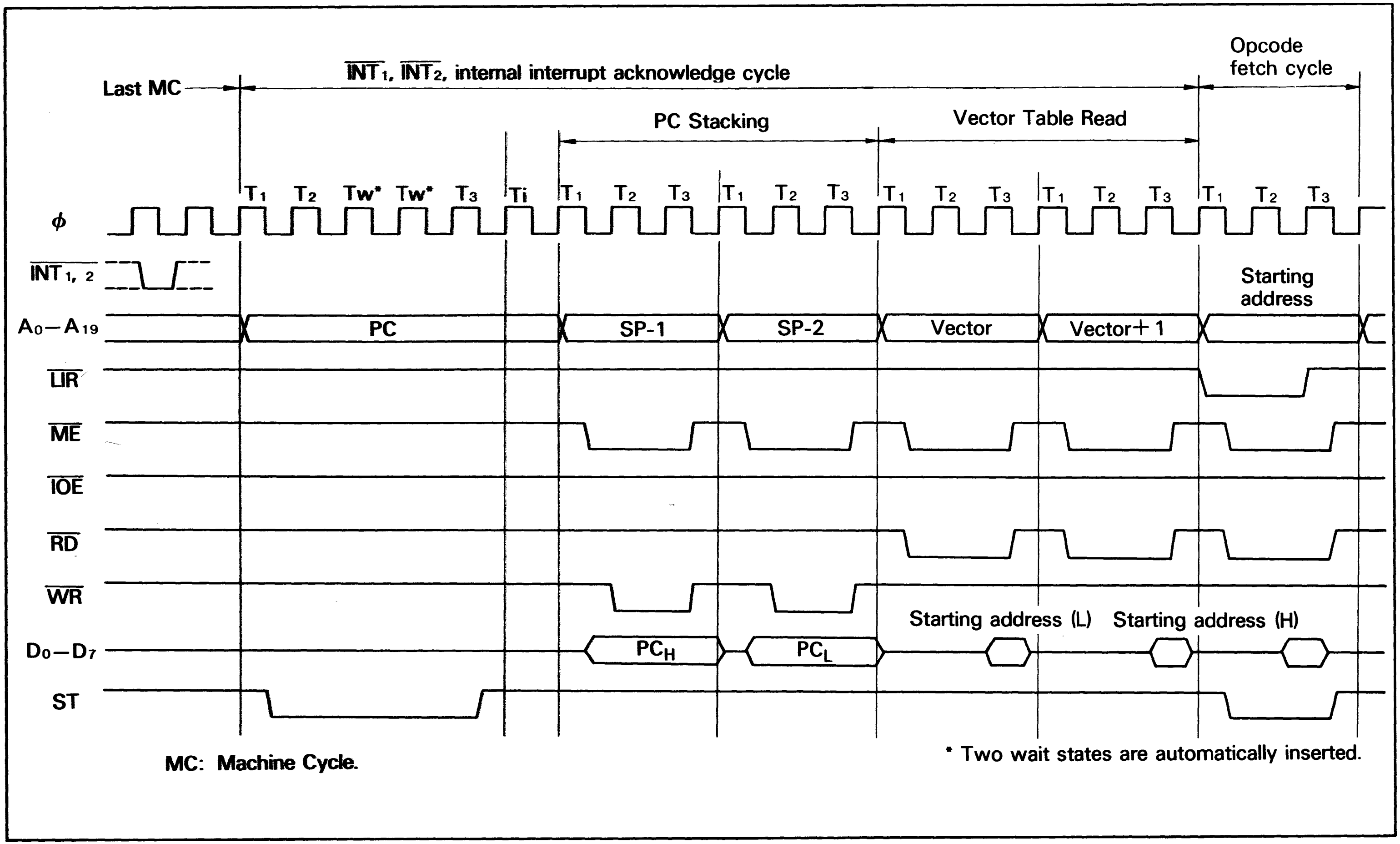


Figure 8-14. $\overline{INT_1}$, $\overline{INT_2}$, and Internal Interrupt Timing

8.5 Interrupt Sources and Reset

8.5.1 Interrupt Vector Register (I)

All bits of I are reset to 0.

Since I = 0 locates the vector tables starting at logical address 0000H, vectored interrupts ($\overline{INT_0}$ Mode 2, $\overline{INT_1}$, $\overline{INT_2}$ and internal interrupts) will overlap with fixed restart interrupts like RESET (0), $\overline{NMI}$ (0066H), $\overline{INT_0}$ Mode 1 (0038H) and RST (0000H - 0038H). The vector table(s) can be built elsewhere in memory and located on 256 bytes boundaries by reprogramming I with the LD I, A instruction.

8.5.2 IL Register

Bits 7–5 of IL are of IL reset to 0.

The IL register can be programmed to locate the vector table for $\overline{INT_1}$, $\overline{INT_2}$ and internal interrupts on 32 bytes sub-boundaries within the 256 bytes area specified by I.

8.5.3 IEF_1, IEF_2 Flags

IEF_1 and IEF_2 are reset to 0.

Interrupts other than $\overline{NMI}$ and trap are disabled.

8.5.4 ITC Register

ITE0 is set to 1. ITE1, ITE2 are reset to 0.

$\overline{INT_0}$ can be enabled by the EI instruction, which sets IEF_1 = 1. To enable $\overline{INT_1}$ and $\overline{INT_2}$ the ITE1 and ITE2 bits must also be set = 1 by writing to ITC.

8.5.5 I/O Control Registers

I/O control register interrupt enable bits are reset to 0.

All HD647180X on-chip I/O (PRT, DMAC, CSI/O, ASCI) interrupts are disabled and can be individually enabled by writing to each I/O control register's interrupt enable bit.

8.6 Difference Between $\overline{INT_0}$ Interrupt and the Other Interrupts

As shown in figures 8-8, 8-10, 8-12, and 8-14, the interrupt acknowledge cycle of $\overline{INT_0}$ is different from those of the other interrupts, that is, $\overline{INT_1}$, $\overline{INT_2}$ and internal interrupts concerning the state of control signals. The state of the control signals in each interrupt acknowledge cycle are:

- $\overline{INT_0}$ interrupt acknowledge cycle: $\overline{LIR} = 0$, $\overline{IOE} = 0$, $ST = 0$
- $\overline{INT_1}$, $\overline{INT_2}$, and internal interrupt acknowledge cycle: $\overline{LIR} = 1$, $\overline{IOE} = 1$, $ST = 0$

8.7 Notes on $\overline{INT_0}$ Mode 0

8.7.1 Problem

In $\overline{INT_0}$ mode 0, CPU executes an instruction which is placed on the data bus during the interrupt acknowledge cycle. Usually, an RST (1-byte instruction) or CALL (3-byte instruction) is placed on the data bus. Then, the CPU pushes the program counter (PC) onto the stack and jumps to the interrupt service routine. For a RST instruction, the correct return address is pused onto the stack. However, in the case of CALL instruction, the pushed return address is equal to the correct return address + 2.

8.7.2 Explanation of Operation

During the first opcode fetch cycle in the interrupt acknowledge cycle, the CPU stops incrementing the PC. At this time, the PC contains the return address. After the first opcode is fetched, the CPU restarts incrementing the PC. Therefore, if an RST (1-byte instruction) is executed in the interrupt acknowledge cycle, the correct return address is pushed onto the stack and the CPU can return from the interrupt service routine correctly. While, if a CALL (3-byte instruction) is executed in the interrupt acknowledge cycle, the PC is incremented twice during the operand read cycle of the 2 bytes after the first opcode is fetched. Therefore, the return address + 2 in the PC is pushed onto the stack. So, when RETI is executed at the end of the interrupt service routine, the CPU can not return from the interrupt correctly.

Figure 8-15. shows CALL execution timing in $\overline{INT_0}$ mode 0.

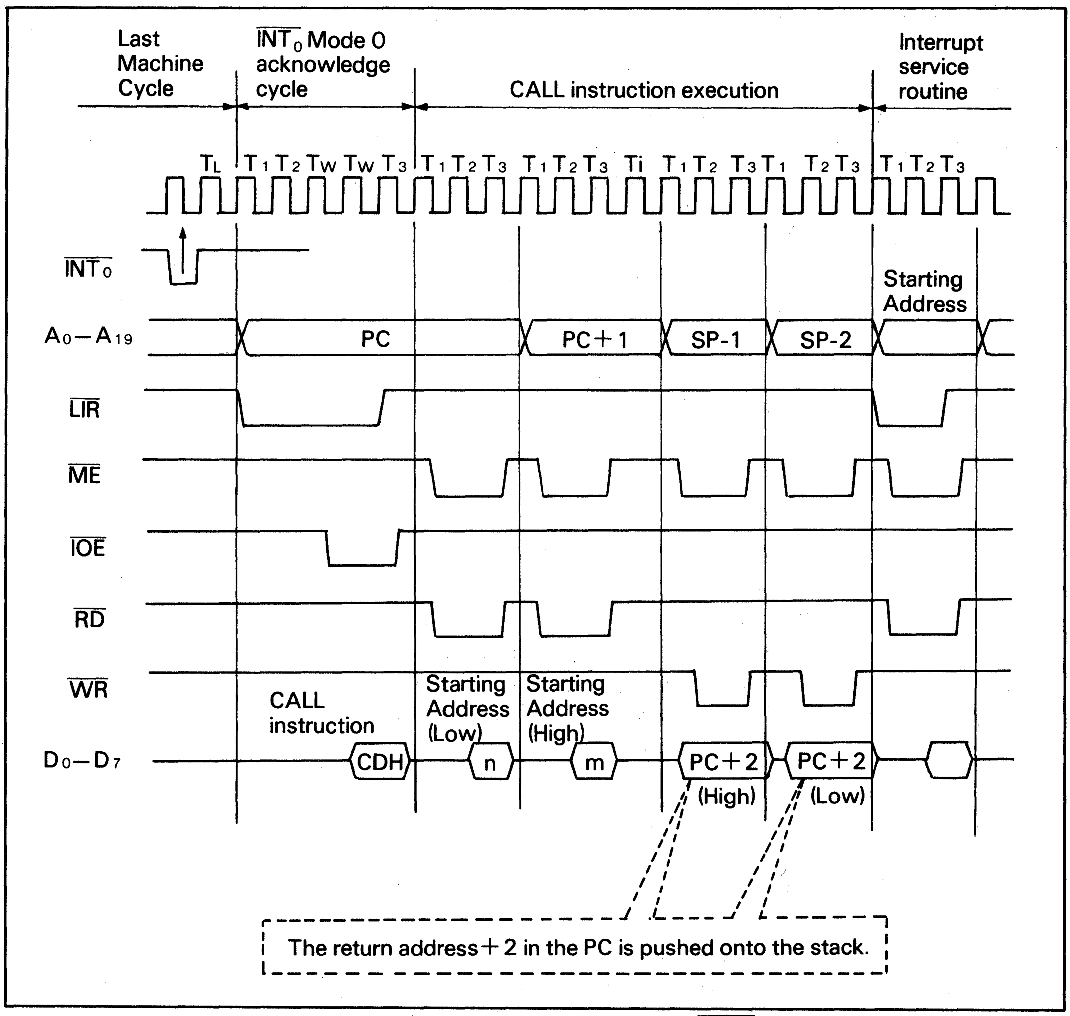


Figure 8-15. CALL Execution Timing in $\overline{INT_0}$ mode 0

8.7.3 Countermeasure

The following explains the countermeasures of the problem in $\overline{INT_0}$ mode 0.

RST: When RST is executed, the correct return address in the PC is pushed onto the stack.

CALL: When CALL is executed, the stack contents must be decremented by two in the interrupt service routine to return from the interrupt correctly.
Table 8-4. summarizes how to adjust the stack contents depending on the instruction to be executed.

Table 8-4. Stack Contents Adjustment

Instruction	Stack Contents Adjustment
RST	No
CALL	Decrement the stack contents by two
Other instructions	No (The PC is not stacked.)

The $\overline{\mathrm{INT}_0}$ mode 0 sequences when executing RST and CALL are shown in figure 8-16.

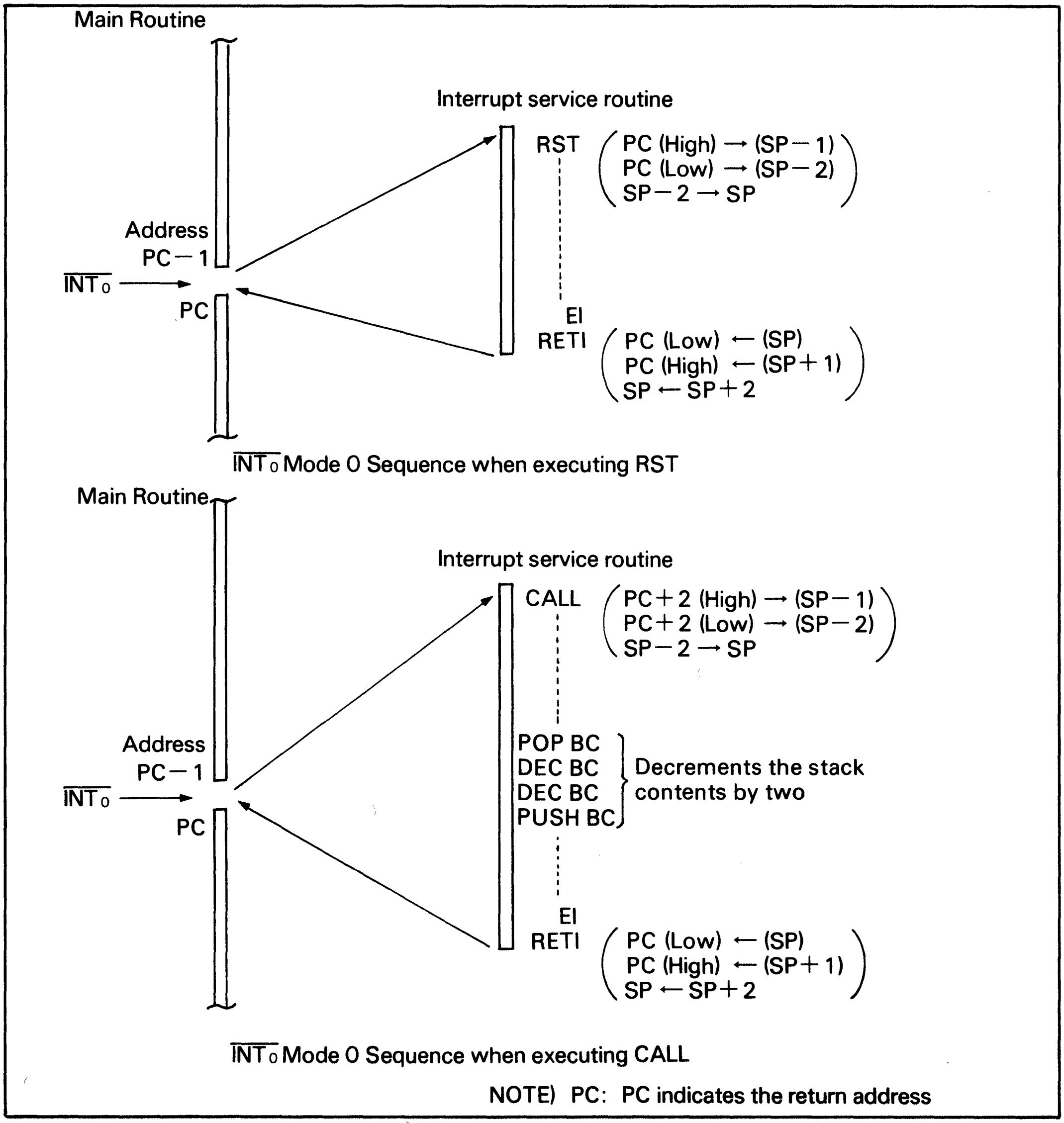


Figure 8-16. $\overline{\mathrm{INT}_0}$, Mode 0 Sequence

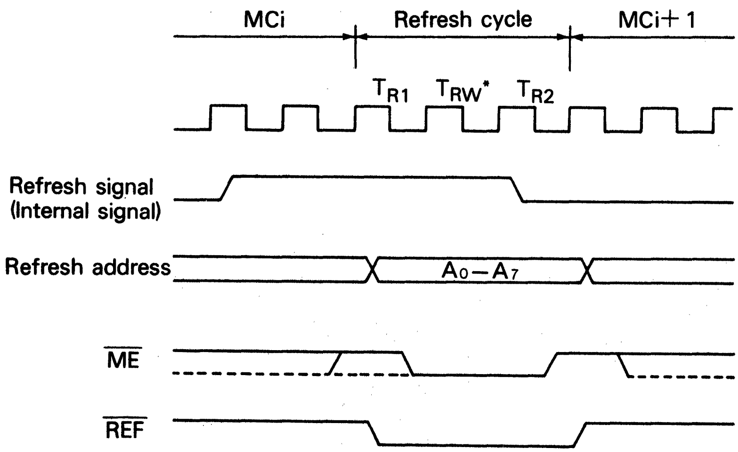
SECTION 9. DYNAMIC RAM REFRESH CONTROL

The HD647180X incorporates a dynamic RAM refresh control circuit including 8-bit refresh address generation and programmable refresh timing. This circuit generates asynchronous refresh cycles inserted at the programmed interval independent of CPU program execution. For systems which don't use dynamic RAM, the refresh function can be disabled.

When the internal refresh controller determines that a refresh cycle should occur, the current instruction is interrupted at the first breakpoint between machine cycles. The refresh cycle is inserted by placing the refresh address on A_0-A_7 and the $\overline{REF}$ output is driven low.

Refresh cycles may be programmed to be either two or three clock cycles in duration by programming the REFW (refresh wait) bit in refresh control register (RCR). Note that the external $\overline{WAIT}$ input and the internal wait state generator are not effective during refresh.

Figure 9-1 shows the timing of a refresh cycle with a refresh wait (T_{RW}) cycle.



Note: * If three refresh cycles are specified, T_{RW} is inserted. Otherwise, T_{RW} is not inserted.
MC: Machine Cycle

Figure 9-1. Refresh Timing

9.1 Refresh Control Register (RCR)

RCR (figure 9-2) specifies the interval and length of refresh cycles, as well as enabling or disabling the refresh function.

Refresh Control Register (RCR: I/O Address = 36H)

bit	7	6	5	4	3	2	1	0
	REFE	REFW	—	—	—	—	CYC1	CYC0
	R/W	R/W					R/W	R/W

Figure 9-2. Refresh Control Register

9.1.1 REFE: Refresh Enable (Bit 7)

REFE = 0 disables the refresh controller while REFE = 1 enables refresh cycle insertion. REFE is set to 1 during RESET.

9.1.2 REFW: Refresh Wait (Bit 6)

REFW = 0 causes the refresh cycle to be two clocks in duration. REFW = 1 causes the refresh cycle to be three clocks in duration by adding a refresh wait cycle (T_{RW}). REFW is set to 1 during RESET.

9.1.3 CYC1, CYC0: Cycle Interval (Bits 1, 0)

CYC1 and CYC0 specify the interval (in clock cycles) between refresh cycles.

In the case of dynamic RAMs requiring 128 refresh cycles every 2 ms (or 256 cycles every 4 ms), the required refresh interval is less than or equal to 15.625 μs. Thus, the underlined values in table 9-1 indicate the best refresh interval depending on CPU clock frequency. CYC0 and CYC1 are cleared to 0 during RESET.

Table 9-1. Refresh Interval

		Insertion	Time interval			
CYC1	CYC0	interval	ϕ: 8 MHz	6 MHz	4 MHz	2.5 MHz
0	0	10 states	1.25 μs	1.66 μs	2.5 μs	4.0 μs
0	1	20 states	2.5 μs	3.3 μs	5.0 μs	$\underline{8.0\ \mu s}$
1	0	40 states	5.0 μs	6.6 μs	$\underline{10.0\ \mu s}$	16.0 μs
1	1	80 states	$\underline{10.0\ \mu s}$	$\underline{13.3\ \mu s}$	20.0 μs	32.0 μs

9.2 Refresh Control and Reset

After reset, based on the initialized value of RCR, refresh cycles will occur with an interval of 10 clock cycles and be 3 clock cycles in duration.

9.3 Dynamic RAM Refresh Operation Notes

Refresh cycle insertion is stopped when the CPU is in the following states:

- During reset
- When the bus is released in response to $\overline{\text{BUSREQ}}$
- During sleep mode
- During wait states

Refresh cycles are suppressed when the bus is released in response to $\overline{\text{BUSREQ}}$. However, the refresh timer continues to operate. Thus, the time at which the first refresh cycle occurs after the HD647180X re-acquires the bus depends on the refresh timer, and has no timing relationship with the bus exchange.

Refresh cycles are suppressed during sleep mode. If a refresh cycle is requested during sleep mode, the refresh cycle request is internally 'latched' (until replaced with the next refresh request). The 'latched' refresh cycle is inserted at the end of the first machine cycle after sleep mode is exited. After this initial cycle, the time at which the next refresh cycle will occur depends on the refresh time, and has no timing relationship with the exit from sleep mode.

The refresh address is incremented by 1 for each successful refresh cycle, not for each refresh request. Thus, independent of the number of 'missed' (suppressed) refresh requests, each refresh bus cycle will use a refresh address incremented by 1 from that of the previous refresh bus cycles.

When internal refresh requests are asserted during bus release mode (figure 9-3), one request of them is retained and one refresh cycle is executed, following one machine cycle of the CPU after completion of bus release mode as shown in figure 9-3.

In mode 0 (single-chip mode), the refresh controller can insert refresh cycles at a constant interval. However, external devices cannot tell whether or not the current bus cycle is a refresh cycle since the $\overline{\text{REF}}$ pin functions as the PB_6 pin. Therefore, the REFE bit should be cleared to 0 in mode 0.

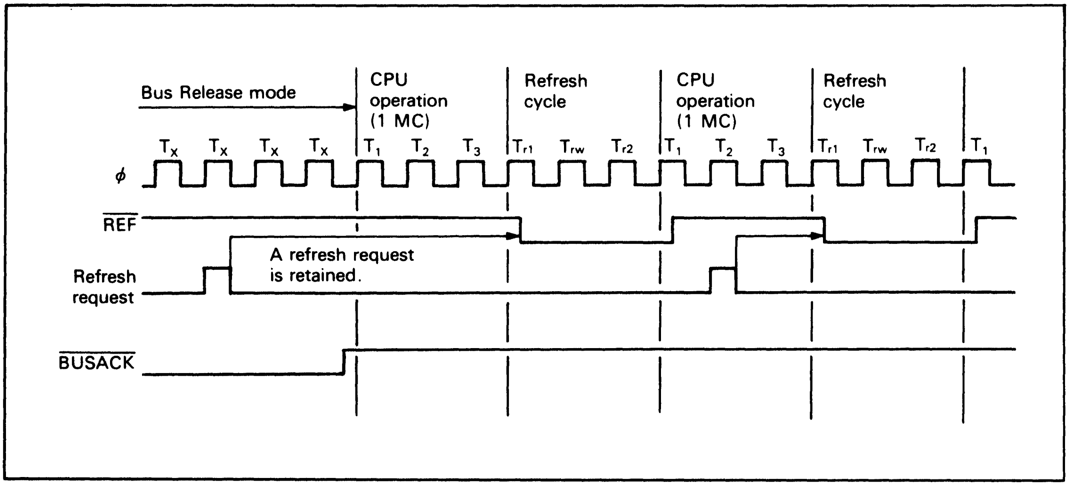


Figure 9-3. Refresh Requests during Bus Release Mode

SECTION 10. DMA CONTROLLER (DMAC)

The HD647180X contains a two-channel DMA (direct memory access) controller which supports high-speed data transfer. Both channels (channel 0 and channel 1) have the following capabilities:

Memory Address Space: Memory source and destination addresses can be directly specified anywhere within the 1-Mbyte physical address space using 20-bit source and destination memory addresses. In addition, memory transfers can arbitrarily cross 64-kbyte physical address boundaries without CPU intervention.

I/O Address Space: I/O source and destination addresses can be directly specified anywhere within the 64-kbyte I/O address space (16-bit source and destination I/O addresses).

Transfer Length: Up to 64-kbyte can be transferred based on a 16-bit byte count register.

$\overline{\text{DREQ}}$ Input: Level and edge sense $\overline{\text{DREQ}}$ input detection can be selected.

$\overline{\text{TEND}}$ Output: $\overline{\text{TEND}}$ indicates DMA completion to external devices.

Transfer Rate: A byte transfer can occur every six clock cycles. Wait states can be inserted in DMA cycles for slow memory or I/O devices. At the system clock (ϕ) = 6 MHz, the DMA transfer rate is as high as 1.0 megabytes/second (no wait states).

An additional feature is DMA interrupt request by DMA END.

Each channel has the following additional specific capabilities.

Channel 0: Channel 0 has the following features:

- Memory to/from memory, memory to/from I/O, memory to/from memory mapped I/O transfers
- Memory address increment, decrement, no change
- Burst or cycle steal memory to/from memory transfers
- DMA to and from both ASCI channels
- Higher priority than DMAC channel 1

Channel 1: Channel 1 has the following features:

- Memory to/from I/O transfer
- Memory address increment, decrement

DMAC Registers: Each channel of the DMAC (channel 0, 1) has three registers specifically associated with that channel.

· Channel 0

SAR0 – Source address register
DAR0 – Destination address register
BCR0 – Byte count register

· Channel 1

MAR1 – Memory address register
IAR1 – I/O address register
BCR1 – Byte count register

The two channels share the following three additional registers.
DSTAT – DMA status register
DMODE – DMA mode register
DCNTL – DMA control register

10.1 DMAC Block Diagram

Figure 10-1 shows the HD647180X DMAC block diagram.

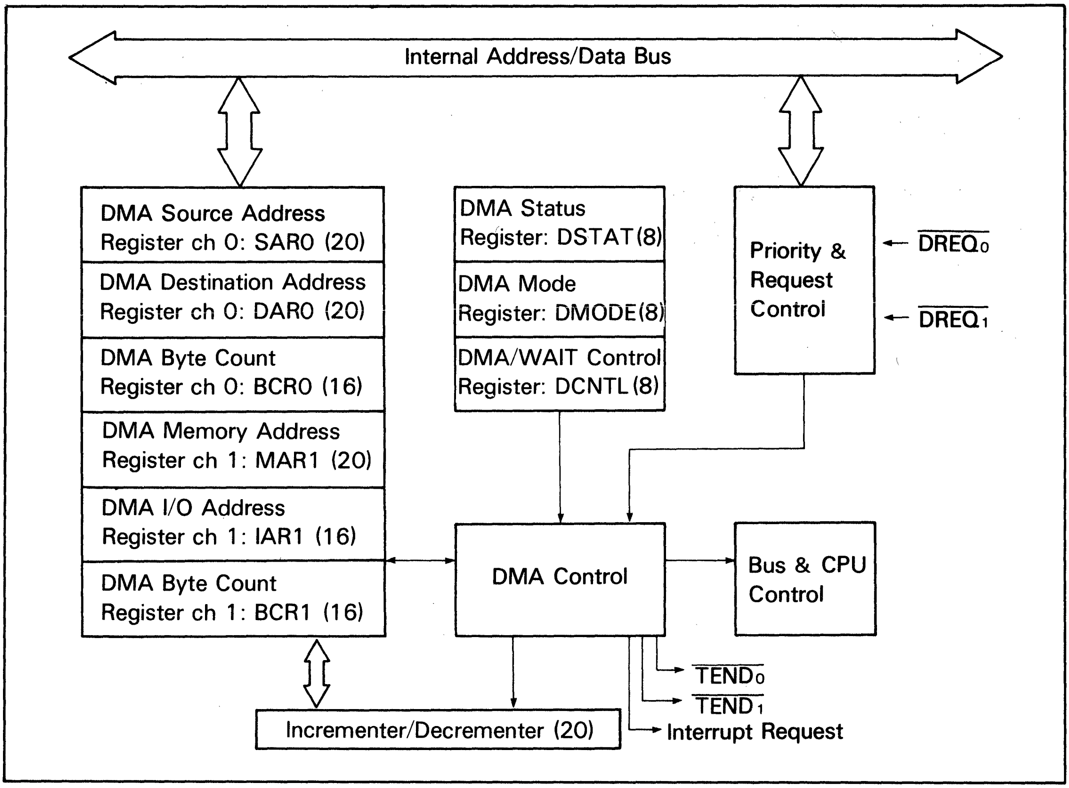


Figure 10-1. DMAC Block Diagram

10.2 DMAC Register Description

10.2.1 DMA Source Address Register Channel 0 (SAR0: I/O Address = 20H to 22H)

SAR0 specifies the physical source address for channel 0 transfers. The register contains 20 bits and may specify up to 1-Mbyte memory addresses or up to 64-kbyte I/O addresses. The channel 0 source can be memory, I/O, or memory-mapped I/O.

10.2.2 DMA Destination Address Register Channel 0 (DAR0: I/O Address = 23H to 25H)

DAR0 specifies the physical destination address for channel 0 transfers. The register contains 20 bits and may specify up to 1-Mbyte memory addresses or up to 64-kbyte I/O addresses. The channel 0 destination can be memory, I/O, or memory-mapped I/O.

10.2.3 DMA Byte Count Register Channel 0 (BCR0: I/O Address = 26H to 27H)

BCR0 specifies the number of bytes to be transferred. This register contains 16 bits and may specify up to 64-kbyte transfers. When one byte is transferred, the register is decremented by one. If n bytes should be transferred, n must be stored before the DMA operation.

10.2.4 DMA Memory Address Register Channel 1 (MAR1: I/O Address = 28H to 2AH)

MAR1 specifies the physical memory address for channel 1 transfers. This may be destination or source memory address. This register contains 20 bits and may specify up to 1-Mbyte memory addresses.

10.2.5 DMA I/O Address Register Channel 1 (IAR1: I/O Address = 2BH to 2CH)

IAR1 specifies the I/O address for channel 1 transfers. This may be destination or source I/O address. This register contains 16 bits and may specify up to 64-kbyte I/O addresses.

10.2.6 DMA Byte Count Register Channel 1 (BCR1: I/O Address = 2EH to 2FH)

BCR1 specifies the number of bytes to be transferred. This register contains 16 bits and may specify up to 64-kbyte transfers. When one byte is transferred, the register is decremented by one.

10.2.7 DMA Status Register (DSTAT)

DSTAT (figure 10-2) enables and disable DMA transfer and DMA termination interrupts. DSTAT also allows determining the status of a DMA transfer, that is, completed or in progress.

DMA Status Register (DSTAT: I/O Address = 30H)

bit	7	6	5	4	3	2	1	0
	DE1	DE0	$\overline{\text{DWE1}}$	$\overline{\text{DWE0}}$	DIE1	DIE0	–	DME
	R/W	R/W	W	W	R/W	R/W		R

Figure 10-2. DMA Status Register

DE1: DMA Enable Channel 1 (Bit 7): When DE1 = 1 and DME = 1, channel 1 DMA is enabled. When a DMA transfer terminates (BCR1 = 0), DE1 is reset to 0 by the DMAC. When DE1 = 0 and the DMA interrupt is enabled (DIE1 = 1), a DMA interrupt request is made to the CPU.

To perform a software write to DE1, 0 should be written to $\overline{\text{DWE1}}$ during the same register write access. Writing DE1 to 0 disables channel 1 DMA, but DMA can be restarted. Writing 1 to DE1 enables channel 1 DMA and automatically sets DME (DMA main enable) to 1. DE1 is cleared to 0 during reset.

DE0: DMA Enable Channel 0 (Bit 6): When DE0 = 1 and DME = 1, channel 0 DMA is enabled. When a DMA transfer terminates (BCR0 = 0), DE0 is reset to 0 by the DMAC. When DE0 = 0 and the DMA interrupt is enabled (DIE0 = 1), a DMA interrupt request is made to the CPU.

To perform a software write to DE0, 0 should be written to $\overline{\text{DWE0}}$ during the same register write access. Writing DE0 to 0 disables channel 0 DMA. Writing 1 to DE0 enables channel 0 DMA and automatically sets DME (DMA main enable) to 1. DE0 is cleared to 0 during reset.

$\overline{\text{DWE1}}$: DE1 Bit Write Enable (Bit 5): When performing any software write to DE1, 0 should be written to $\overline{\text{DWE1}}$ during the same access. $\overline{\text{DWE1}}$ does not keep write value of 0 is always read as 1.

$\overline{\text{DWE0}}$: DE0 Bit Write Enable (Bit 4): When performing any software write to DE0, 0 should be written to $\overline{\text{DWE0}}$ during the same access. $\overline{\text{DWE0}}$ does not keep write value of 0 is always read as 1.

DIE1: DMA Interrupt Enable Channel 1 (Bit 3): When DIE1 is set to 1, the termination of channel 1 DMA transfer (indicated when DE1 = 0) causes a CPU interrupt request to be generated. When DIE1 = 0, the channel 1 DMA termination interrupt is disabled. DIE1 is cleared to 0 during reset.

DIE0: DMA Interrupt Enable Channel 0 (Bit 2): When DIE0 is set to 1, the termination of channel 0 DMA transfer (indicated when DE0 = 0) causes a CPU interrupt request to be generated. When DIE0 = 0, the channel 0 DMA termination interrupt is disabled. DIE0 is cleared to 0 during reset.

DME: DMA Main Enable (Bit 0): A DMA operation is only enabled when its DE bit (DE0 for channel 0, DE1 for channel 1) and the DME bit are set to 1.

When $\overline{\text{NMI}}$ occurs, DME is reset to 0, thus disabling DMA activity during the $\overline{\text{NMI}}$ interrupt service routine. To restart DMA, 1 should be written to DE0 and/or DE1 (even if the contents are already 1). This automatically sets DME to 1, allowing DMA operations to continue. Note that DME cannot be directly written to. It is cleared to 0 by $\overline{\text{NMI}}$ or indirectly set to 1 by setting DE0 and/or DE1 to 1. DME is cleared to 0 during reset.

10.2.8 DMA Mode Register (DMODE)

DMODE (figure 10-3) sets the addressing and transfer mode for channel 0.

DMA Mode Register (DMODE: I/O Address = 31H)

bit	7	6	5	4	3	2	1	0
	–	–	DM1	DM0	SM1	SM0	MMOD	–
			R/W	R/W	R/W	R/W	R/W	

Figure 10-3. DMA Mode Register

DM1, DM0: Destination Mode Channel 0 (Bits 5, 4): DM1 and DM0 specify whether the destination for channel 0 transfers is memory, I/O, or memory-mapped I/O and the corresponding address modifier (table 10-1). DM1 and DM0 are cleared to 0 during reset.

Table 10-1. Destination

DM1	DM0	Memory/I/O	Address Increment/Decrement
0	0	Memory	+1
0	1	Memory	−1
1	0	Memory	Fixed
1	1	I/O	Fixed

SM1, SM0: Source Mode Channel 0 (Bits 3, 2): SM1 and SM0 specify whether the source for channel 0 transfers is memory, I/O, or memory-mapped I/O and the corresponding address modifier (table 10-2). SM1 and SM0 are cleared to 0 during reset.

Table 10-2. Source

SM1	SM0	Memory/I/O	Address Increment/Decrement
0	0	Memory	+1
0	1	Memory	−1
1	0	Memory	Fixed
1	1	I/O	Fixed

Table 10-3 shows all DMA transfer mode combinations of DM0, DM1, SM0, SM1. Since I/O to/from I/O transfers are not implemented, twelve combinations are available.

Table 10-3. Transfer Mode Combinations

DM1	DM0	SM1	SM0	Transfer Mode	Address Increment/Decrement
0	0	0	0	Memory to Memory	SAR0 + 1, DAR0 + 1
0	0	0	1	Memory to Memory	SAR0 − 1, DAR0 + 1
0	0	1	0	Memory* to Memory	SAR0 fixed, DAR0 + 1
0	0	1	1	I/O to Memory	SAR0 fixed, DAR0 + 1
0	1	0	0	Memory to Memory	SAR0 + 1, DAR0 − 1
0	1	0	1	Memory to Memory	SAR0 − 1, DAR0 − 1
0	1	1	0	Memory* to Memory	SAR0 fixed, DAR0 − 1
0	1	1	1	I/O to Memory	SAR0 fixed, DAR0 − 1
1	0	0	0	Memory to Memory*	SAR0 + 1, DAR0 fixed
1	0	0	1	Memory to Memory*	SAR0 − 1, DAR0 fixed
1	0	1	0	reserved	
1	0	1	1	reserved	
1	1	0	0	Memory to I/O	SAR0 + 1, DAR0 fixed
1	1	0	1	Memory to I/O	SAR0 − 1, DAR0 fixed
1	1	1	0	reserved	
1	1	1	1	reserved	

Note: * = includes memory mapped I/O

MMOD: Memory Mode Channel 0 (Bit 1): When channel 0 is configured for memory to/from memory transfers, the external $\overline{DREQ_0}$ input is not used to control the transfer timing. Instead, two automatic transfer timing modes are selectable – burst (MMOD = 1) and cycle steal (MMOD = 0). For burst memory to/from memory transfers, the DMAC will sieze control of the bus continuously until the DMA transfer completes (as shown by the byte count register = 0). In cycle steal mode, the CPU is given a cycle for each DMA byte transfer cycle until the transfer is completed.

For channel 0 DMA with I/O source or destination, the $\overline{DREQ_0}$ input times the transfer and thus MMOD is ignored. MMOD is cleared to 0 during reset.

10.2.9 DMA/Wait Control Register (DCNTL)

DCNTL (figure 10-4) controls the insertion of wait states into DMAC (and CPU) accesses to memory or I/O. Also, the DMA request mode for each $\overline{DREQ}$ ($\overline{DREQ_0}$ and $\overline{DREQ_1}$) input is defined as level or edge sense. DCNTL also sets the DMA transfer mode for channel 1, which is limited to memory to/from I/O transfers.

DMA/Wait Control Register (DCNTL: I/O Address = 32H)

bit	7	6	5	4	3	2	1	0
	MWI1	MWI0	IWI1	IWI0	DMS1	DMS0	DIM1	DIM0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Figure 10-4. DMA/Wait Control Register

MWI1, MWI0: Memory Wait Insertion (Bits 7-6): MWI1 and MWI0 specify the number of wait states introduced into CPU or DMAC memory access cycles. MWI1 and MWI0 are set to 1 during reset. See section 4, Wait State Generator for details.

IWI1, IWI0: I/O Wait Insertion (Bits 5-4): IWI1 and IWI0 specify the number of wait states introduced into CPU or DMAC I/O access cycles. IWI1 and IWI0 are set to 1 during reset. See section 4, Wait State Generator for details.

DMS1, DMS0: DMA Request Sense (Bits 3-2): DMS1 and DMS0 specify the DMA request sense for channel 0 ($\overline{DREQ_0}$) and channel 1 ($\overline{DREQ_1}$) respectively. When reset to 0, the input is level sensitive. When set to 1, the input is edge sensitive. DMS1 and DMS0 are cleared to 0 during reset.

DIM1, DIM0: DMA Channel 1 I/O and Memory Mode (Bits 1-0): DIM1 and DIM0 specify the source/destination and address modifier for channel 1 memory to/from I/O transfer modes (table 10-4). IM1 and IM0 are cleared to 0 during reset.

Table 10-4. Channel 1 Transfer Mode

DIM1	DIM0	Transfer Mode	Address Increment/Decrement
0	0	Memory to I/O	MAR1+1, IAR1 fixed
0	1	Memory to I/O	MAR1−1, IAR1 fixed
1	0	I/O to Memory	IAR1 fixed, MAR1+1
1	1	I/O to Memory	IAR1 fixed, MAR1−1

10.3 DMA Operation

This section discusses the three DMA operation modes for channel 0, memory to/from memory, memory to/from I/O and memory to/from memory mapped I/O. In addition, the operation of channel 0 DMA with the on-chip ASCI (asynchronous serial communication interface) and channel 1 DMA are described.

10.3.1 Memory to/from Memory—Channel 0

For memory to/from memory transfers, the external $\overline{DREQ_0}$ input is not used for DMA transfer timing. Rather, the DMA operation is timed in one of two programmable modes—burst or cycle steal. In both modes, the DMA operation will automatically proceed until termination as shown by byte count (BCR0) = 0.

In burst mode, the DMA operation will proceed until termination. In this case, the CPU cannot perform any program execution until the DMA operation is completed.

In cycle steal mode, DMA and CPU operation alternate after each DMA byte transfer until the DMA is completed. The sequence:

1 CPU machine cycle
DMA byte transfer

is repeated until DMA is completed. Figure 10-5 shows cycle steal mode DMA timing.

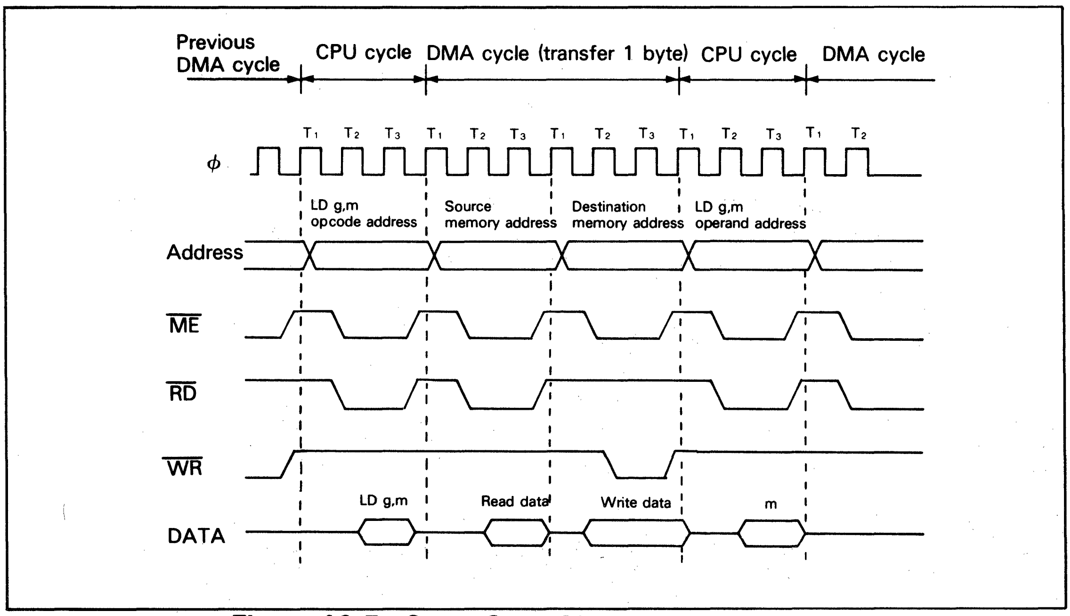


Figure 10-5. Cycle Steal Mode DMA Timing

To initiate memory to/from memory DMA transfer for channel 0, perform the following operations:

1. Load the memory source and destination addresses into SAR0 and DAR0.
2. Specify memory to/from memory mode and address increment/decrement in the SM0, SM1, DM0 and DM1 bits of DMODE.
3. Load the number of bytes to transfer in BCR0.
4. Specify burst or cycle steal mode in the MMOD bit of DCNTL.
5. Program DE0 = 1 (with $\overline{\text{DWE0}}$ = 0 in the same access) in DSTAT and the DMA operation will start 1 machine cycle later. If an interrupt occurs at the same time, the DIE0 bit should be set to 1.

10.3.2 Memory to/from I/O (Memory-Mapped I/O)—Channel 0

Memory to/from I/O (and memory to/from memory-mapped I/O) the $\overline{\text{DREQ}_0}$ input is used to time the DMA transfers. In addition, the $\overline{\text{TEND}_0}$ (transfer end) output is used to indicate the last (byte count register BCR0 = 00H) transfer.

The $\overline{\text{DREQ}_0}$ input can be programmed as level or edge sensitive.

This transfer mode can be applied to the internal I/O port.

When programmed for level sense, the DMA operation begins when $\overline{\text{DREQ}_0}$ is sampled low. If $\overline{\text{DREQ}_0}$ is sampled high, after the next DMA byte transfer, control is relinquished to the HD647180X CPU. As shown in Figure 10-6, $\overline{\text{DREQ}_0}$ is sampled at the rising edge of the clock cycle prior to T_3, that is, either T_2 or Tw.

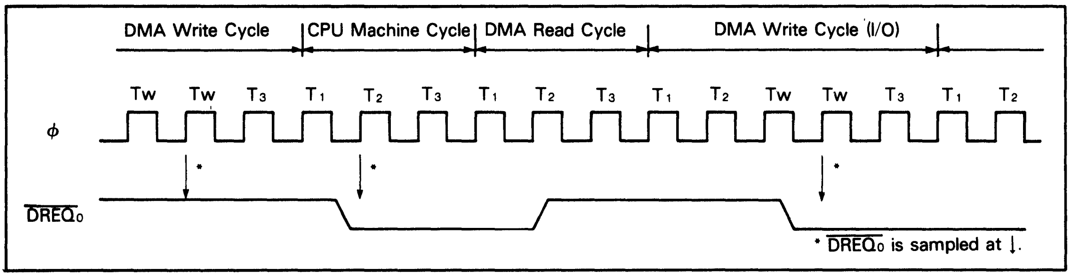


Figure 10-6. CPU Operation and DMA Operation ($\overline{\text{DREQ}_0}$ Level Sensitive)

When programmed for edge sense, DMA operation begins at the falling edge of $\overline{\text{DREQ}_0}$. If another falling edge is detected before the rising edge of the clock prior to T_3 during DMA write cycle (that is, T_2 or Tw), the DMAC continues operating. If an edge is not detected, the CPU is given control after the current byte DMA transfer completes. The CPU will continue operating until a $\overline{\text{DREQ}_0}$ falling edge is detected before the rising edge of the clock prior to T_3 at which time the DMA operation will (re)start. Figure 10-7 shows the edge sense DMA timing.

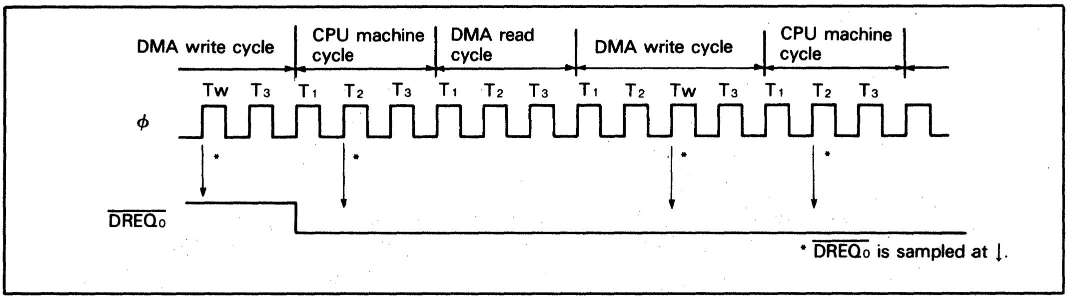


Figure 10-7. CPU Operation and DMA Operation ($\overline{DREQ_0}$ Edge Sensitive)

During the transfers for channel 0, the $\overline{TEND_0}$ output will go low synchronous with the write cycle of the last (BCR0 = 00H) DMA transfer as shown in figure 10-8.

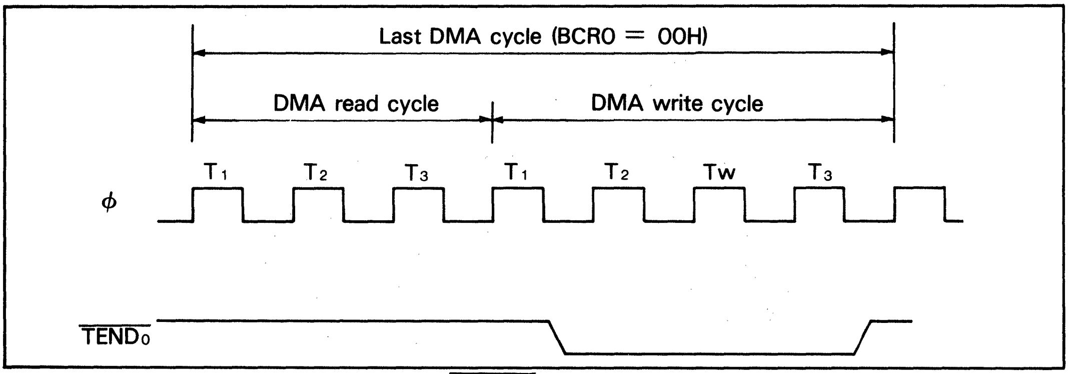


Figure 10-8. $\overline{TEND_0}$ Output Timing

The $\overline{DREQ_0}$ and $\overline{TEND_0}$ pins are programmably multiplexed with the CKA0 and CKA1 ASCI clock input/outputs. However, when DMA channel 0 is programmed for memory to/from I/O (and memory to/from memory-mapped I/O) transfers, the CKA0/$\overline{DREQ_0}$ pin automatically functions as input pin even if it has been programmed as output pin for CKA0. The CKA1/$\overline{TEND_0}$ pin is selected as output pin for $\overline{TEND_0}$ by setting CKA1D to 1 in CNTLA1.

Figure 10-9 shows memory to/from memory-mapped I/O transfer timing and figure 10-10 shows memory to I/O transfer timing.

To initiate memory to/from I/O (and memory to/from memory-mapped I/O) DMA transfer for channel 0, perform the following operations:

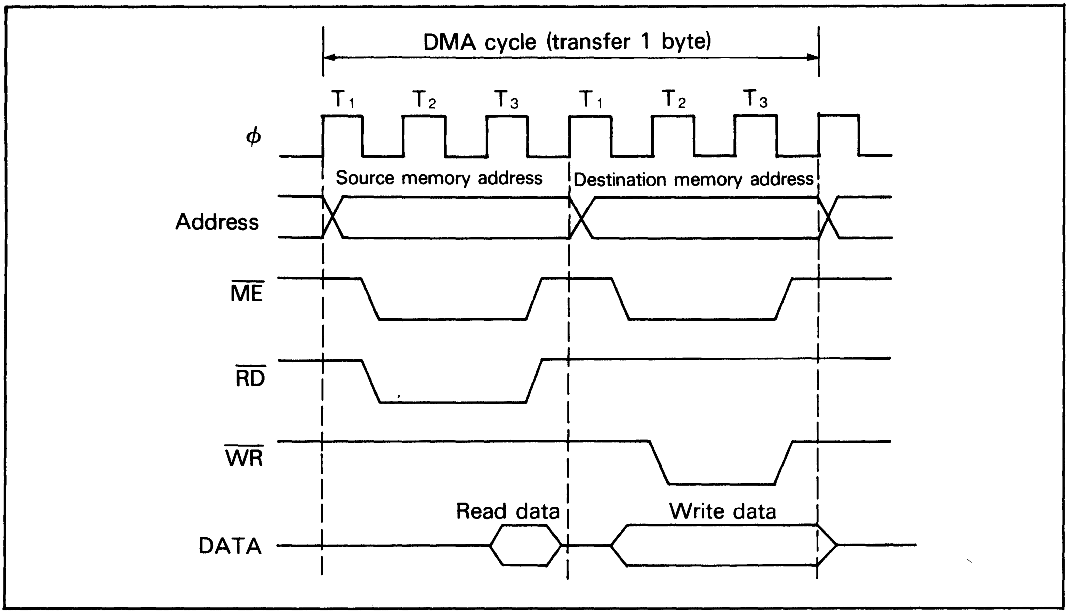


Figure 10-9. DMA Cycle (Memory to/from Memory-Mapped I/O (Memory))

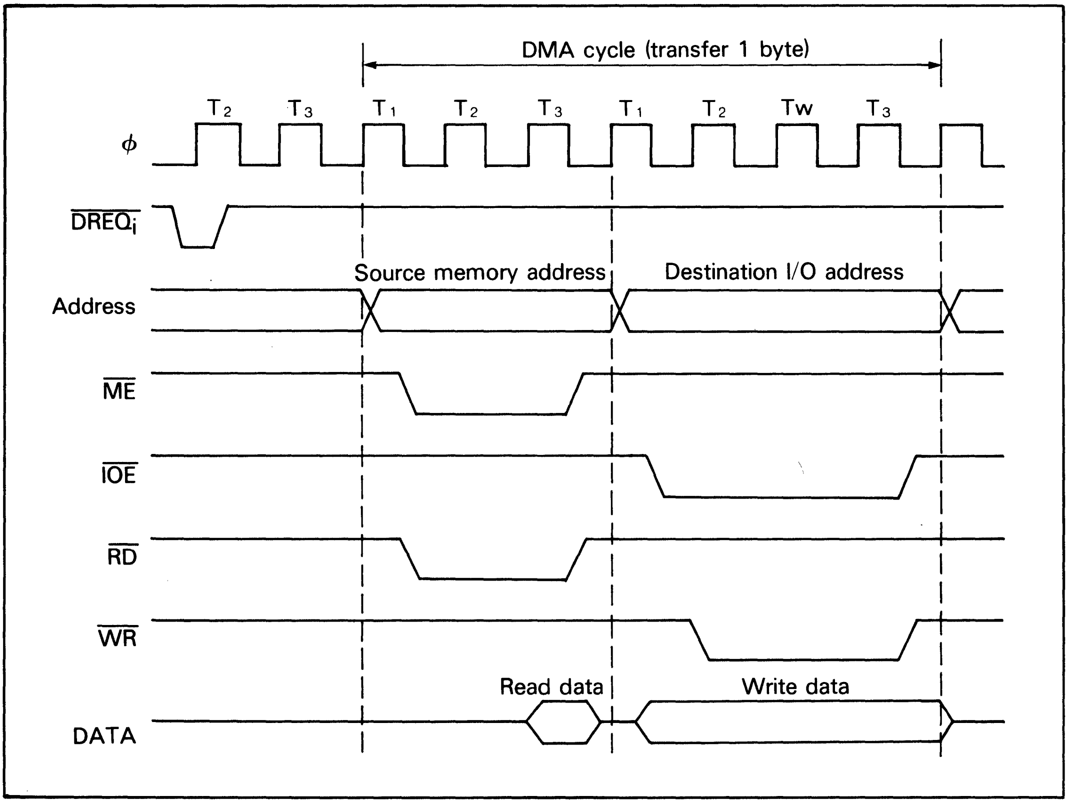


Figure 10-10. DMA Cycle (Memory to I/O)

1. Load the memory and I/O or memory-mapped I/O source and destination addresses (A_9-A_{19}) into SAR0 and DAR0. Note that I/O addresses (not memory mapped I/O) are limited to 16 bits (A_0-A_{15}).
2. Specify memory to/from I/O or memory to/from memory-mapped I/O mode and address increment/decrement in the SM0, SM1, DM0, and DM1 bits of DMODE.
3. Load the number of bytes to transfer in BCR0.
4. Specify whether $\overline{DREQ_0}$ is edge or level sense by programming the DMS0 bit of DCNTL.
5. Enable or disable DMA termination interrupt with the DIE0 bit in DSTAT.
6. Program DE0 = 1 (with $\overline{DWE0}$ = 0 in the same access) in DSTAT and the DMA operation will begin under the control of the $\overline{DREQ_0}$ input.

10.3.3 Memory to/from ASCI—Channel 0

Channel 0 has the extra capability to support DMA transfer to and from the on-chip two-channel ASCI. In this case the external $\overline{DREQ_0}$ input is not used for DMA timing. Rather, the ASCI status bits are used to generate an internal $\overline{DREQ_0}$. The TDRE (transmit data register empty) bit and the RDRF (receive data register full) bit generate an internal $\overline{DREQ_0}$ for ASCI transmission and reception respectively.

To initiate memory to/from ASCI DMA transfer, perform the following operations:

1. Load the source and destination addresses into SAR0 and DAR0. Specify the I/O (ASCI) address as follows:

   · Bits A_0-A_7 should be contain the address of the ASCI channel transmitter or receiver (I/O addresses 06H-09H).
   · Bits A_8-A_{15} should equal 0.
   · Bits A_{17}-A_{16} should be set according to table 10-5 to enable use of the appropriate ASCI status bit as an internal DMA request.

2. Specify memory to/from I/O transfer mode and address increment/decrement in the SM0, SM1, DM0, and DM1 bits of DMODE.
3. Load the number of bytes to transfer in BCR0.
4. The DMA request sense mode (DMS0 bit in DCNTL) MUST be specified as edge sense.
5. Enable or disable DMA termination interrupt with the DIE0 bit in DSTAT.
6. Program DE0 = 1 (with $\overline{DWE0}$ = 0 in the same access) in DSTAT and the DMA operation with the ASCI will begin under control of the ASCI generated internal DMA request.

Table 10-5. DMA Request

SAR19	SAR18	SAR17	SAR16	DMA Transfer Request
X	X	0	0	$\overline{DREQ_0}$
X	X	0	1	RDRF (ASCI channel 0)
X	X	1	0	RDRF (ASCI channel 1)
X	X	1	1	reserved

X: Don't care

DAR19	DAR18	DAR17	DAR16	DMA Transfer Request
X	X	0	0	$\overline{DREQ_0}$
X	X	0	1	TDRE (ASCI channel 0)
X	X	1	0	TDRE (ASCI channel 1)
X	X	1	1	reserved

X: Don't care

The ASCI receiver or transmitter being used for DMA must be initialized to allow the first DMA transfer to begin. The ASCI receiver must be empty as shown by RDRF = 0. The ASCI transmitter must be full as shown by TDRE = 0. Thus, the first byte should be written to the ASCI transmit data register under program control. The remaining bytes will be transferred using DMA.

10.3.4 Channel 1 DMA

DMAC channel 1 can perform memory to/from I/O transfers. Except for different registers and status/control bits, operation is exactly the same as described for channel 0 memory to/from I/O DMA.

To initiate DMA channel 1 memory to/from I/O transfer perform the following operations:

1. Load the memory address (20 bits) into MAR1.
2. Load the I/O address (16 bits) into IAR1.
3. Program the source/destination and address increment/decrement mode using the DIM1 and DIM0 bits in DCNTL.
4. Specify whether $\overline{DREQ_1}$ is level or edge sense in the DMS1 bit in DCNTL.
5. Enable or disable DMA termination interrupt with the DIE1 bit in DSTAT.
6. Program DE1 = 1 (with $\overline{DWE1}$ = 0 in the same access) in DSTAT and the DMA operation with the external I/O device will begin using the external $\overline{DREQ_1}$ input and $\overline{TEND_1}$ output.

10.4 DMA Bus Timing

When memory (and memory-mapped I/O) is specified as a source or destination, $\overline{ME}$ goes low during the memory access. When I/O is specified as a source or destination, $\overline{IOE}$ goes low during the I/O access.

When I/O (and memory-mapped I/O) is specified as a source or destination, the DMA timing is controlled by the external $\overline{DREQ}$ input and the $\overline{TEND}$ output indicates DMA termination. Note that external I/O devices may not overlap addresses with internal I/O and control registers, even using DMA.

For I/O accesses, one wait state is automatically inserted. Additional wait states can be inserted by programming the on-chip wait state generator or using the external $\overline{WAIT}$ input. Note that for memory-mapped I/O accesses, this automatic I/O wait state is not inserted.

For memory to memory transfers (channel 0 only), the external $\overline{DREQ_0}$ input is ignored. Automatic DMA timing is programmed as either burst or cycle steal.

When a DMA memory address carry/borrow between bits A_{15} and A_{16} of the address bus occurs (when crossing 64-kbyte boundaries), the minimum bus cycle is extended to four clocks by automatic insertion of one internal Ti state.

10.5 DMAC Channel Priority

For simultaneous $\overline{DREQ_0}$ and $\overline{DREQ_1}$ requests, channel 0 has priority over channel 1. When channel 0 is performing a memory to/from memory transfer, channel 1 cannot operate until the channel 0 operation has terminated. If channel 1 is operating, channel 0 cannot operate until channel 1 releases control of the bus.

10.6 DMAC and $\overline{BUSREQ}$, $\overline{BUSACK}$

The $\overline{BUSREQ}$ and $\overline{BUSACK}$ inputs allow another bus master to take control of the HD647180X bus. $\overline{BUSREQ}$ and $\overline{BUSACK}$ have priority over the on-chip DMAC and will suspend DMAC operation. The DMAC releases the bus to the external bus master at the breakpoint of the DMAC memory or I/O access. Since a single-byte DMAC transfer requires a read and a write cycle, it is possible for the DMAC to be suspended after the DMAC read, but before the DMAC write. Even in this case, when the external master releases the HD647180X bus ($\overline{BUSREQ}$ high), the on-chip DMAC will correctly continue the suspended DMA operation.

10.7 DMAC Internal Interrupts

Figure 10-11 illustrates the internal DMA interrupt request generation circuit.

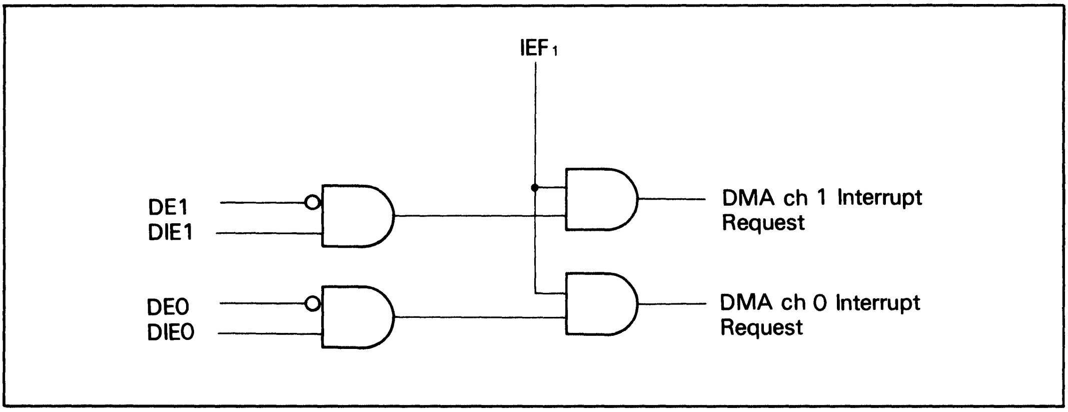


Figure 10-11. DMAC Interrupt Request Circuit Diagram

DE0 and DE1 are automatically cleared to 0 by the HD647180X at the completion (byte count = 0) of a DMA operation for channel 0 and channel 1 respectively. They remain 0 until a 1 is written. Since DE0 and DE1 use level sense, an interrupt will occur if the CPU IEF_1 flag is set to 1. Therefore, the DMA termination interrupt service routine should disable further DMA interrupts (by programming the channel DIE bit = 0) before enabling CPU interrupts (that is, IEF_1 is set to 1). After reloading the DMAC address and count registers, the DIE bit can be set to 1 to reenable the channel interrupt, and at the same time DMA can be restarted by programming the channel DE bit = 1.

10.8 DMAC and $\overline{\text{NMI}}$

$\overline{\text{NMI}}$, unlike all other interrupts, automatically disables DMAC operation by clearing the DME bit of DSTAT. Thus, the $\overline{\text{NMI}}$ interrupt service routine may respond to time critical events without delay due to DMAC bus usage. Also, $\overline{\text{NMI}}$ can be effectively used as an external DMA abort input, recognizing that both channels are suspended by the clearing of DME.

If the falling edge of $\overline{\text{NMI}}$ occurs before the falling clock of the state prior to T_3 (T_2 or Tw) of the DMA write cycle, the DMAC will be suspended and the CPU will start the $\overline{\text{NMI}}$ response at the end of the current cycle.

By setting a channel's DE bit to 1, that channel's operation can be restarted, and DMA will correctly resume from the point at which it was suspended by $\overline{\text{NMI}}$. See figure 10-12 for details.

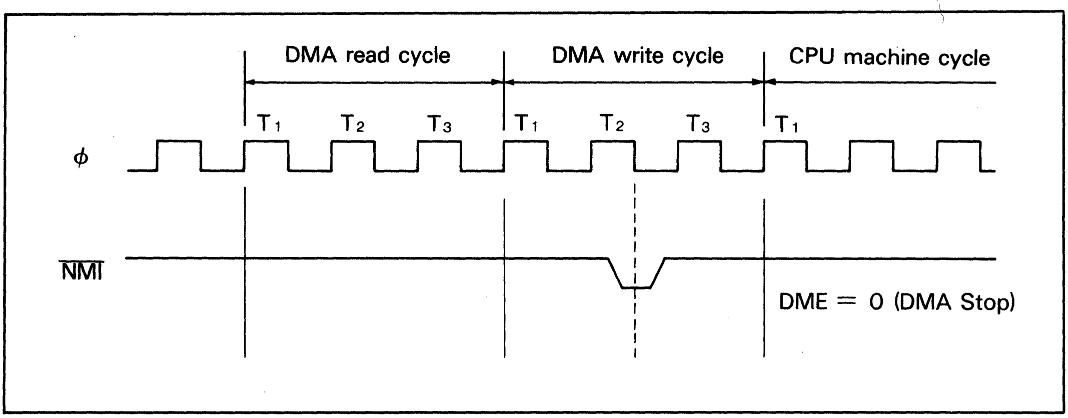


Figure 10-12. $\overline{\text{NMI}}$ and DMA Operation

10.9 DMAC and Reset

During reset the bits in DSTAT, DMODE, and DCNTL are initialized as stated in their individual register descriptions. Any DMA operation in progress is stopped, allowing the CPU to use the bus to perform the reset sequence. However, the address register (SAR0, DAR0, MAR1, IAR1) and byte count register (BCR0, BCR1) contents are not changed during reset.

SECTION 11. ASYNCHRONOUS SERIAL COMMUNICATION INTERFACE (ASCI)

The HD647180X on-chip ASCI has two independent full-duplex channels. Because the following functions are fully programmable, the ASCI can directly communicate with a wide variety of standard UARTs (universal asynchronous receiver/transmitter) including the HD6350 CMOS ACIA and the serial communication interface (SCI) contained on the HD6301 series CMOS single-chip controllers.

The key functions for ASCI are shown below. Each channel is independently programmable.

- Full duplex communication
- 7- or 8-bit data length
- Program controlled 9th data bit for multiprocessor communication
- 1 or 2 stop bits
- Odd, even, no parity
- Parity, overrun, framing error detection
- Programmable baud rate generator, ÷16 and ÷64 modes
  Speed to 38.4 kbits per second (CPU f_C = 6.144 MHz)
- Modem control signals
  - Channel 0: $\overline{DCD_0}$, $\overline{CTS_0}$ and $\overline{RTS_0}$
  - Channel 1: $\overline{CTS_1}$
- Programmable interrupt condition enable and disable
- Operation with on-chip DMAC

11.1 ASCI Block Diagram

Figure 11-1 shows the ASCI block diagram.

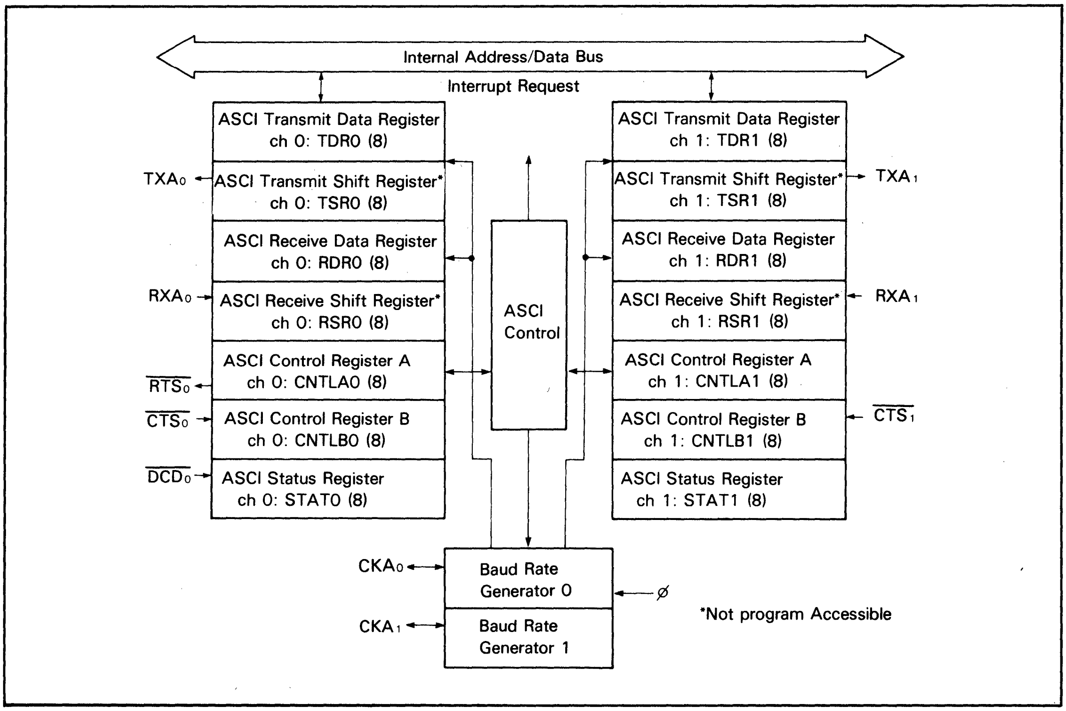


Figure 11-1. ASCI Block Diagram

11.2 ASCI Register Description

11.2.1 ASCI Transmit Shift Register 0, 1 (TSR0, TSR1)

When the ASCI transmit shift register receives data from the ASCI transmit data register (TDR), the data is shifted out to the TXA pin. When transmission is completed, the next byte (if available) is automatically loaded from TDR into TSR and the next transmission starts. If no data is available for transmission, TSR idles by outputting a continuous high level. This register is not program accessible.

11.2.2 ASCI Transmit Data Register 0, 1 (TDR0, TDR1: I/O Address = 06H, 07H)

Data written to the ASCI transmit data register is transferred to the TSR as soon as TSR is empty. Data can be written to TDR while TSR is shifting out the previous byte of data. Thus, the ASCI transmitter is double bufferred.

Data can be written into and read from the ASCI transmit data register.

If data is read from the ASCI transmit data register, the ASCI data transmit operation won't be affected.

11.2.3 ASCI Receive Shift Register 0, 1 (RSR0, RSR1)

The ASCI receive shift register receives data shifted in on the RXA pin. When full, data is automatically transferred to the ASCI receive data register (RDR) if it is empty. If RSR is not empty when the next incoming data byte is shifted in, an overrun error occurs. This register is not program accessible.

11.2.4 ASCI Receive Data Register 0, 1 (RDR0, RDR1: I/O Address = 08H, 09H)

When a complete incoming data byte is assembled in RSR, it is automatically transferred to the RDR if RDR is empty. The next incoming data byte can be shifted into RSR while RDR contains the previous received data byte. Thus, the ASCI receiver is double buffered.

The ASCI receive data register is a read-only register. However, if RDRF = 0, data can be written into the ASCI receive data register, and the data can be read.

11.2.5 ASCI Status Register 0, 1 (STAT0, STAT1)

Each channel status register (figure 11-2) allows interrogation of ASCI communication, error, and modem control signal status as well as enabling and disabling of ASCI interrupts.

ASCI Status Register 0 (STAT0: I/O Address = 04H)

bit	7	6	5	4	3	2	1	0
	RDRF	OVRN	PE	FE	RIE	$\overline{DCD_0}$	TDRE	TIE
	R	R	R	R	R/W	R	R	R/W

ASCI Status Register 1 (STAT1: I/O Address = 05H)

bit	7	6	5	4	3	2	1	0
	RDRF	OVRN	PE	FE	RIE	CTS1E	TDRE	TIE
	R	R	R	R	R/W	R/W	R	R/W

Figure 11-2. ASCI Status Registers 0, 1

RDRF: Receive Data Register Full (Bit 7): RDRF is set to 1 when an incoming data byte is loaded into RDR. Note that if a framing or parity error occurs, RDRF is still set and the receive data (which generated the error) is still loaded into RDR. RDRF is cleared to 0 by reading RDR, when the $\overline{DCD_0}$ input is high, in I/O stop mode, and during reset.

OVRN: Overrun Error (Bit 6): OVRN is set to 1 when RDR is full and RSR becomes full. OVRN is cleared to 0 when the EFR bit (Error Flag Reset) of CNTLA is written to 0, when $\overline{DCD_0}$ is high, in I/O stop mode and during reset.

PE: Parity Error (Bit 5): PE is set to 1 when a parity error is detected in an incoming data byte and ASCI parity detection is enabled (the MOD1 bit of CNTLA is set to 1). PE is cleared to 0 when 0 is written to the EFR bit (error flag reset) of CNTLA when $\overline{DCD_0}$ is high, in I/O stop mode, and during reset.

FE: Framing Error (Bit 4): If a receive data byte frame is delimited by an invalid stop bit (that is, 0, should be 1), FE is set to 1. FE is cleared to 0 when 0 is written to the EFR bit (error flag reset) of CNTLA when $\overline{DCD_0}$ is high, in I/O stop mode and during reset.

RIE: Receive Interrupt Enable (Bit 3): RIE should be set to 1 to enable ASCI receive interrupt requests. When RIE is set to 1, if any of the flags RDRF, OVRN, PE, FE become set to 1, an interrupt request is generated. For channel 0, an interrupt will also be generated by the transition of the external $\overline{DCD_0}$ input from low to high. RIE is cleared to 0 during reset.

$\overline{DCD_0}$: Data Carrier Detect (Bit 2 STAT0): Channel 0 has an external $\overline{DCD_0}$ input pin. The $\overline{DCD_0}$ bit is set to 1 when the $\overline{DCD_0}$ input is high. It is cleared to 0 on the first read of STAT0 following the $\overline{DCD_0}$ input transition from high to low and during reset. When $\overline{DCD_0}$ = 1, receiver unit is reset and receiver operation is inhibited.

CTS1E: Channel 1 $\overline{CTS}$ Enable (Bit 2 STAT1): Channel 1 has an external $\overline{CTS_1}$ input which is multiplexed with the receive data pin (RXS) for the CSI/O (clocked serial I/O port). Setting CTS1E to 1 selects the $\overline{CTS_1}$ function and clearing CTS1E to 0 selects the RXS function.

TDRE: Transmit Data Register Empty (Bit 1): TDRE = 1 indicates that the TDR is empty and the next transmit data byte can be written to TDR. After the byte is written to TDR, TDRE is cleared to 0 until the ASCI transfers the byte from the TDR to the TSR, at which time TDRE is again set to 1. TDRE is set to 1 in I/O stop mode and during reset. When the external $\overline{CTS}$ input is high, TDRE is reset to 0.

TIE: Transmit Interrupt Enable (Bit 0): TIE should be set to 1 to enable ASCI transmit interrupt requests. If TIE = 1, an interrupt will be requested when TDRE = 1. TIE is cleared to 0 during reset.

11.2.6 ASCI Control Register A 0, 1 (CNTLA0, CNTLA1)

Each ASCI channel control register A (figure 11-3) configures the major operating modes, such as receiver/transmitter enable and disable, data format, and multiprocessor communication mode.

ASCI Control Register A 0 (CNTLA0: I/O Address = 00H)

bit	7	6	5	4	3	2	1	0
	MPE	RE	TE	$\overline{RTS_0}$	MPBR/EFR	MOD2	MOD1	MOD0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

ASCI Control Register A 1 (CNTLA1: I/O Address = 01H)

bit	7	6	5	4	3	2	1	0
	MPE	RE	TE	CKA1D	MPBR/EFR	MOD2	MOD1	MOD0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Figure 11-3. ASCI Control Registers 0, 1

MPE: Multi Processor Mode Enable (Bit 7): The ASCI has a multiprocessor communication mode which utilizes an extra data bit for selective communication when a number of processors share a common serial bus. Multiprocessor data format is selected when the MP bit in CNTLB is set to 1. If multiprocessor mode is not selected (MP bit in CNTLB = 0), MPE has no effect. If multiprocessor mode is selected, MPE enables or disables the 'wake-up' feature as follows. If MPE is set to 1, only received bytes in which the MPB (multiprocessor bit) = 1 can affect the RDRF and error flags. Effectively, other bytes (with MPB = 0) are ignored by the ASCI. If MPE is reset to 0, all bytes, regardless of the state of the MPB data bit, affect the RDRF and error flags. MPE is cleared to 0 during reset.

RE: Receiver Enable (Bit 6): When RE is set to 1, the ASCI receiver is enabled. When RE is reset to 0, the receiver is disabled and any receive operation in progress is interrupted. However, the RDRF and error flags are not reset and the previous contents of RDRF and error flags are held. RE is cleared to 0 in I/O stop mode and during reset.

TE: Transmitter Enable (Bit 5): When TE is set to 1, the ASCI transmitter is enabled. When TE is reset to 0, the transmitter is disabled and any transmit operation in progress is interrupted. However, the TDRE flag is not reset and the previous contents of TDRE are held. TE is cleared to 0 in I/O stop mode and during reset.

$\overline{RTS_0}$: Request to Send Channel 0 (Bit 4 CNTLA0): When $\overline{RTS_0}$ is reset to 0, the $\overline{RTS_0}$ output pin will go low. When $\overline{RTS_0}$ is set to 1, the $\overline{RTS_0}$ output immediately goes high. $\overline{RTS_0}$ is set to 1 during reset.

CKA1D: CKA1 Clock Disable (Bit 4 CNTLA1): When CKA1D is set to 1, the multiplexed CKA1/$\overline{TEND_0}$ pin is used for the $\overline{TEND_0}$ function. When CKA1D = 0, the pin is used as CKA1, an external data clock input/output for channel 1. CKA1D is cleared to 0 during reset.

MPBR/EFR: Multiprocessor Bit Receive/Error Flag Reset (Bit 3): When multiprocessor mode is enabled (MP in CNTLB = 1), MPBR, when read, contains the value of the MPB bit for the last receive operation. When 0 is written to MPBR/EFR, the EFR function is selected to reset all error flags (OVRN, FE, and PE) to 0. MPBR/EFR is undefined during reset.

MOD2, MOD1, MOD0: ASCI Data Format Mode 2, 1, 0 (Bits 2-0): MOD2, MOD1, MOD0 program the ASCI data format as follows:

MOD2
  = 0 → 7 bit data
  = 1 → 8 bit data
MOD1
  = 0 → No parity
  = 1 → Parity enabled
MOD0
  = 0 → 1 stop bit
  = 1 → 2 stop bits

The data formats available based on all combinations of MOD2, MOD1, and MOD0 are shown in table 11-1.

Table 11-1. Combination of Data Formats

MOD2	MOD1	MOD0	Data Format
0	0	0	Start + 7 bit data + 1 stop
0	0	1	Start + 7 bit data + 2 stop
0	1	0	Start + 7 bit data + parity + 1 stop
0	1	1	Start + 7 bit data + parity + 2 stop
1	0	0	Start + 8 bit data + 1 stop
1	0	1	Start + 8 bit data + 2 stop
1	1	0	Start + 8 bit data + parity + 1 stop
1	1	1	Start + 8 bit data + parity + 2 stop

11.2.7 ASCI Control Register B 0, 1 (CNTLB0, CNTLB1)

Each ASCI channel control register B (figure 11-4) configures multiprocessor mode, parity, and baud rate selection.

ASCI Control Register B 0 (CNTLB0: I/O Address = 02H)
ASCI Control Register B 1 (CNTLB1: I/O Address = 03H)

bit	7	6	5	4	3	2	1	0
	MPBT	MP	$\overline{\text{CTS}}$/PS	PEO	DR	SS2	SS1	SS0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Figure 11-4. ASCI Control Register B 0, 1

MPBT: Multiprocessor Bit Transmit (Bit 7): When multiprocessor communication format is selected (MP bit = 1), MPBT is used to specify the MPB data bit for transmission. If MPBT = 1, then MPB = 1 is transmitted. If MPBT = 0, then MPB = 0 is transmitted. MPBT state is undefined during and after reset.

MP: Multiprocessor Mode (Bit 6): When MP is set to 1, the data format is configured for multiprocessor mode based on the MOD2 (number of data bits) and MOD0 (number of stop bits) bits in CNTLA. The format is as follows:

Start bit + 7 or 8 data bits + MPB bit + 1 or 2 stop bits

Note that multiprocessor (MP = 1) format has no provision for parity. If MP = 0, the data format is based on MOD0, MOD1, and MOD2 and may include parity. The MP bit is cleared to 0 during reset.

$\overline{\text{CTS}}$/PS: Clear to Send/Prescale (Bit 5): When read, $\overline{\text{CTS}}$/PS reflects the state of the external $\overline{\text{CTS}}$ input. If the $\overline{\text{CTS}}$ input pin is high, $\overline{\text{CTS}}$/PS will be read as 1. Note that when the $\overline{\text{CTS}}$ input pin is high, the TDRE bit is inhibited (that is, held at 0). For channel 1, the $\overline{\text{CTS}_1}$ input is multiplexed with RXS pin (clocked serial receive data). Thus, $\overline{\text{CTS}}$/PS is only valid when read if the channel 1 CTS1E bit = 1 and the $\overline{\text{CTS}_1}$ input pin function is selected. The read data of $\overline{\text{CTS}}$/PS is not affected by reset.

When written, $\overline{\text{CTS}}$/PS specifies the baud rate generator prescale factor. If $\overline{\text{CTS}}$/PS is set to 1, the system clock (ϕ) is prescaled by 30 while if $\overline{\text{CTS}}$/PS is cleared to 0, the system clock is prescaled by 10. $\overline{\text{CTS}}$/PS is cleared to 0 during reset.

PEO: Parity Even/Odd (Bit 4): PEO selects even or odd parity. PEO does not affect the enabling/disabling of parity (MOD1 bit of CNTLA). If PEO is cleared to 0, even parity is selected. If PEO is set to 1, odd parity is selected. PEO is cleared to 0 during reset.

DR: Divide Ratio (Bit 3): DR specifies the divider used to obtain the baud rate from the data sampling clock. If DR is reset to 0, divide by 16 is used, while if DR is set to 1, divide by 64 is used. DR is cleared to 0 during reset.

SS2, SS1, SS0: Source/Speed Select 2, 1, 0 (Bits 2-0): SS2-SS0 specify the data clock source (internal or external) and baud rate prescale factor. SS2, SS1, SS0 are all set to 1 during reset. Table 11-2 shows the divide ratio corresponding to SS2, SS1, and SS0.

Table 11-2. Divide Ratio

SS2	SS1	SS0	Divide Ratio
0	0	0	÷1
0	0	1	÷2
0	1	0	÷4
0	1	1	÷8
1	0	0	÷16
1	0	1	÷32
1	1	0	÷64
1	1	1	External clock

The external ASCI channel 0 data clock pins are multiplexed with DMA control lines ($CKA_0/\overline{DREQ_0}$ and $CKA_1/\overline{TEND_0}$). During reset, these pins are initialized as ASCI data clock inputs. If SS2, SS1, and SS0 are reprogrammed (any other value than SS2, SS1, SS0 = 1) these pins become ASCI data clock outputs. However, if DMAC channel 0 is configured to perform memory to/from I/O (and memory-mapped I/O) transfers the $CKA_0/\overline{DREQ_0}$ pin revert to DMA control signals regardless of SS2, SS1, SS0 programming. Also, if the CKA1D bit in the CNTLA register is set to 1, then $CKA_1/\overline{TEND_0}$ reverts to the DMA control output function regardless of SS2, SS1, and SS0 programming.

Final data clock rates are based on $\overline{CTS}$/PS (prescale), DR, SS2, SS1, SS0, and the HD647180X system clock (ϕ) frequency as shown in table 11-3.

Table 11-3. Baud Rate List

Prescaler		Sampling Rate		Baud Rate				General Divide Ratio	Baud Rate (Example) (BPS)			CKA	
PS	Divide Ratio	DR	Rate	SS2	SS1	SS0	Divide Ratio		ϕ=6.144 MHz	ϕ=4.608 MHz	ϕ=3.072 MHz	I/O	Clock Frequency
0	$\phi \div 10$	0	16	0	0	0	÷1	$\phi \div 160$	38400		19200	O	$\phi \div 10$
				0	0	1	2	320	19200		9600		20
				0	1	0	4	640	9600		4800		40
				0	1	1	8	1280	4800		2400		80
				1	0	0	16	2560	2400		1200		160
				1	0	1	32	5120	1200		600		320
				1	1	0	64	10240	600		300		640
				1	1	1	–	fc÷16	–	–	–	I	fc
		1	64	0	0	0	÷1	$\phi \div 640$	9600		4800	O	$\phi \div 10$
				0	0	1	2	1280	4800		2400		20
				0	1	0	4	2560	2400		1200		40
				0	1	1	8	5120	1200		600		80
				1	0	0	16	10240	600		300		160
				1	0	1	32	20480	300		150		320
				1	1	0	64	40960	150		75		640
				1	1	1	–	fc÷64	–	–	–	I	fc
1	$\phi \div 30$	0	16	0	0	0	÷1	$\phi \div 480$		9600		O	$\phi \div 30$
				0	0	1	2	960		4800			60
				0	1	0	4	1920		2400			120
				0	1	1	8	3840		1200			240
				1	0	0	16	7680		600			480
				1	0	1	32	15360		300			960
				1	1	0	64	30720		150			1920
				1	1	1	–	fc÷16	–	–	–	I	fc
		1	64	0	0	0	÷1	$\phi \div 1920$		2400		O	$\phi \div 30$
				0	0	1	2	3840		1200			60
				0	1	0	4	7680		600			120
				0	1	1	8	15360		300			240
				1	0	0	16	30720		150			480
				1	0	1	32	61440		75			960
				1	1	0	64	122880		37.5			1920
				1	1	1	–	fc÷64	–	–	–	I	fc

11.3 Modem Control Signals

ASCI channel 0 has $\overline{CTS_0}$, $\overline{DCD_0}$, and $\overline{RTS_0}$ external modem control signals. ASCI channel 1 has a $\overline{CTS_1}$ modem control signal which is multiplexed with RXS pin (clocked serial receive data).

11.3.1 $\overline{CTS_0}$: Clear to Send 0 (Input)

The $\overline{CTS_0}$ input allows external control (start/stop) of ASCI channel 0 transmit operations. When $\overline{CTS_0}$ is high, channel 0 TDRE bit is held at 0 regardless of whether the TDR0 (transmit data register) is full or empty. When $\overline{CTS_0}$ is low, TDRE will reflect the state of TDR0. Note that the actual transmit operation is not disabled by $\overline{CTS_0}$ high, only TDRE is inhibited.

11.3.2 $\overline{DCD_0}$: Data Carrier Detect 0 (Input)

The $\overline{DCD_0}$ input allows external control (start/stop) of ASCI channel 0 receive operations. When $\overline{DCD_0}$ is high, channel 0 RDRF bit is held at 0 regardless of whether the RDR0 (receive data register) is full or empty. The error flags (PE, FE, and OVRN bits) are also held at 0. Even after the $\overline{DCD_0}$ input goes low, these bits will not resume normal operation until the status register (STAT0) is read. Note that this first read of STAT0, while enabling normal operation, will still indicate the $\overline{DCD_0}$ input is high ($\overline{DCD0}$ bit = 1) even though it has gone low. Thus, the STAT0 register should be read twice to insure that the $\overline{DCD0}$ bit is reset to 0.

11.3.3 $\overline{RTS_0}$: Request to Send 0 (Output)

$\overline{RTS_0}$ allows the ASCI to control (start/stop) another communication device's transmission (for example, by connection to that device's $\overline{CTS}$ input). $\overline{RTS_0}$ is essentially a 1-bit output port, having no side effects on other ASCI registers or flags.

11.3.4 $\overline{CTS_1}$: Clear to Send 1 (Input)

Channel 1 $\overline{CTS_1}$ input is multiplexed with the RXS pin (clocked serial receive data). The $\overline{CTS_1}$ function is selected when the CTS1E bit in STAT1 is set to 1. When enabled, the $\overline{CTS_1}$ operation is equivalent to $\overline{CTS_0}$.

Modem control signal timing is shown in figures 11-5 and 11-6.

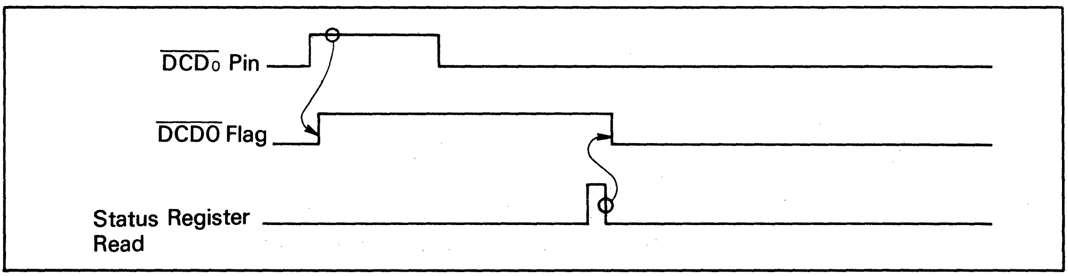


Figure 11-5. $\overline{DCD_0}$ Timing

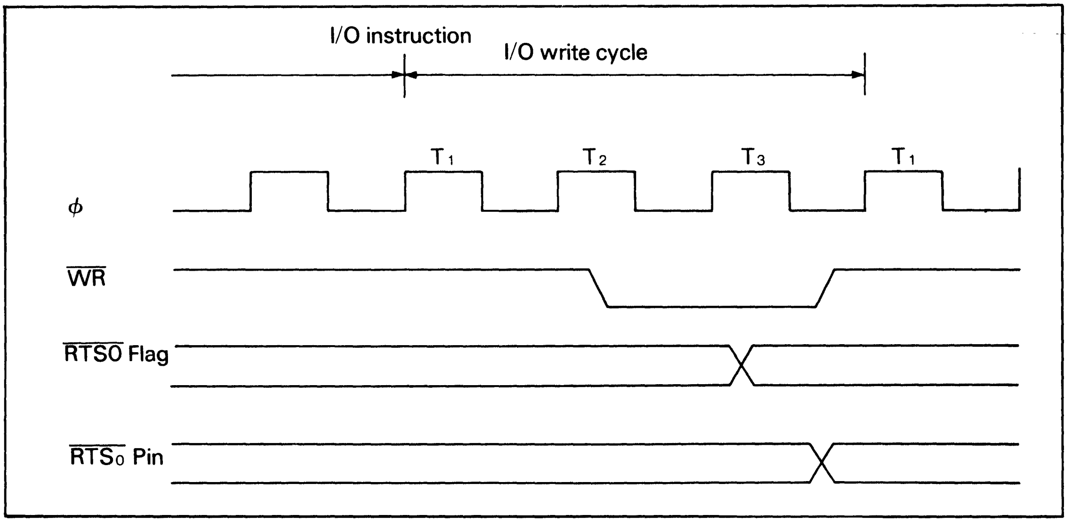


Figure 11-6. $\overline{\text{RTS}_0}$ Timing

11.4 ASCI Interrupts

Figure 11-7 shows the ASCI interrupt request generation circuit.

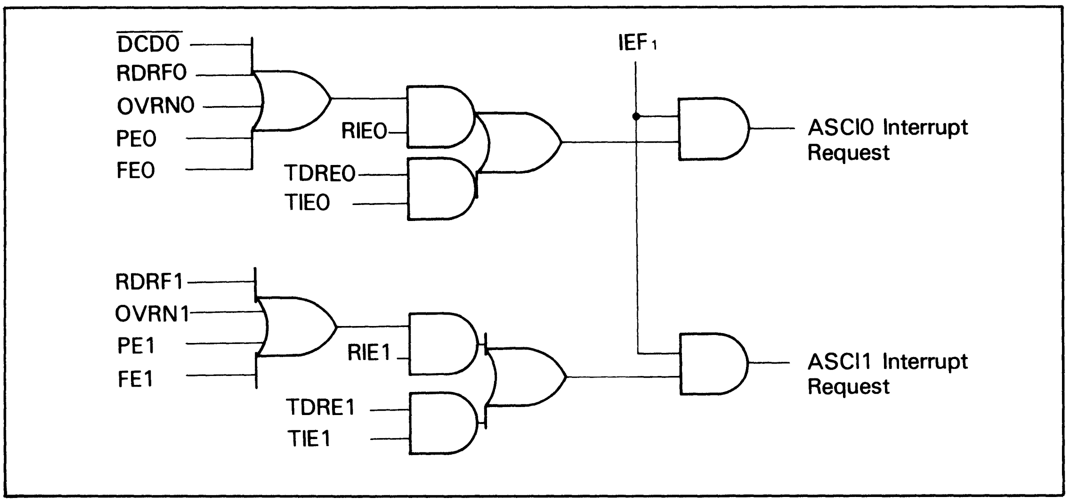


Figure 11-7. ASCI Interrupt Request Circuit Diagram

11.5 ASCI to/from DMAC Operation

Operation of the ASCI with the on-chip DMAC channel 0 requires the DMAC be correctly configured to utilize the ASCI flags as DMA request signals.

11.6 ASCI and Reset

During reset, the ASCI status and control registers are initialized as defined in the individual register descriptions.

Receive and transmit operations are stopped during reset. However, the contents of the transmit and receive data registers (TDR and RDR) are not changed by reset.

11.7 ASCI Clock

In external clock input mode, the external clock is directly input to the sampling rate (÷ 16/÷ 64) as shown in figure 11-8.

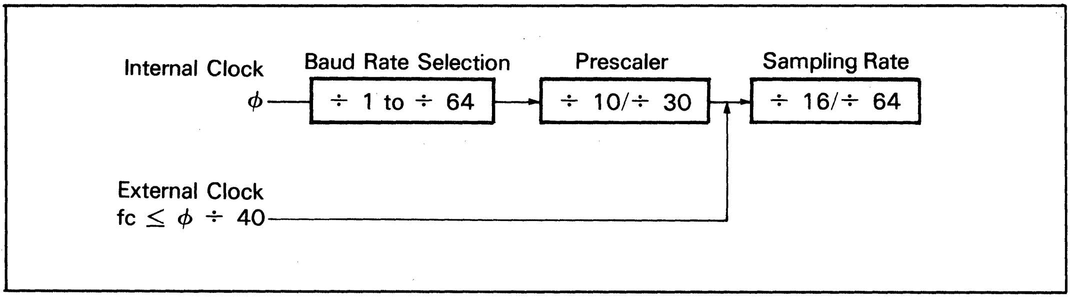


Figure 11-8. ASCI Clock Block Diagram

SECTION 12. CLOCKED SERIAL I/O PORT (CSI/O)

The HD647180X includes a simple, high-speed clock synchronous serial I/O port. The CSI/O includes transmit/receive (half-duplex), fixed 8-bit data, and internal or external data clock selection. High-speed operation (baud rate as high as 200 kbits/second at $f_C = 4$ MHz) is provided. The CSI/O is ideal for implementing a multiprocessor communication link between the HD647180X and the HMCS400 series (4-bit) and the HD6301 series (8-bit) single-chip controllers as well as additional HD647180X CPUs. These secondary devices may typically perform a portion of the system I/O processing such as keyboard scan/decode, LDC interface, etc.

12.1 CSI/O Block Diagram

The CSI/O (figure 12-1) consists of two registers—the transmit/receive data register (TRDR) and control register (CNTR).

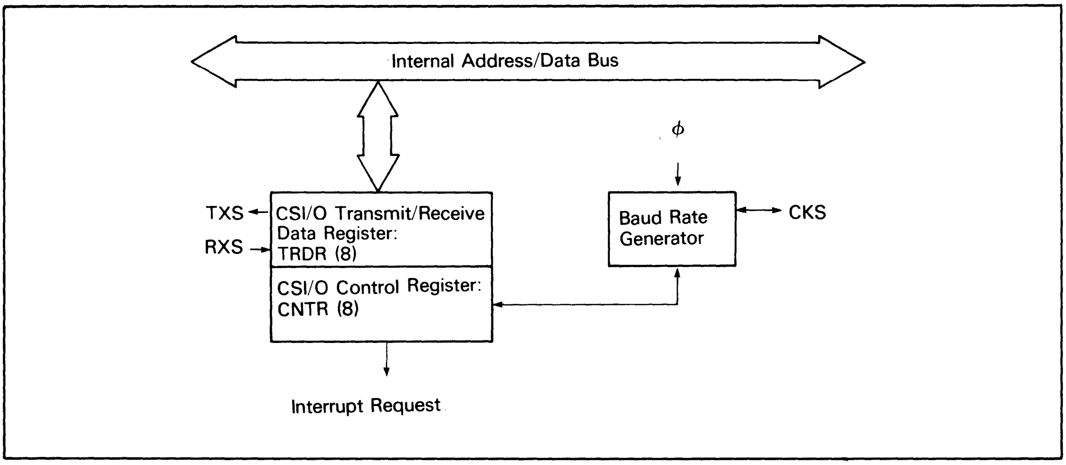


Figure 12-1. CSI/O Block Diagram

12.2 CSI/O Register Description

12.2.1 CSI/O Transmit/Receive Data Register (TRDR: I/O Address = 0BH)

TRDR is used for both CSI/O transmission and reception. Thus, the system design must insure that the constraints of half-duplex operation are met (transmit and receive operations can't occur simultaneously). For example, if a CSI/O transmission is attempted at the same time that the CSI/O is receiving data, the CSI/O will not work. Also note that TRDR is not buffered. Therefore, attempting to perform a CSI/O transmit while the previous transmit data is still being shifted out causes the shift data to be immediately updated, thereby corrupting the transmit operation in progress. Similarly, reading TRDR while a transmit or receive is in progress should be avoided.

12.2.2 CSI/O Control/Status Register (CNTR: I/O Address = 0AH)

CNTR (figure 12-2) monitors CSI/O status, enables and disables the CSI/O, enables and disables interrupt generation, and selects the data clock speed and source.

CSI/O Control Register (CNTR: I/O Address = 0AH)

bit	7	6	5	4	3	2	1	0
	EF	EIE	RE	TE	—	SS2	SS1	SS0
	R	R/W	R/W	R/W		R/W	R/W	R/W

Figure 12-2. CSI/O Control Register

EF: End Flag (Bit 7): EF is set to 1 by the CSI/O to indicate completion of an 8-bit data transmit or receive operation. If EIE (end interrupt enable) bit = 1 when EF is set to 1, a CPU interrupt request will be generated. Program access of TRDR should only occur if EF = 1. The CSI/O clears EF to 0 when TRDR is read or written. EF is cleared to 0 during reset and I/O stop mode.

EIE: End Interrupt Enable (Bit 6): EIE should be set to 1 to enable EF = 1 to generate a CPU interrupt request. The interrupt request is inhibited if EIE is reset to 0. EIE is cleared to 0 during reset.

RE: Receive Enable (Bit 5): A CSI/O receive operation is started by setting RE to 1. When RE is set to 1, the data clock is enabled. In internal clock mode, the data clock is output from the CKS pin. In external clock mode, the clock is input on the CKS pin. In either case, data is shifted in on the RXS pin in synchronization with the (internal or external) data clock. After receiving 8 bits of data, the CSI/O automatically clears RE to 0, sets EF to 1 and generates an interrupt (if enabled by EIE = 1). Note that RE and TE should never both be set to 1 at the same time. RE is cleared to 0 during reset and I/O stop mode.

Note that the RXS pin is multiplexed with the $\overline{CTS_1}$ modem control input of ASCI channel 1. In order to enable the RXS function, the CTS1E bit in CNTA1 should be reset to 0.

TE: Transmit Enable (Bit 4): A CSI/O transmit operation is started by setting TE to 1. When TE is set to 1, the data clock is enabled. In internal clock mode, the data clock is output from the CKS pin. In external clock mode, the clock is input on the CKS pin. In either case, data is shifted out on the TXS pin synchronous with the (internal or external) data clock. While transmitting the eighth bit of data, the CSI/O automatically clears TE to 0, sets EF to 1 and generates an interrupt (if enabled by EIE = 1). Note that TE and RE should never both be set to 1 at the same time. TE is cleared to 0 during reset and I/O stop mode.

SS2, SS1, SS0: Speed Select 2, 1, 0 (Bits 2-0): SS2, SS1, and SS0 select the CSI/O transmit/receive clock source and speed. SS2, SS1, and SS0 are all set to 1 during reset. Table 12-1 shows CSI/O baud rate selection.

Table 12-1. CSI/O Baud Rate Selection

SS2	SS1	SS0	Divide Ratio	Baud Rate (Note)
0	0	0	÷20	200000
0	0	1	÷40	100000
0	1	0	÷80	50000
0	1	1	÷160	25000
1	0	0	÷320	12500
1	0	1	÷640	6250
1	1	0	÷1280	3125
1	1	1	External clock input (less than ÷20)	

Note: ϕ = 4 MHz.

After reset, the CKS pin is configured as an external clock input (SS2, SS1, SS0 = 1). Changing these values causes CKS to become an output pin and the selected clock will be output when transmit or receive operations are enabled.

12.3 CSI/O Interrupts

The CSI/O interrupt request circuit is shown in figure 12-3.

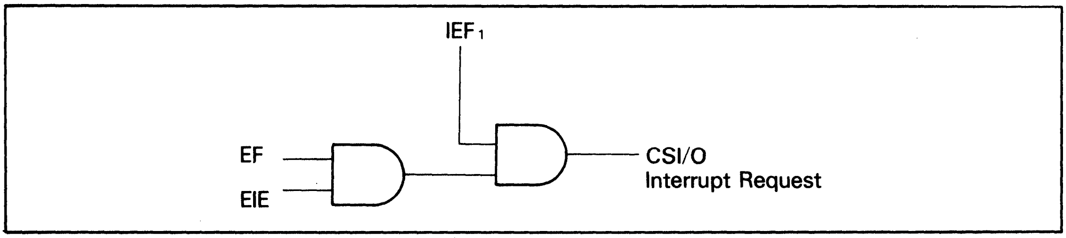


Figure 12-3. CSI/O Interrupt Circuit Diagram

12.4 CSI/O Operation

The CSI/O can be operated using status polling or interrupt driven algorithms.

12.4.1 Transmit—Polling

1. Poll the TE bit in CNTR until TE = 0.
2. Write the transmit data into TRDR.

3. Set the TE bit in CNTR to 1.
4. Repeat 1 to 3 for each transmit data byte.

12.4.2 Transmit—Interrupts

1. Poll the TE bit in CNTR until TE = 0.
2. Write the first transmit data byte into TRDR.
3. Set the TE and EIE bits in CNTR to 1.
4. When the transmit interrupt occurs, write the next transmit data byte into TRDR.
5. Set the TE bit in CNTR to 1.
6. Repeat 4 to 5 for each transmit data byte.

12.4.3 Receive—Polling

1. Poll the RE bit in CNTR until RE = 0.
2. Set the RE bit in CNTR to 1.
3. Poll the RE bit in CNTR until RE = 0.
4. Read the receive data from TRDR.
5. Repeat 2 to 4 for each receive data byte.

12.4.4 Receive—Interrupts

1. Poll the RE bit in CNTR until RE = 0.
2. Set the RE and EIE bits in CNTR to 1.
3. When the receive interrupt occurs read the receive data from TRDR.
4. Set the RE bit in CNTR to 1.
5. Repeat 3 to 4 for each receive data byte.

12.5 CSI/O Operation Timing Notes

Note that transmitter clocking and receiver sampling timings are different from internal and external clocking modes. Figure 12-4 to 12-7 shows CSI/O transmit/receive timing.

The transmitter and receiver should be disabled (TE and RE = 0) when initializing or changing the baud rate.

12.6 CSI/O Operation Notes

Disable the transmitter and receiver (TE and RE = 0) before initializing or changing the baud rate. When changing the baud rate after completing transmission or reception, a delay of at least one bit time is required before baud rate modification.

When RE or TE is cleared to 0 by software, the corresponding receive or transmit operation is immediately terminated. Normally, TE or RE should only be cleared to

0 when EF = 1.

Simultaneous transmission and reception is not possible. Thus, TE and RE should not both be 1 at the same time.

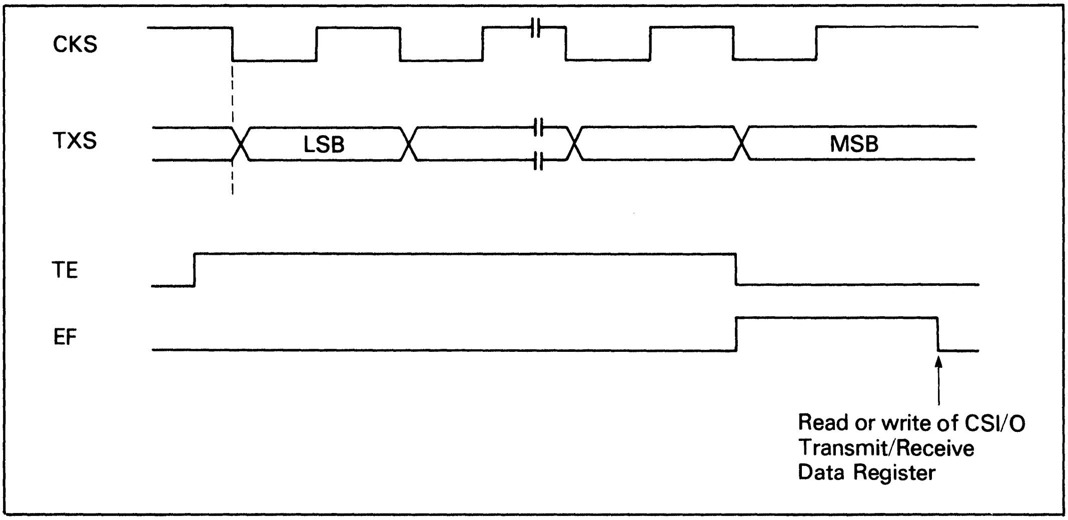


Figure 12-4. Transmit Timing — Internal Clock

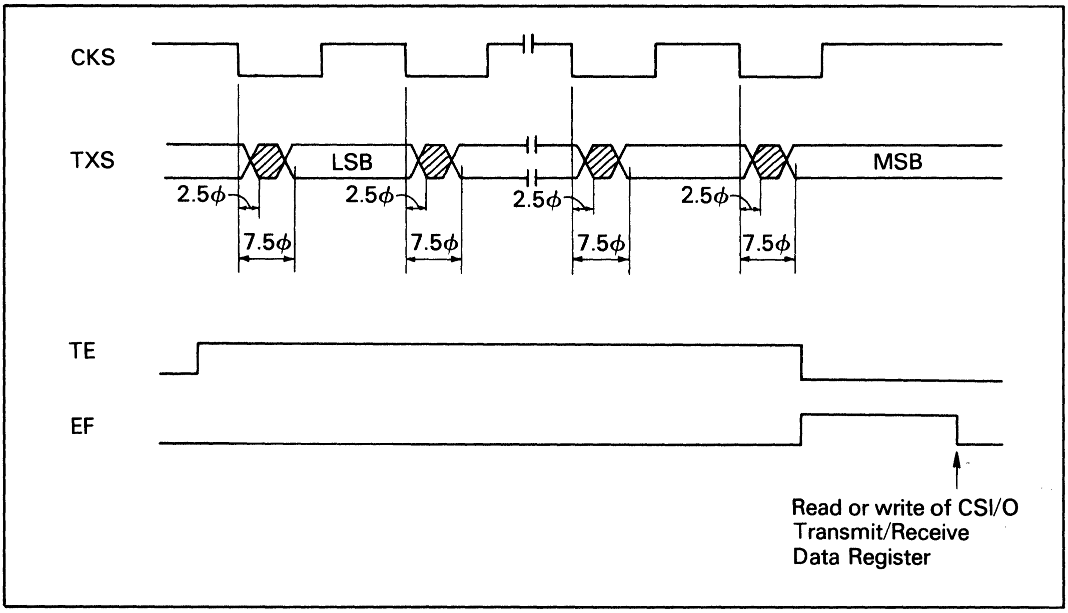


Figure 12-5. Transmit Timing — External Clock

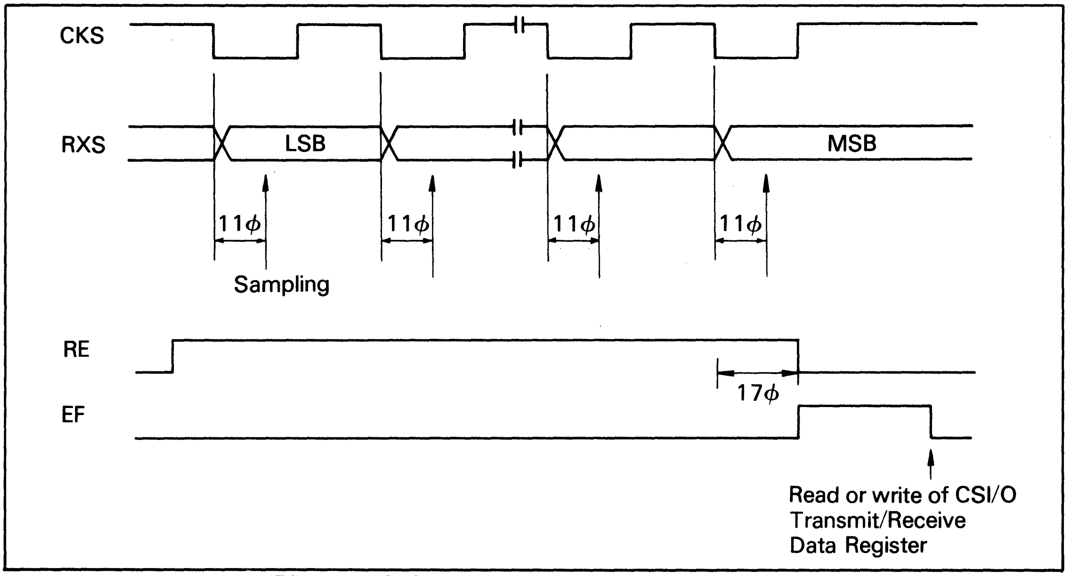


Figure 12-6. Receive Timing—Internal Clock

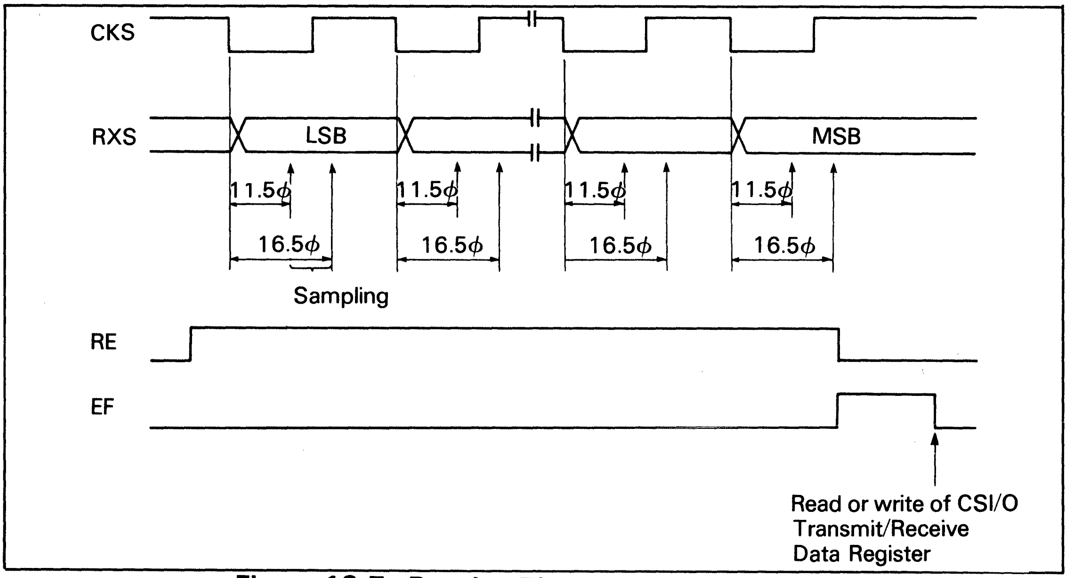


Figure 12-7. Receive Timing—External Clock

12.7 CSI/O and Reset

During reset each bit in the CNTR is initialized as defined in the CNTR register description.

CSI/O transmit and receive operations in progress are aborted during reset. However, the contents of TRDR are not changed.

SECTION 13. PROGRAMMABLE RELOAD TIMER (PRT)

The HD647180X contains a two-channel 16-bit programmable reload timer. Each PRT channel contains a 16-bit down counter and a 16-bit reload register. The down counter can be directly read and written and a down counter overflow interrupt can be programmably enabled or disabled. In addition, PRT channel 1 has a TOUT output pin which can be set high, low or toggled. Thus PRT1 can perform programmable output waveform generation.

13.1 PRT Block Diagram

The PRT block diagram is shown in figure 13-1. The two channels have separate timer data and reload registers and a common status/control register. The PRT input clock for both channels is equal to the system clock (ϕ) divided by 20.

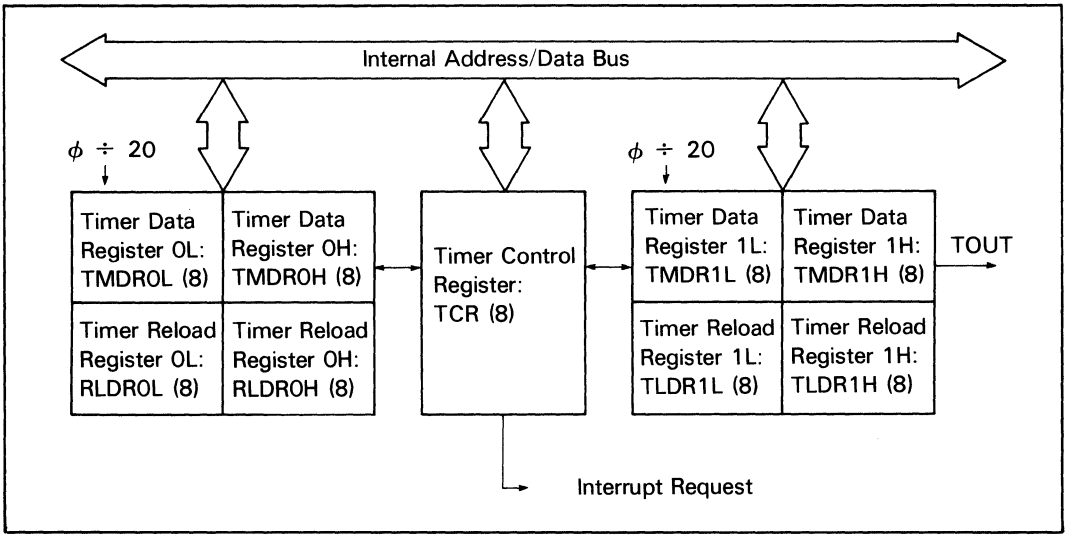


Figure 13-1. PRT Block Diagram

13.2 PRT Register Description

13.2.1 Timer Data Register (TMDR: I/O Address = CH0: 0DH, 0CH/CH1: 15H, 14H)

PRT0 and PRT1 each have 16-bit timer data registers (TMDR). TMDR0 and TMDR1 are each accessed as low- and high-byte registers (TMDR0H, TMDR0L and TMDR1H, TMDR1L). During reset, TMDR0 and TMDR1 are set to FFFFH.

TMDR is decremented once every twenty ϕ clocks. When TMDR counts down to 0, it is automatically reloaded with the value contained in the reload register (RLDR).

TMDR can be read and written by software using the following procedures. The read procedure uses a PRT internal temporary storage register to return accurate data without requiring the timer to be stopped. The write procedure requires the PRT to be stopped.

For reading (without stopping the timer), TMDR must be read in the order of lower byte—higher byte (TMDRnL, TMDRnH). The lower byte read (TMDRnL) will store the higher byte value in an internal register. The following higher byte read (TMDRnH) will access this internal register. This procedure insures timer data validity by eliminating the problem of the 16-bit timer updating between each 8-bit read. Specifically, reading TMDR in higher byte—lower byte order may result in invalid data. Note the implications of TMDR higher byte internal storage for applications which may read only the lower and/or higher bytes. In normal operation all TMDR read routines should access both the lower and higher bytes, in that order.

For writing, TMDR down-counting must be inhibited using the TDE (timer down count enable) bits in the TCR (timer control register), following which either or both higher and lower bytes of TMDR can be freely written (and read) in any order.

13.2.2 Timer Reload Register (RLDR: I/O Address = CH0: 0EH, 0FH/CH1: 16H, 17H)

PRT0 and PRT1 each have 16-bit timer reload registers (RLDR). RLDR0 and RLDR1 are each accessed as low- and high-byte registers (RLDR0H, RLDR0L and RLDR1H, RLDR1L). During reset RLDR0 and RLDR1 are set to FFFFH.

When the TMDR counts down to 0, it is automatically reloaded with the contents of RLDR.

13.2.4 Timer Control Register (TCR)

TCR (figure 13-2) monitors both channels' (PRT0, PRT1) TMDR status and controls enabling and disabling of down counting and interrupts as well as controlling the output pin (TOUT1) for PRT 1.

Timer Control Register (TCR: I/O Address = 10H)

bit	7	6	5	4	3	2	1	0
	TIF1	TIF0	TIE1	TIE0	TOC1	TOC0	TDE1	TDE0
	R	R	R/W	R/W	R/W	R/W	R/W	R/W

Figure 13-2. Timer Control Register

TIF1: Timer Interrupt Flag 1 (Bit 7): When TMDR1 decrements to 0, TIF1 is set to 1. This can generate an interrupt request if enabled by TIE1 = 1. TIF1 is reset to 0 when TCR is read and the higher or lower byte of TMDR1 are read. During reset, TIF1 is cleared to 0.

TIF0: Timer Interrupt Flag 0 (Bit 6): When TMDR0 decrements to 0, TIF0 is set to 1. This can generate an interrupt request if enabled by TIE0 = 1. TIF0 is reset to 0 when TCR is read and the higher or lower byte of TMDR0 is read. During reset, TIF0 is cleared to 0.

TIE1: Timer Interrupt Enable 1 (Bit 5): When TIE1 is set to 1, TIF1 = 1 will generate a CPU interrupt request. When TIE1 is reset to 0, the interrupt request is inhibited. During reset, TIE1 is cleared to 0.

TIE0: Timer Interrupt Enable 0 (Bit 4): When TIE0 is set = 1, TIF0 to 1 will generate a CPU interrupt request. When TIE0 is reset to 0, the interrupt request is inhibited. During reset, TIE0 is cleared to 0.

TOC1, TOC0: Timer Output Control (Bits 3, 2): TOC1 and TOC0 control the output of PRT1 using the TOUT1 pin as shown in table 13-1. During reset, TOC1 and TOC0 are cleared to 0. This sets TOUT1 to 1. By programming TOC1 and TOC0, the TOUT1 pin can be forced high, low or toggled when TMDR1 decrements to 0.

Table 13-1. Timer Output

TOC1	TOC0	TOUT1
0	0	1
0	1	Toggled (Note)
1	0	0
1	1	1

Note: When TMDR1 decrements to 0, TOUT1 level is reversed. This produces a square wave with 50% duty cycle at the output without any software support.

TDE1, TDE0: Timer Down Count Enable (Bits 1, 0): TDE1 and TDE0 enable and disable down-counting for TMDR1 and TMDR0 respectively. When TDEn (n = 0, 1) is set to 1, TMDRn counts down. When TDEn is reset to 0, down-counting is stopped and TMDRn can be freely read or written. TDE1 and TDE0 are cleared to 0 during reset and TMDRn will not decrement until TDEn is set to 1.

Figure 13-3 shows timer initialization, count down, and reload timing. Figure 13-4 shows timer output (TOUT1) timing.

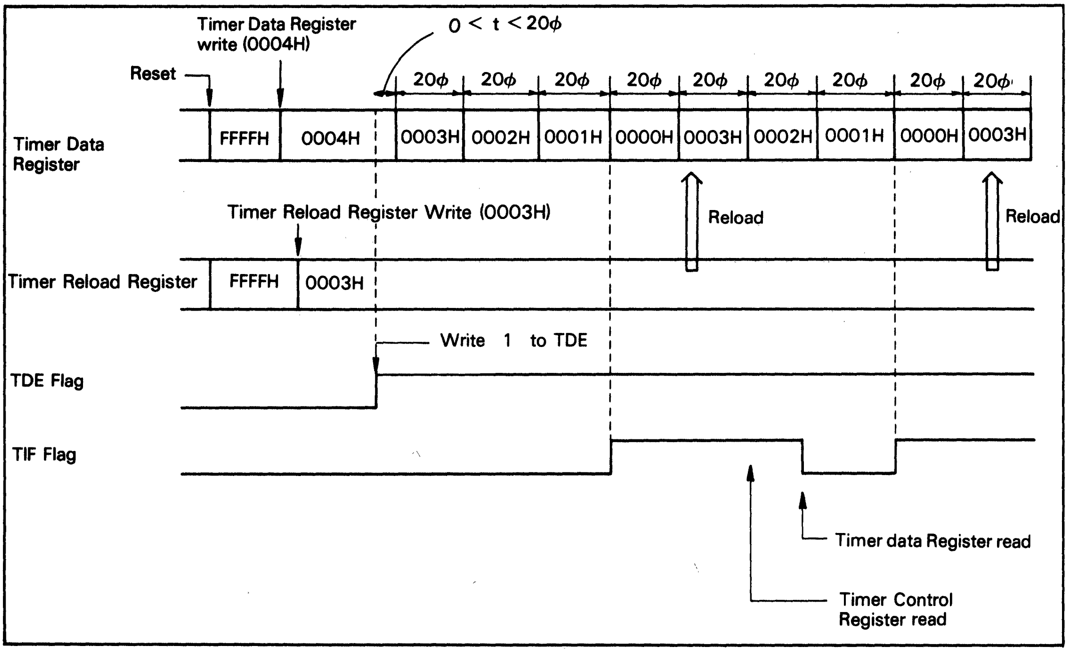


Figure 13-3. PRT Operation Timing

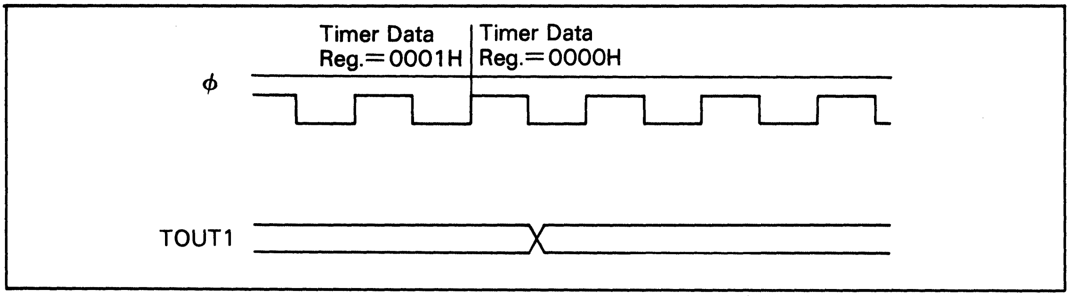


Figure 13-4. PRT Output Timing

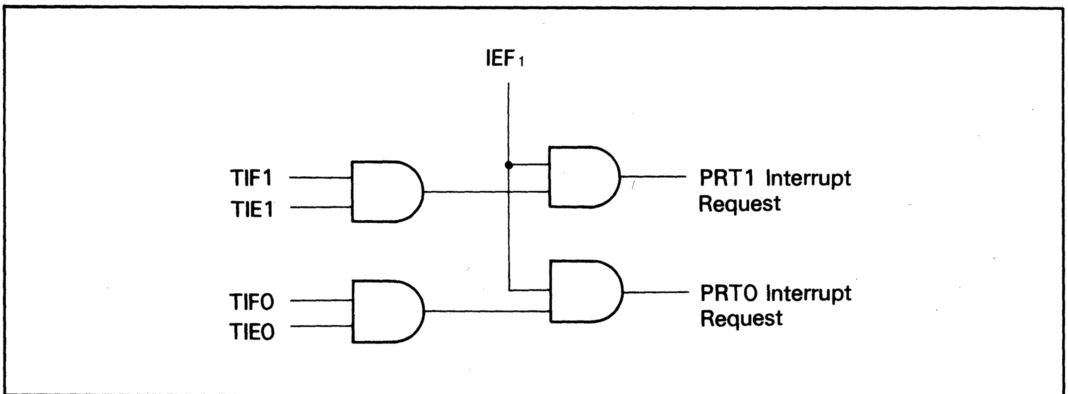


Figure 13-5. PRT Interrupt Request Circuit Diagram

13.3 PRT Interrupts

The PRT interrupt request circuit is shown in figure 13-5.

13.4 PRT and Reset

During reset the bits in TCR are initialized as defined in the TCR register description. Down-counting is stopped and the TMDR and RLDR registers are initialized to FFFFH.

13.5 PRT Operation Notes

TMDR data can be accurately read without stopping down-counting by reading the lower (TMDRnL) and higher (TMDRnH) bytes in that order (n=0, 1). Or TMDR can be freely read or written by stopping the down-counting.

Care should be taken to insure that a timer reload does not occur during or between lower (RLDRnL) and higher (RLDRnH) byte writes (n=0, 1). This may be guaranteed by system timing design or by stopping down-counting (with TMDR containing a non-zero value) when updating RLDR.

Similarly, in applications in which TMDR is written at each TMDR overflow, the system/software design should guarantee that RLDR can be updated before the next overflow occurs. Otherwise, the time base will be inaccurate.

By reprogramming the TOC1 and TOC0 bits, the timer output function for PRT channel 1 can be selected. The following shows the initial state of the TOUT1 pin after TOC1 and TOC0 are programmed to select the PRT channel 1 timer output function.

- PRT (channel 1) has not counted down to 0.

If the PRT has not counted down to 0 (timed out), the initial state of TOUT1 depends on the programmed value in TOC1 and TOC0. (table 13-2).

Table 13-2. Timer Output If PRT Has Not Timed Out

TOC1	TOC0	TOUT1 State After Programming TOC1/TOC0	TOUT1 State After Next Timeout
0	1	High (1)	Low (0)
1	0	High (1)	Low (0)
1	1	High (1)	High (1)

· PRT (channel 1) has counted down to 0 at least once.

If the PRT has counted down to 0 (timed out) at least once, the initial state of TOUT1 depends on the number of time outs (even or odd) that have occurred (table 13-3).

Table 13-3. Timer Output When PRT Has Timed Out

Numbers of Timeouts	TOUT1 State After Programming TOC1/TOC0
Even (2, 4, 6 ...)	High (1)
Odd (1, 3, 5 ...)	Low (0)

SECTION 14. PROGRAMMABLE TIMER 2 (PT2)

The HD647180X provides a one-channel 16-bit programmable timer 2 (PT2).

The PT2 can perform input waveform measurement and generate two independent output waveforms at the same time.

The 16-bit free-running counter counts up synchronously with the system clock (ϕ) divided by 4 and a timer 2 overflow interrupt occurs when the count value returns from FFFFH to 0000H.

14.1 PT2 Block Diagram

The PT2 block diagram is shown in figure 14-1.

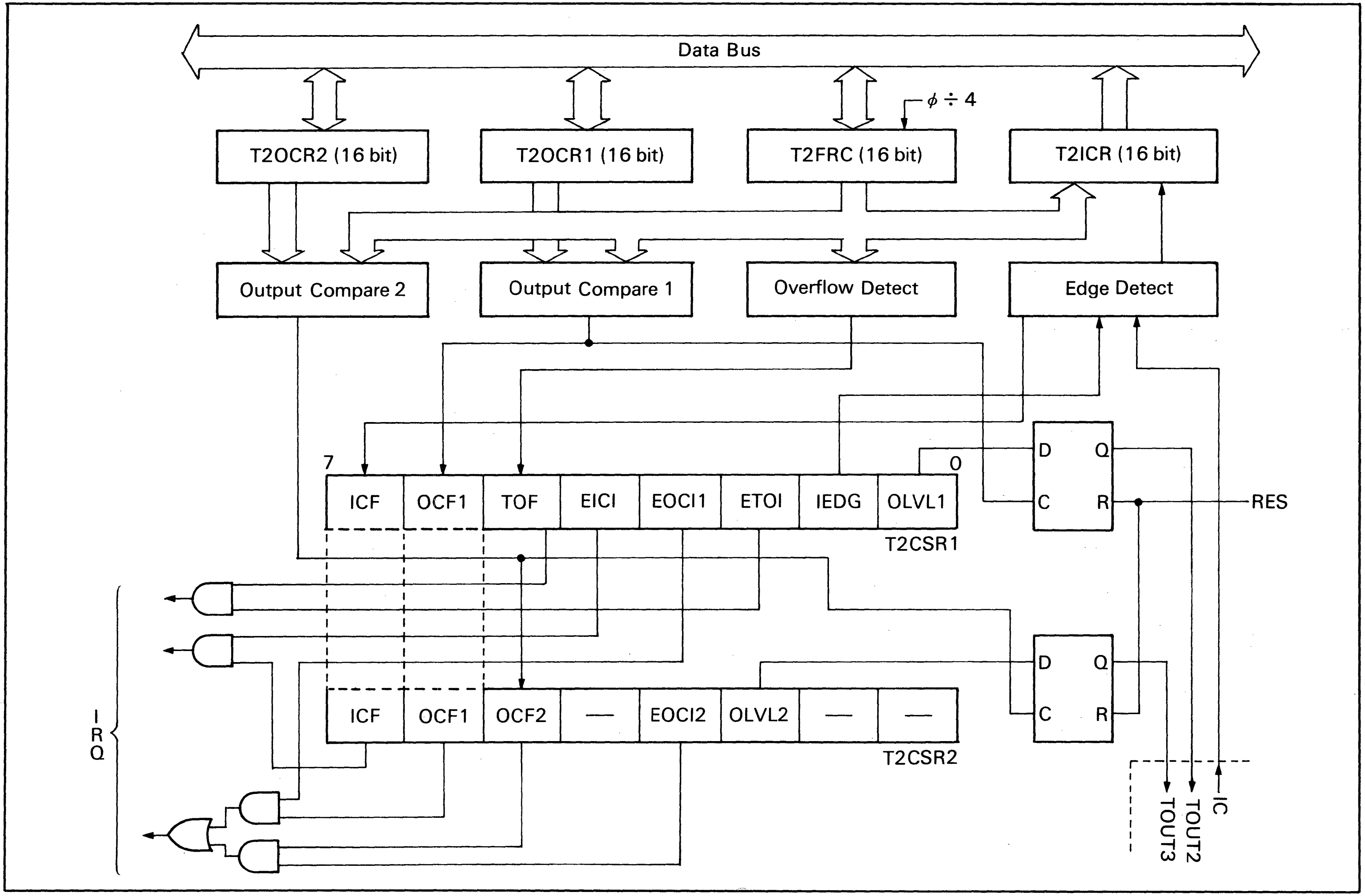


Figure 14-1. PT2 Block Diagram

14.2 PT2 Registers

Table 14-1 shows the PT2 register.

Table 14-1. PT2 Registers

Name	R/W
Timer 2 Control/Status Register 1	R/W
Timer 2 Control/Status Register 2	R/W
Timer 2 Free-Running Counter H	R/W
Timer 2 Free-Running Counter L	R/W
Output Compare Register 1H	R/W
Output Compare Register 1L	R/W
Output Compare Register 2H	R/W
Output Compare Register 2L	R/W
Input Capture Register H	R
Input Capture Register L	R

14.2.1 Timer 2 Free-Running Counter (T2FRC: I/O Address = 40H, 41H)

The T2FRC 16-bit free-running counter incremented every four ϕ clocks. This register can be read by software and cleared at reset.

14.2.2 Output Compare Register (T2OCR1: I/O Address = 42H, 43H)
(T2OCR2: I/O Address = 44H, 45H)

The T2OCR1 and T2OCR2 16-bit read/write registers control an output waveform. These registers are always compared with T2FRC. If a match occurs, the output compare flag 1 (OCF1) bit in the timer 2 control/status registers is set.

14.2.3 Input Capture Register (T2ICR: I/O Address = 46H, 47H)

T2ICR is a 16-bit read-only register. The contents of T2FRC is transferred to this register on the positive or negative edge of the IC input signal. The IEDG bit in T2CSR1 determines which edge triggers the input capture. ICR is cleared at reset.

14.2.4 Timer 2 Control/Status Register 1 (T2CSR1: I/O Address = 48H)

The T2CSR1 (figure 14-2) 8-bit register controls timer 2 and holds the timer 2 status information. All bits can be read and the lowest 5 bits can also be written. T2CSR1 is cleared at reset.

bit	7	6	5	4	3	2	1	0
	ICF	OCF1	TOF	EICI	EOCI1	ETOI	IEDG	OLVL1
	R	R	R	R/W	R/W	R/W	R/W	R/W

Timer 2 Control/Status Register 1 (T2CSR1)

Figure 14-2. Timer 2 Control/Status Register 1

OLVL1: Output Level 1 (Bit 0): When the timer 2 free-running counter (T2FRC) contents agree with those of output compare register 1 (T2OCR1), the OLV1 value appears at TOUT2.

IEDG: Input Edge (Bit 1): IEDG determine which edge of the IC input signal triggers data transfer from T2FRC to T2ICR.

· IEDG = 1 selects the negative edge
· IEDG = 0 selects the positive edge

ETOI: Enable Timer 2 Overflow Interrupt (Bit 2): If ETOI is set, timer 2 overflow interrupt (TOI) is enabled; if cleared, the interrupt is disabled.

EOCI1: Enable Output Compare Interrupt 1 (Bit 3): If EOCI1 is set, timer 2 output compare 1 interrupt (OCI1) is enabled. If cleared, the interrupt is disabled.

EICI: Enable Input Capture Interrupt (Bit 4): If EICI is set, timer 2 input capture interrupt (ICI) is enabled. If cleared, the interrupt is disabled.

TOF: Timer 2 Overflow Flag (Bit 5): TOF is set when T2FRC is incremented by 1 from $FFFF to $0000. It is cleared:

1. at reset, or
2. when the CPU reads T2CSR1 then reads the high-order byte of T2FRC.

OCF1: Output Compare Flag 1 (Bit 6): OCF1 is set when T2OCR1 and T2FRC agree. It is cleared:

1. at reset, or
2. when the CPU reads either T2CSR1 or T2CSR2, then writes T2OCR1.

ICF: Input Capture Flag (Bit 7): ICF is set when the contents of T2FRC are transferred to T2ICR. It is cleared:

1. at reset, or
2. when the CPU reads T2CSR1 or T2CSR2, then the T2ICR high-order byte, and finally the T2ICR low-order byte.

14.2.5 Timer 2 Control/Status Register 2 (T2CSR2: I/O Address = 49H)

The T2CSR2 (figure 14-3) 4-bit register preserves status information. Bit 3 is a read/write bit and the high-order 3 bits are read-only bits.

All bits except for bits 0, 1, 2, and 4 are cleared to 0 at reset. (Since bits 0, 1, and 4 are not defined, they are always regarded as 1's.)

Timer 2 Control/Status Register 2 (T2CSR2)

bit	7	6	5	4	3	2	1	0
	ICF	OCF1	OCF2	—	EOCI2	OLVL2	—	—
	R	R	R		R/W	R/W		

Figure 14-3. Timer 2 Control/Status Register 2

OLVL2: Output level 2 (Bit 2): When the timer 2 free-running counter (T2FRC) contents agree with those of output compare register 2 (T2OCR2), this bit value appears at TOUT3.

EOCI2: Enable Output Compare Interrupt 2 (Bit 3): If EOCI2 is set, timer 2 output compare interrupt 2 (T2OCI2) is enabled. If cleared, the interrupt is disabled.

OCF2: Output Compare Flag 2 (Bit 5): OCF2 is set when T2OCR2 and T2FRC agree. It is cleared:

1. at reset, or
2. when the CPU writes T2OCR2 after reading T2CSR2.

OCF1: Output Compare Flag 1 (Bit 6), ICF: Input Capture Flag (Bit 7) OCF1 and ICF are the same as OCF1 and ICF in timer 2 control/status register 1 (T2CSR1). The same OCF1 and ICF bit information can be obtained by reading either T2CSR1 or T2CSR2.

14.3 Precautions in Using PT2

When the HD647180X is released from the I/O stop mode, the timer 2 control/status registers, the timer 2 free-running counter (T2FRC), the output compare registers (T2OCR1, T2OCR2), and the input capture register (T2ICR) are placed in the initial state.

The CPU must write to the output compare register high-order byte first, then low-order byte. The output compare function is disabled during a write operation to this register.

The CPU must read the input capture register high-order byte first, then low-order byte. The input capture function is disabled during a read operation from this register.

The CPU must write to the timer 2 free-running counter high-order byte first and then the low-order byte. The timer 2 overflow, input capture signal input, and output compare are ignored during a write operation to this register.

Timer 2 returns from I/O stop mode 5 clocks after the CPU has returned from I/O stop mode.

Pulse width input to the IC pin must be at least 5 clocks. Otherwise timer 2 may not receive the input-capture input correctly.

SECTION 15. ANALOG COMPARATOR

The HD647180X provides an analog comparator with 6 channels. Each channel can be programmed as a reference voltage (V_{ref}) input pin or a compared voltage (V_{in}) input pin. In addition, these channels are multiplexed with port G and can also be used as TTL level input pins. (See 16-7 Port G)

Figure 15-1 shows the block diagram of this analog comparator.

The analog comparator is activated by reading or writing the comparator control/ status register (CCSR). In addition, since the comparator output affects the result bit of CCSR, the CPU is informed of the comparison result by reading this bit.

Bits 0 to 5 in CCSR are used to select two channels used as V_{ref} and V_{in} inputs as shown in tables 15-1 and 15-2. At reset, ch4 and ch5 are set to the V_{ref} channel and V_{in} channel, respectively.

The same channel cannot function as both V_{in} and V_{ref} at the same time. If the same channel is selected, the result bit value cannot be guaranteed.

When these channels are used for analog comparators, they do not function as port G. Note that full current will flow if the CPU reads the port G input data while these channels are used for analog comparator.

Table 15-1.
Compared Voltage Channel Selection (V_{in})

AIN2	AIN1	AIN0	Channel
0	0	0	Ch 0
0	0	1	Ch 1
0	1	0	Ch 2
0	1	1	Ch 3
1	0	0	Ch 4
1	0	1	Ch 5

Table 15-2.
Reference Voltage Channel Selection (V_{ref})

REF2	REF1	REF0	Channel
0	0	0	Ch 0
0	0	1	Ch 1
0	1	0	Ch 2
0	1	1	Ch 3
1	0	0	Ch 4
1	0	1	Ch 5

15.1 Analog Comparator Block Diagram

The analog comparator is show in figure 15-1.

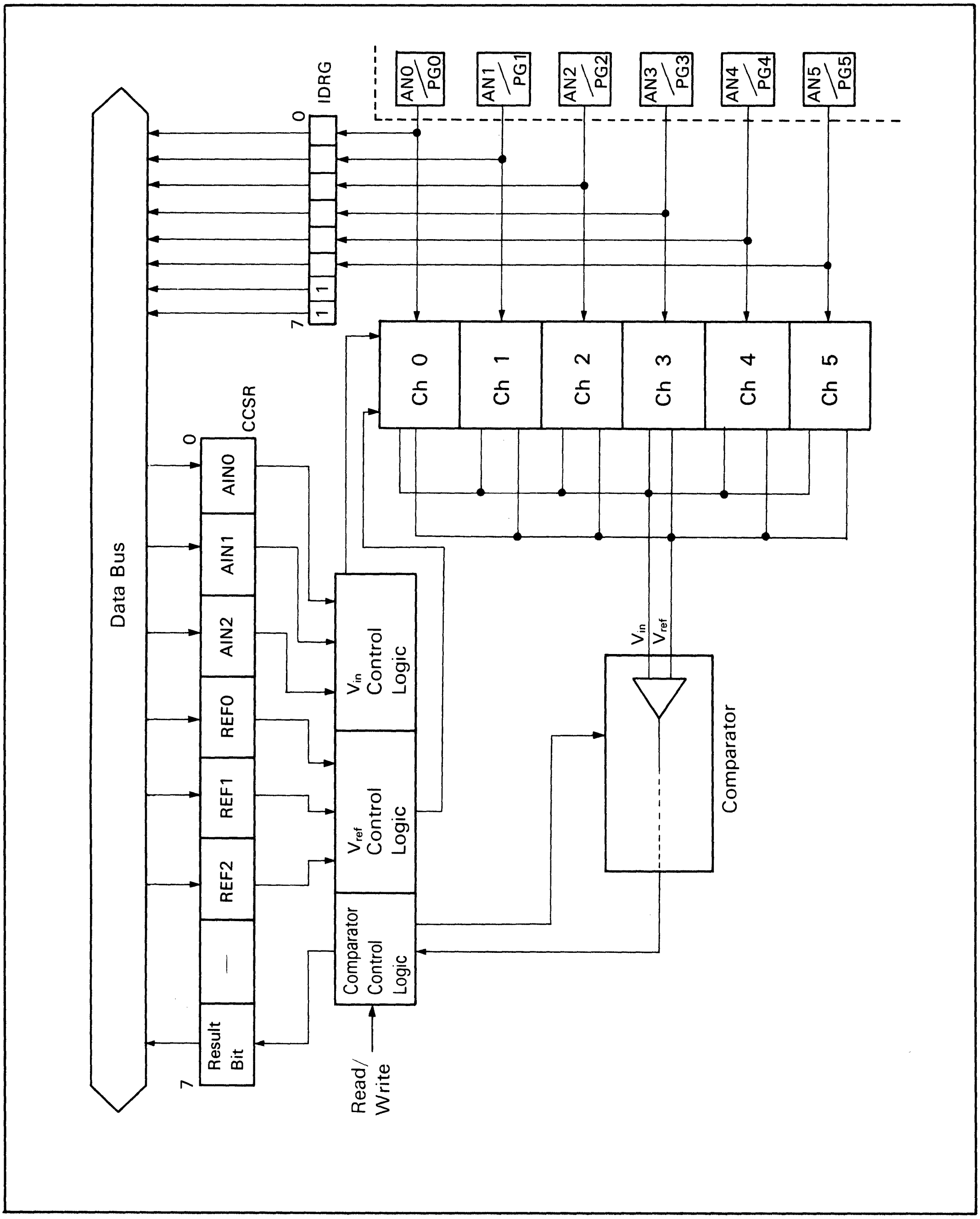


Figure 15-1. Analog Comparator Block Diagram

15.2 Comparator Control/Status Register (CCSR: I/O Address = 50H)

15.2.1 RBIT: Result Bit (Bit 7)

The read-only RBIT stores the result of the comparison between V_{ref} and V_{in}. If V_{ref} is greater than V_{in}, this bit is 0; if V_{ref} is smaller than V_{in}, this bit is 1. However, if V_{ref} is equal to V_{in}, the bit value cannot be guaranteed.

15.2.2 REF2-REF0: Reference Channel Bits (Bits 5 to 3)

The REF2-REF0 read/write bits selects the channel to be used as the reference voltage (V_{ref}) channel.

15.2.3 AIN2-AIN0: Analog Channel Bits (Bits 2 to 0)

The AIN2-AIN0 read/write bits select the channel to be used as the compared voltage (V_{in}) channel.

15.3 Precautions in Using the Analog Comparator

It takes three system clock cycles after comparison starts until the result affects the result bit, independent of the operating frequency (see figure 15-2). Accordingly, the CPU must read the result bit 13 clock cycles or more after activating the comparator.

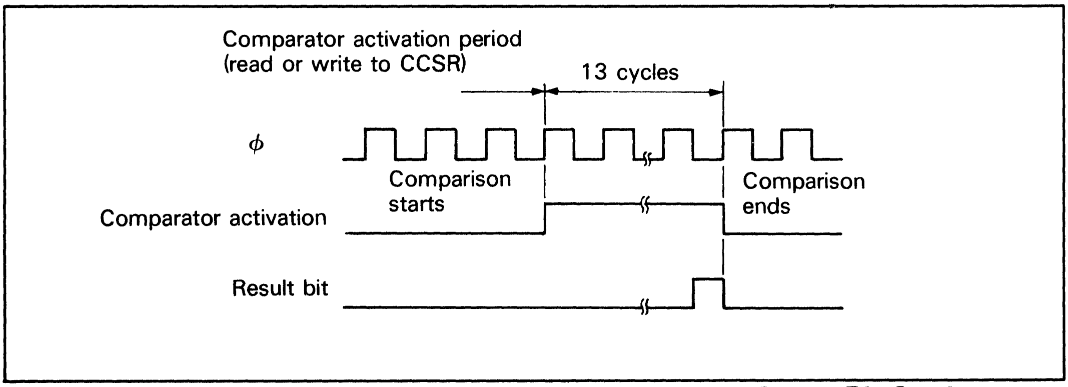


Figure 15-2. Timing for Comparator Activation and Result Bit Setting

If the compared voltage (V_{in}) enters an undefined area or crosses the reference voltage (V_{ref}) during comparison, the result bit value cannot be guaranteed (figure 15-3).

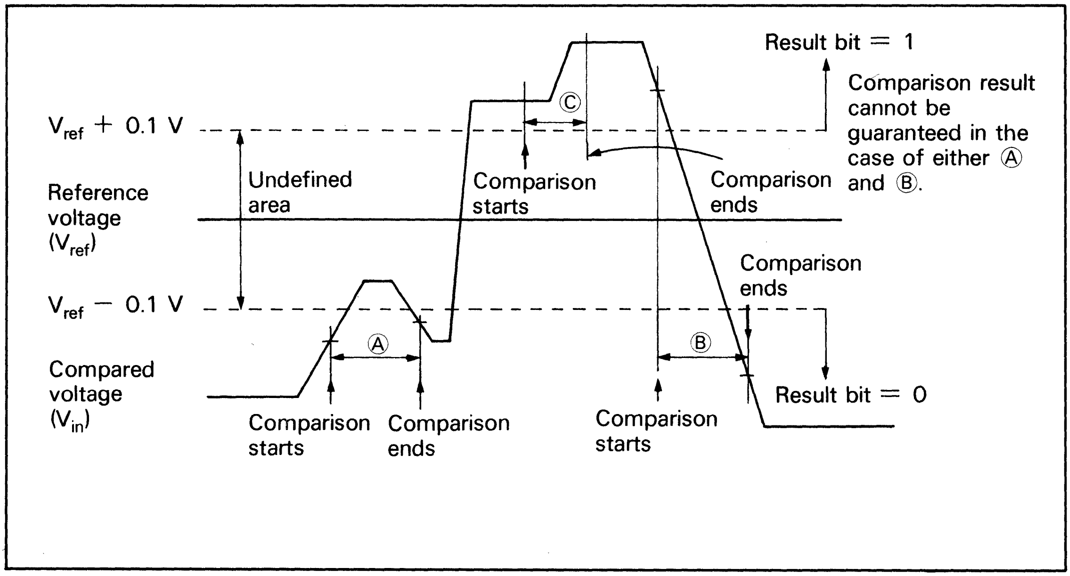


Figure 15-3. Correspondence between Reference Voltage (V_{ref}) and Compared Voltage (V_{in})

SECTION 16. I/O PORTS

The HD647180X provides seven I/O ports. Ports A, B, C, D, E, and F are 8-bit I/O ports. Port G is a 6-bit input-only port.

In the expanded modes (modes 1 and 2), port F is configured as the data bus, port C, port D, port E0 to port E3 pins are used as the address bus and port B and port E4 to port E7 pins function as control signal inputs or outputs.

Each port consists of a data direction register (DDR) to determine the directions of the individual pins, an output data register (ODR) to hold output data and an input data register (IDR) to latch input data.

At reset, DDRs are initialized to 0, configuring all ports as input ports. Accordingly, in order to configure a port as an output after reset, the CPU must set the corresponding DDR bit to 1. Each bit in the DDR is programmable as an input or output mode. However port G does not have DDR since it is an input-only port.

ODRs hold data to be output via the ports. (Port G does not have an ODR.) These registers are not initialized at reset. Accordingly, the CPU must write data to be output in ODRs before programming the corresponding DDR bits to the output mode.

If DDR bits are configured in the output mode before data is written to the ODR, output data at the corresponding port cannot be guaranteed.

When a port is in the output mode, the CPU can know the contents of the corresponding ODR by reading the port. However, ODR bits corresponding to port D0 to port D7 pins and port E0 to port E3 pins are write-only pins. If these bits are read, the read data are always 1's.

IDRs latch the input data at ports. Accordingly, reading a port means reading the corresponding IDR.

In addition to the above registes, port A is provided with a port A disable register (DERA). Port A pins are configured as either I/O ports or ASCI channel 1 or DMA channel 1 pins depending on this register.

16.1 Port A

Port A is an 8-bit I/O port. Port A pins can also be used as ASCI channel 1 pins or DMA channel 1 pins.

This port has a data direction register A (DDRA) for determining the direction of port A pins, a port A output data register (ODRA) for holding output data and a port A input data register (IDRA) for latching input data (figure 16-1).

It also has a port A disable register (DERA) which determines whether the individual port A pins are used as I/O pins or not. This is an 8-bit read/write register. Setting all DERA bits to 1 disables the port A pins, thus preventing them from functioning as I/O pins and enabling them to function as ASCI or DMA pins. On the other hand, clearing DERA bits enables port A pins to function as I/O pins.

DERA can affect DDRA. When using port A as an I/O port again after writing 1's to DERA, determine the direction of the port A pins.

Port A Output Data Register (ODRA: I/O Address = 60H)

bit	7	6	5	4	3	2	1	0
	ODRA7	ODRA6	ODRA5	ODRA4	ODRA3	ODRA2	ODRA1	ODRA0
	W	W	W	W	W	W	W	W

Data Direction Register A (DDRA: I/O Address = 70H)

bit	7	6	5	4	3	2	1	0
	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port A Input Data Register (IDRA: I/O Address = 60H)

bit	7	6	5	4	3	2	1	0
	IDRA7	IDRA6	IDRA5	IDRA4	IDRA3	IDRA2	IDRA1	IDRA0
	R	R	R	R	R	R	R	R

Port A Disable Register (DERA: I/O Address = 53H)

bit	7	6	5	4	3	2	1	0
	TEND1E	DREQ1E	CKSE	RXSE	TXSE	CKA1E	RXA1E	TAX1E
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Figure 16-1. Port A Registers

The individual pins are described below.

16.1.1 PA_0, PA_3, PA_7 (TXA_1, TXS, $\overline{TEND_1}$)

Figure 16-2 shows the PA_0, PA_3, and PA_7 block diagram. At reset, both DDRA and DERA are initialized to 0's; this configures these pins as inputs. In order to configure these pins as outputs after reset, the CPU must set the corresponding DDRA bits to 1's.

When the corresponding DERA bits (TXA1E, TXSE, TEND1E) are set to 1's, these pins are configured as TXA_1, TXS and $\overline{TEND_1}$ outputs. At this time, DDRA0, DDRA3, and DDRA7 are forced to 1's. Accordingly, in order to configure DDRA0, DDRA3, and DDRA7 as inputs when they are used as I/O pins again, the CPU must write 0's to DDRA0, DDRA3, and DDRA7 bits.

When these pins are used as TXA_1, TXS and $\overline{TEND_1}$, the read data of the port A_0, port A_3, and port A_7 pins are always 1's.

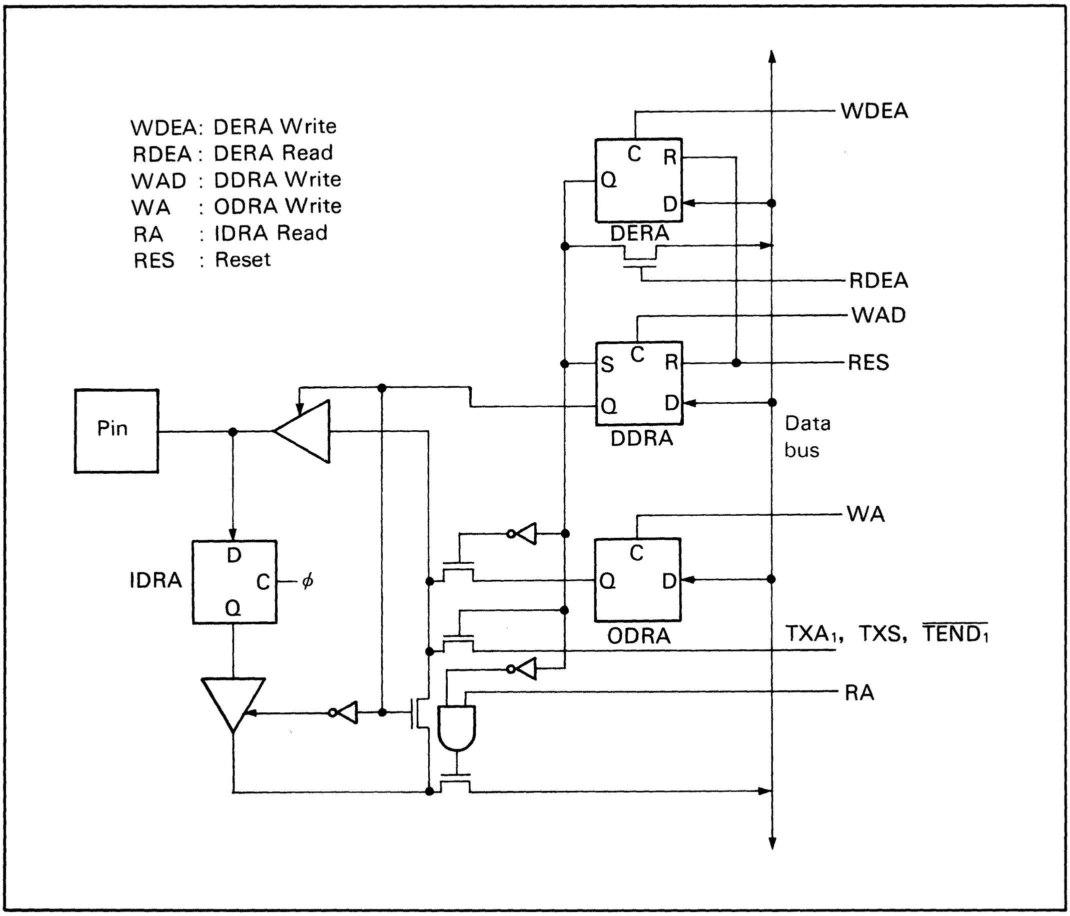


Figure 16-2. PA_0, PA_3, and PA_7 Block Diagram

16.1.2 PA_1, PA_4, PA_6 (RXA_1, $RXS/\overline{CTS_1}$, $\overline{DREQ_1}$)

Figure 16-3 shows the PA_1, PA_4, and PA_6 block diagram. At reset, both DDRA and DERA are initialized to 0's, which configures these pins as inputs. In order to configure these pins as outputs after reset, the CPU must set the corresponding DDRA bits to 1's.

When the corresponding DERA bits (RXA1E, RXSE, DREQE) are set to 1's, these pins are configured as RXA_1, RXS, and $\overline{DREQ_1}$ inputs. At this time, DDRA1, DDRA4, and DDRA6 are forced to 0's. Accordingly, in order to configure them as outputs when they are used as I/O pins again, the CPU must write 1's to DDRA1, DDRA4, and DDRA6 bits.

The configuration of $RXS/\overline{CTS_1}$ depends on the state of the CTS1E bit of the ASCI status register channel 1.

When these pins are used as RXA_1, RXS, and $\overline{DREQ_1}$, the read data of the port A_1, port A_4, and port A_6 pins are always 1's.

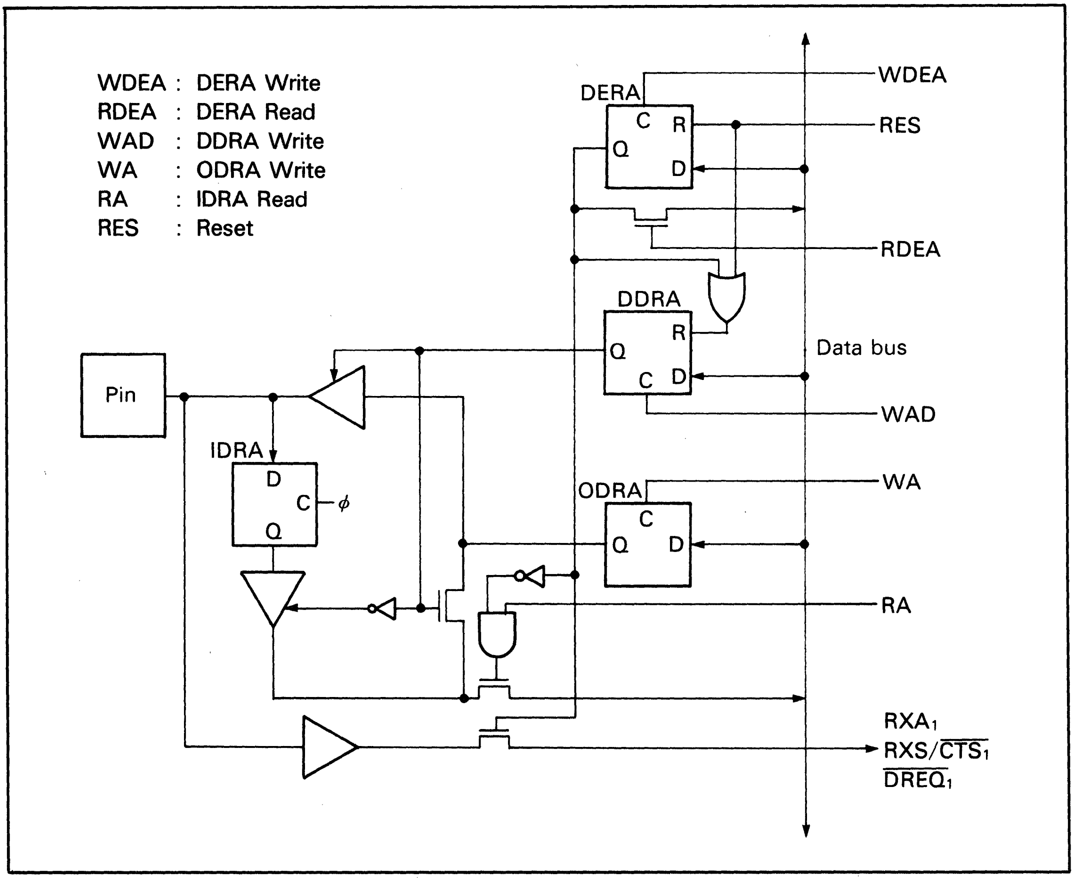


Figure 16-3. PA_1, PA_4, and PA_6 Block Diagram

16.1.3 PA_2, PA_5 ($CKA_1/\overline{TEND_0}$, CKS)

Figure 16-4 shows the PA_2 and PA_5 block diagram. At reset, both DDRA and DERA are initialized to 0's, which configures these pins as inputs. In order to configure these pins as outputs after reset, the CPU must set the corresponding DDRA bits to 1's.

When the corresponding DERA bits (CKA1E, CKSE) are set to 1's, these pins are configured as CKA_1 and CKS inputs.

Since the DDR bits corresponding to PA_2 and PA_5 are not affected by DERA, it is not necessary to configure these bits again when using these pins as I/O ports after using them as CKA_1 and CKS.

The configuration of $CKA_1/\overline{TEND_0}$ depends on the state of the CKA1D bit of the ASCI status register channel 1.

When these pins are used as $CKA_1/\overline{TEND_0}$ and CKS, the read data of the port A_2 and port A_7 pins are always 1's.

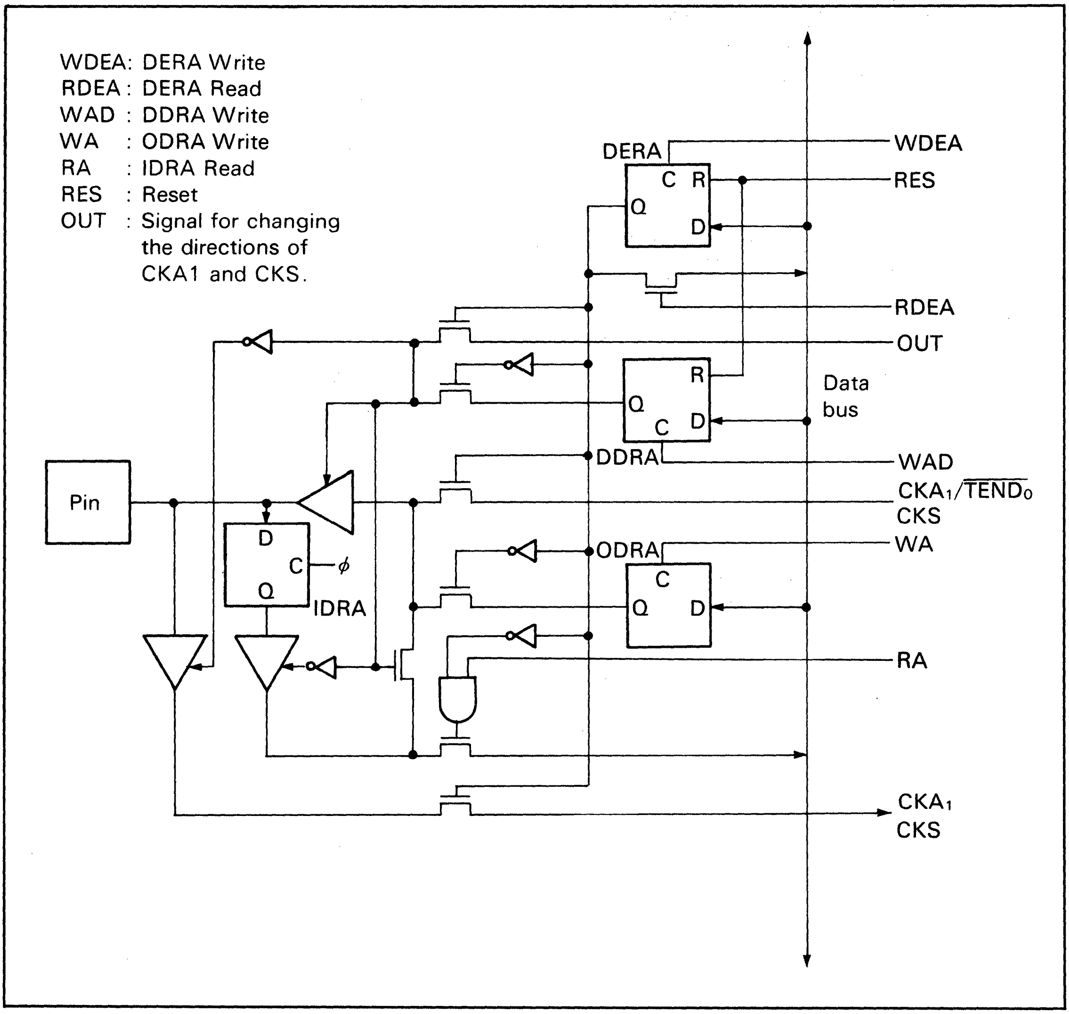


Figure 16-4. PA_2, PA_5 Block Diagram

16.2 Port B

Port B is an 8-bit input/output port. Input/output switching is performed by data direction register B (DDRB). When a bit of DDRB is reset to 0, the corresponding bit of port B can be used as an input port. To use a bit of port B as an output port, set the corresponding bit of DDRB to 1 (figure 16-5).

In the expanded mode (modes 1 and 2), port B is used for bus state control, and the input/output direction by DDRB is ignored.

This port is read as 1 during expanded mode (modes 1 and 2) operation.

The block configuration of port B is shown in figure 16-6.

Port B Output Data Register (ODRB: I/O Address = 61H)

bit	7	6	5	4	3	2	1	0
	ODRB7	ODRB6	ODRB5	ODRB4	ODRB3	ODRB2	ODRB1	ODRB0
	W	W	W	W	W	W	W	W

Data Direction Register B (DDRB: I/O Address = 71H)

bit	7	6	5	4	3	2	1	0
	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port B Input Data Register (IDRB: I/O Address = 61H)

bit	7	6	5	4	3	2	1	0
	IDRB7	IDRB6	IDRB5	IDRB4	IDRB3	IDRB2	IDRB1	IDRB0
	R	R	R	R	R	R	R	R

Figure 16-5. Port B Registers

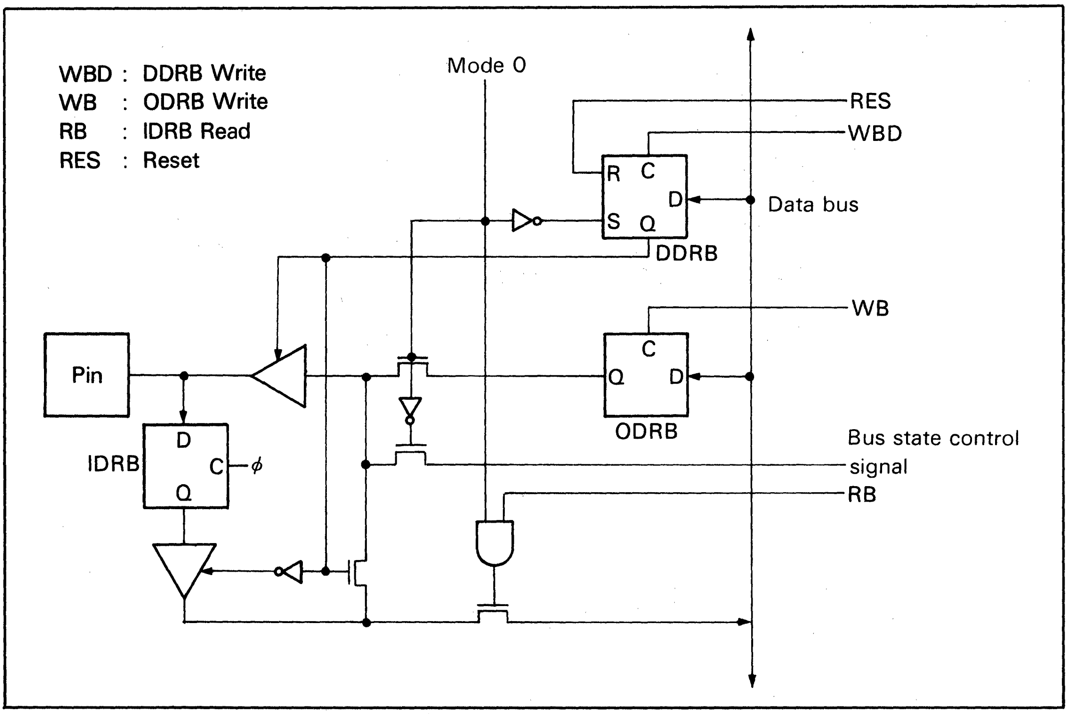


Figure 16-6. Port B Block Diagram

16.3 Port C

Port C is an 8-bit input/output port. Input/output switching is performed by data direction register C (DDRC). When a bit of DDRC is reset to 0, the corresponding bit of port C can be used as an input port. To use a bit of port C as an output port, set the corresponding bit of DDRC to 1 (figure 16-7).

In the expanded mode (modes 1 and 2), port C is used for bus state control, and the input/output direction by DDRC is ignored.

This port is as 1 read during expanded mode (modes 1 and 2) operation.

The block configuration of port C is shown in figure 16-8.

Port C Output Data Register (ODRC: I/O Address = 62H)

bit	7	6	5	4	3	2	1	0
	ODRC7	ODRC6	ODRC5	ODRC4	ODRC3	ODRC2	ODRC1	ODRC0
	W	W	W	W	W	W	W	W

Data Direction Register C (DDRC: I/O Address = 72H)

bit	7	6	5	4	3	2	1	0
	DDRC7	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port C Input Data Register (IDRC: I/O Address = 62H)

bit	7	6	5	4	3	2	1	0
	IDRC7	IDRC6	IDRC5	IDRC4	IDRC3	IDRC2	IDRC1	IDRC0
	R	R	R	R	R	R	R	R

Figure 16-7. Port C Registers

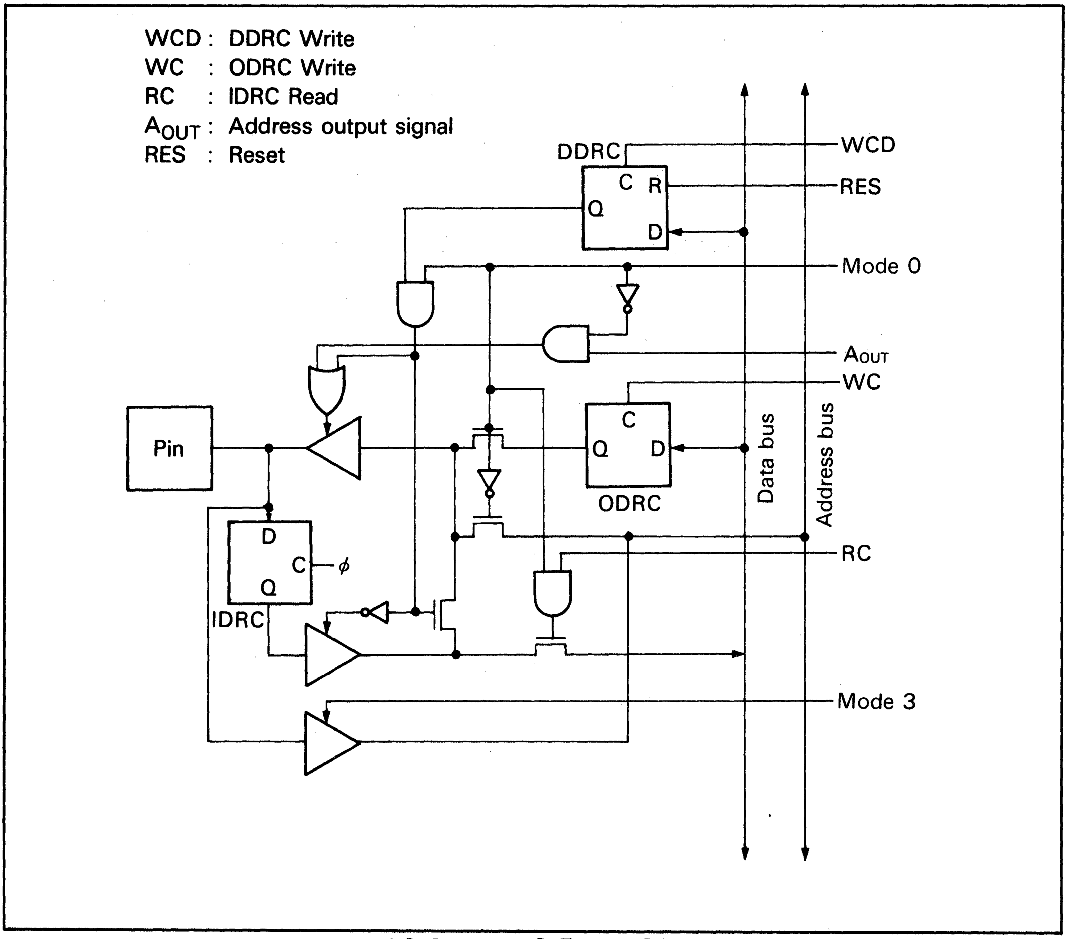


Figure 16-8. Port C Block Diagram

16.4 Port D

Port D is an 8-bit input/output port. Input/output switching is performed by data direction register D (DDRD). When a bit of DDRD is reset to 0, the corresponding bit of port D can be used as an input port. To use a bit of port D as an output port, set the corresponding bit of DDRD to 1 (figure 16-9).

When port D is used as an output port, only 1 is read from the port D input data register (IDRD).

This port is used as an address bus output in mode 1 of the expanded mode (expansion without a built-in ROM), and the input/output direction by DDRD is ignored.

This port is the dedicated input terminal in mode 2 (expansion with a built-in ROM). However in mode 2, when DDRD is set to 1, the upper addresses (A_8–A_{15}) are output.

This port is read as 1 during expansion mode operation (mode 1 and mode 2).

The block configuration of port D is shown in figure 16-10.

Port D Output Data Register (ODRD: I/O Address = 63H)

bit	7	6	5	4	3	2	1	0
	ODRD7	ODRD6	ODRD5	ODRD4	ODRD3	ODRD2	ODRD1	ODRD0
	W	W	W	W	W	W	W	W

Data Direction Register D (DDRD: I/O Address = 73H)

bit	7	6	5	4	3	2	1	0
	DDRD7	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port D Input Data Register (IDRD: I/O Address = 63H)

bit	7	6	5	4	3	2	1	0
	IDRD7	IDRD6	IDRD5	IDRD4	IDRD3	IDRD2	IDRD1	IDRD0
	R	R	R	R	R	R	R	R

Figure 16-9. Port D Registers

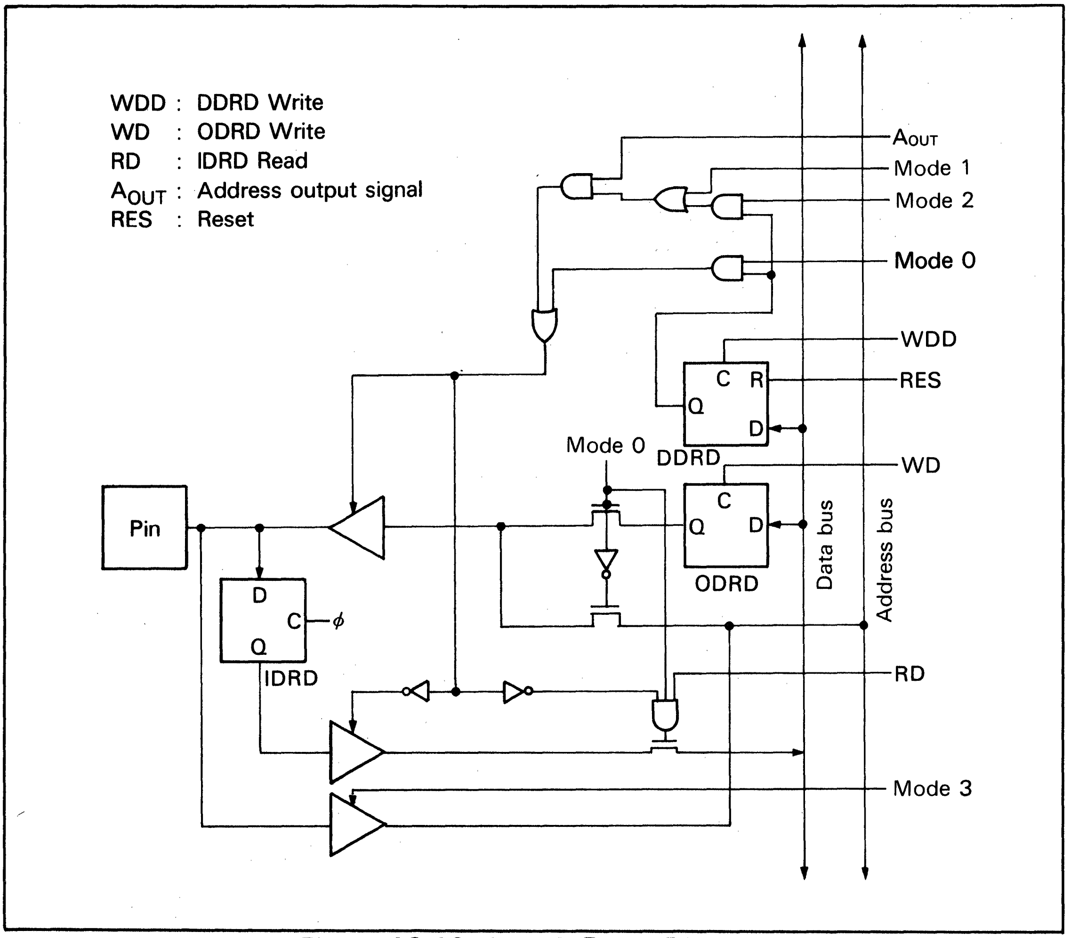


Figure 16-10. Port D Block Diagram

16.5 Port E

Port E is an 8-bit input/output port. Input/outpuit switching is performed by data direction register E (DDRE). When a bit of DDRE is reset to 0, the corresponding bit of port E can be used as an input port. To use a bit of port E as an output port, set the corresponding bit of DDRE to 1 (figure 16-11).

When this port is used as an output port, if port E input data register (IDRE) is read, 1 will be read from PE_0-PE_3, and the contents of port E output data register (ODRE) will be read from PE_4-PE_7.

Mode 1 of the expanded mode (expansion without a built-in ROM), PE_0-PE_3 are used as address bus output ports, while PE_4-PE_7 are used as bus state control signals.

In mode 2 (expansion with a built-in ROM), PE_4-PE_7 are used as bus state control signals, while PE_0-PE_3 are used as dedicated input ports. However, when DDRE 0–DDRE3 are set to 1, the upper addresses ($A_{16}-A_{19}$) are output.

This port is read as 1 during expanded mode (mode 1 and mode 2) operation.

The block diagram of port E is shown in figures 16-12 to 16-14.

Port E Output Data Register (ODRE: I/O Address = 64H)

bit	7	6	5	4	3	2	1	0
	ODRE7	ODRE6	ODRE5	ODRE4	ODRE3	ODRE2	ODRE1	ODRE0
	W	W	W	W	W	W	W	W

Data Direction Register E (DDRE: I/O Address = 74H)

bit	7	6	5	4	3	2	1	0
	DDRE7	DDRE6	DDRE5	DDRE4	DDRE3	DDRE2	DDRE1	DDRE0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port E Input Data Register (IDRE: I/O Address = 64H)

bit	7	6	5	4	3	2	1	0
	IDRE7	IDRE6	IDRE5	IDRE4	IDRE3	IDRE2	IDRE1	IDRE0
	R	R	R	R	R	R	R	R

Figure 16-11. Port E Registers

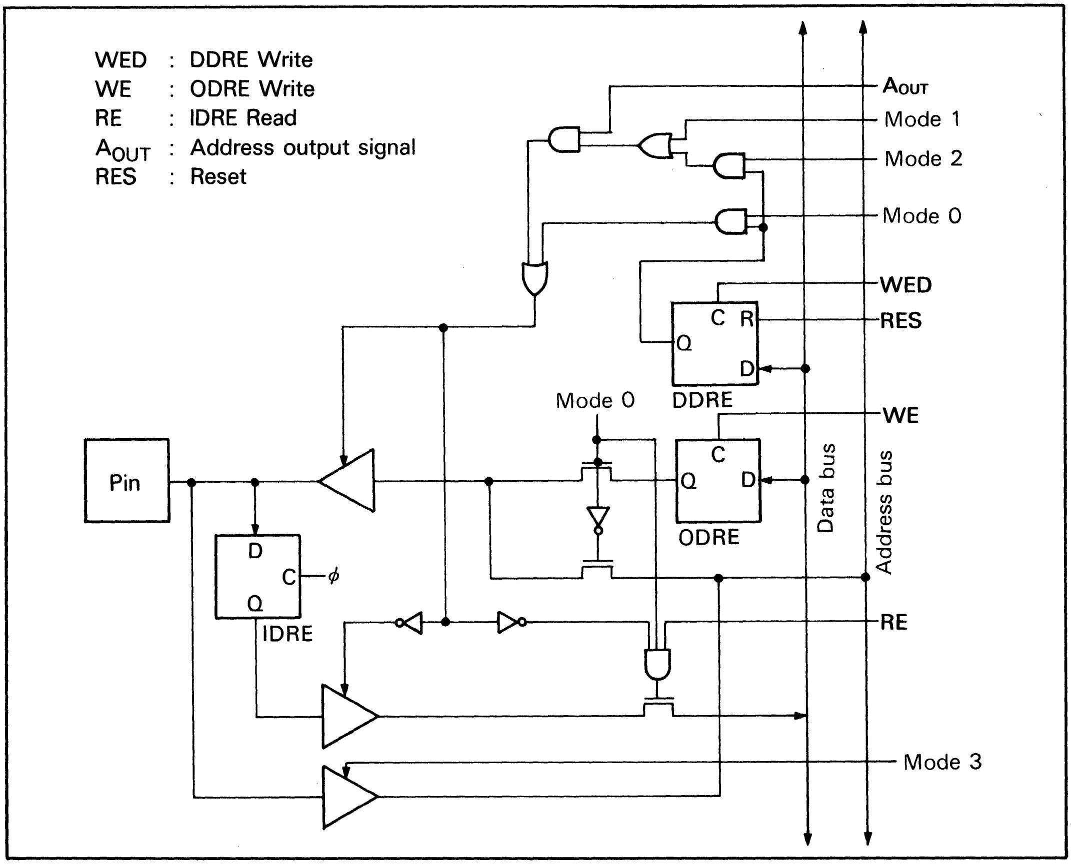


Figure 16-12. PE_0—PE_3 Block Diagram

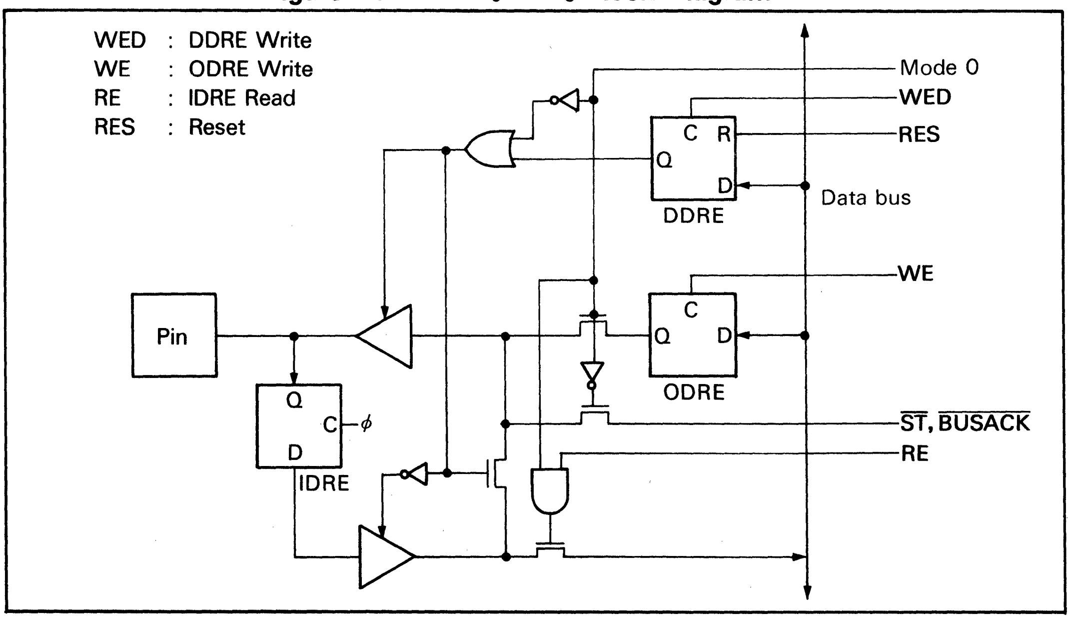


Figure 16-13. PE_4, PE_6 Block Diagram

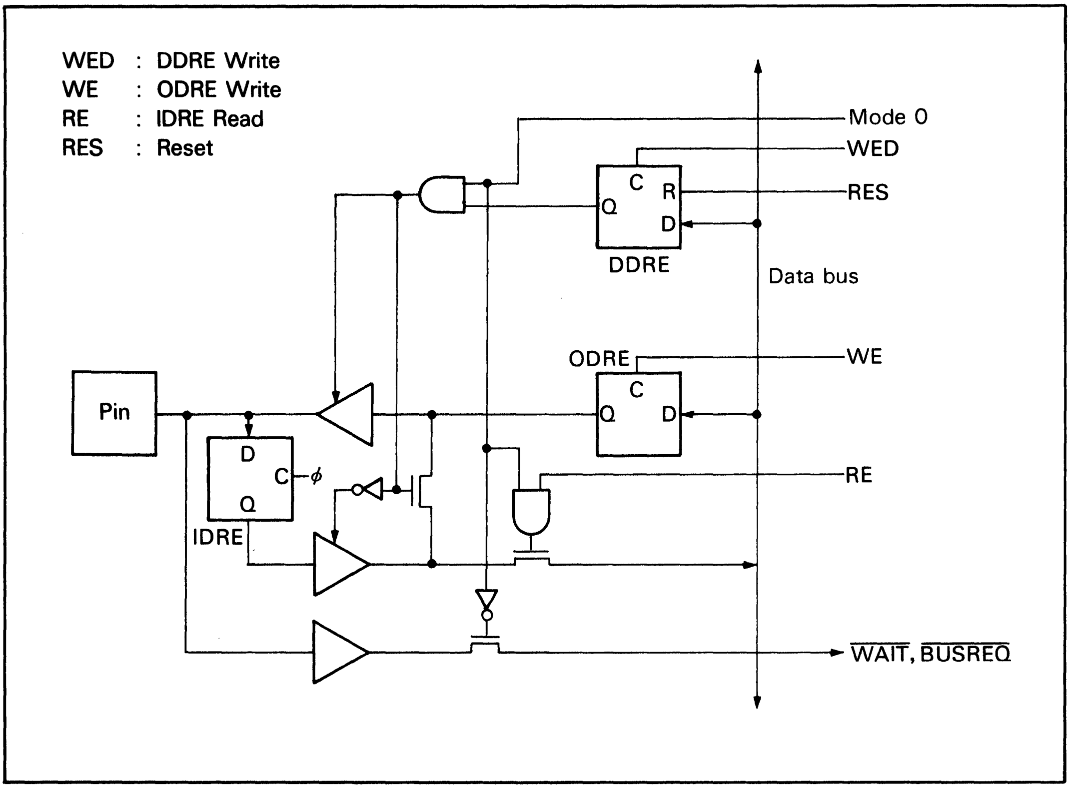


Figure 16-14. PE$_5$, PE$_7$ Block Diagram

16.6 Port F

Port F is an 8-bit input/output port. Input/output switching is performed by data direction register F (DDRF). When a bit of DDRF is reset to 0, the corresponding bit of port F can be used as an input port. To use a bit of port F as an output port, set the corresponding bit of DDRF to 1 (figure 16-15).

In the expanded mode (modes 1 and 2), port F is used as a data bus.

This port is read as 1 during expanded mode (modes 1 and 2) operation.

The block configuration of port F is shown in figure 16-16.

Port F Output Data Register (ODRF: I/O Address = 65H)

bit	7	6	5	4	3	2	1	0
	ODRF7	ODRF6	ODRF5	ODRF4	ODRF3	ODRF2	ODRF1	ODRF0
	W	W	W	W	W	W	W	W

Data Direction Register F (DDRF: I/O Address = 75H)

bit	7	6	5	4	3	2	1	0
	DDRF7	DDRF6	DDRF5	DDRF4	DDRF3	DDRF2	DDRF1	DDRF0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port F Input Data Register (IDRF: I/O Address = 65H)

bit	7	6	5	4	3	2	1	0
	IDRF7	IDRF6	IDRF5	IDRF4	IDRF3	IDRF2	IDRF1	IDRF0
	R	R	R	R	R	R	R	R

Figure 16-15. Port F Registers

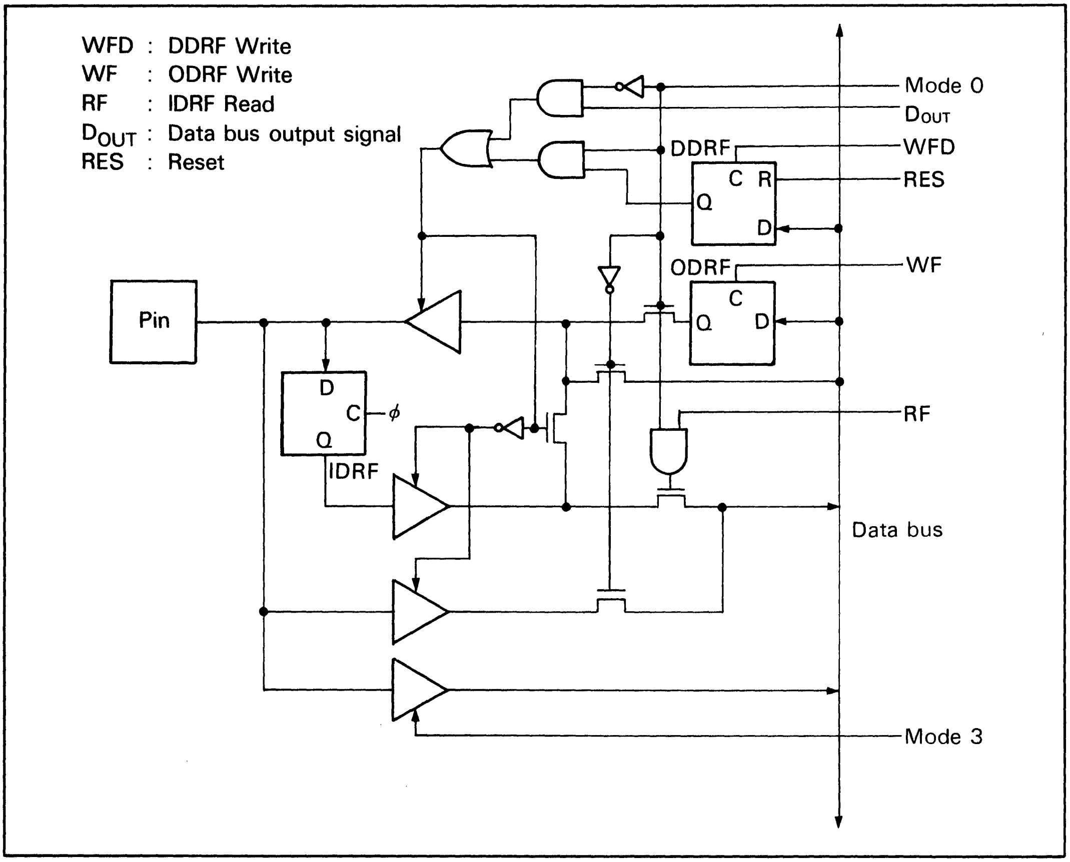


Figure 16-16. Port F Block Diagram

16.7 Port G

Port G is a dedicated 6-bit input port. This port is multiplexed with the channel inputs of the analog converter.

When using this port as a TTL input port, read the port G input data register (IDRG) (figure 16-17). When using this port as an analog comparator read the, comparator control/status register (CCSR). Comparison results are shown by the result bit.

However, port G bits cannot be used bit-by-bit as TTL-level input terminals or the analog comparator's channel input. If IDRG is read when using port G as the channel input, through current may flow. The block configuration of port G is shown in figure 16-18.

Port G Input Data Register (IDRG: I/O Address = 66H)

bit	7	6	5	4	3	2	1	0
	1	1	IDRG5	IDRG4	IDRG3	IDRG2	IDRG1	IDRG0
			R	R	R	R	R	R

Figure 16-17. Port G Input Data Register

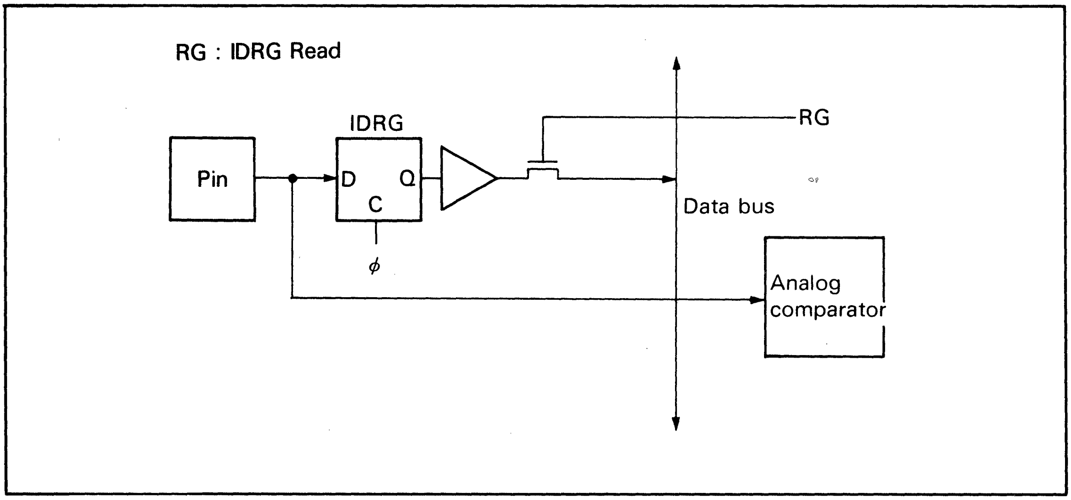


Figure 16-18. Port G Block Diagram

SECTION 17. MEMORY SPACE

The HD647180X has a built-in 16-kbyte PROM and a 512-byte RAM. The ROM and RAM are stored in physical addresses 00000H–03FFFH and 0FE00H–0FFFFH, respectively. RAM can be relocated every 64-kbyte by controlling the RAM control register.

The memory space is each operation mode is shown in figure 17-1.

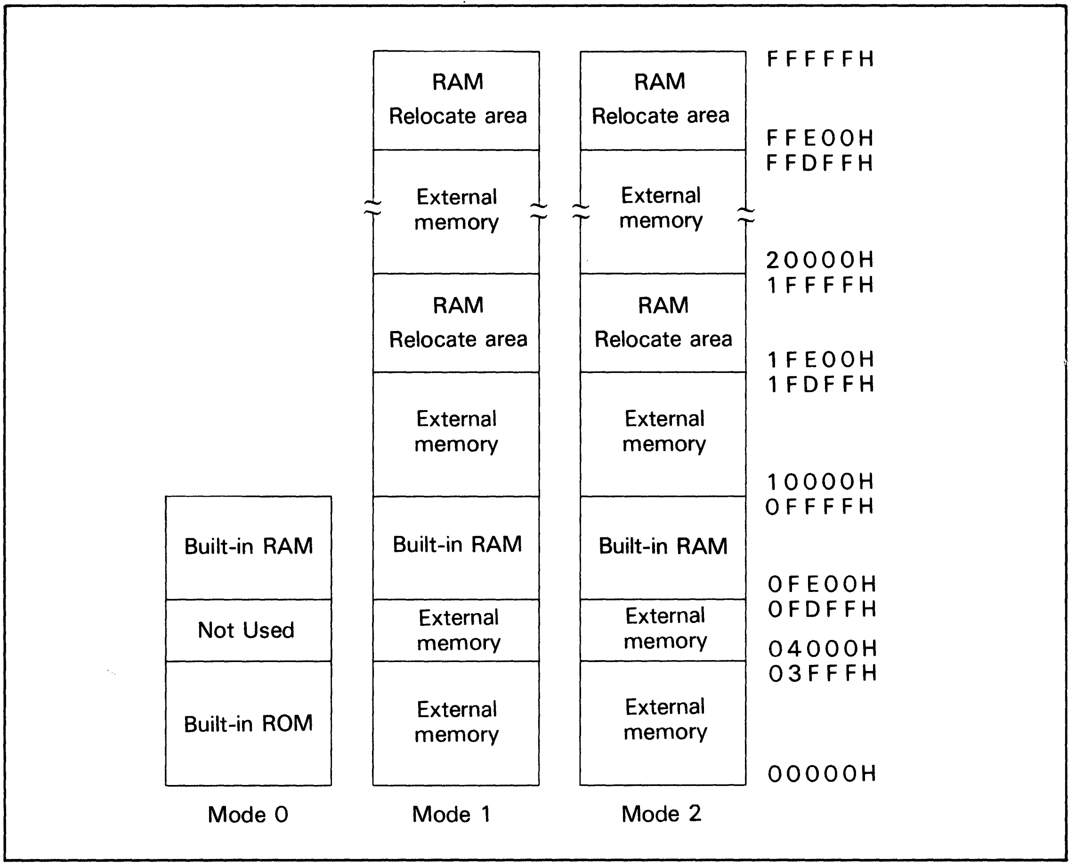


Figure 17-1. Memory Space of Each Operation Mode

RAM control register (RMCR)

RMCR allows the internal RAM addresses to be relocated.

RAM Control Register (RMCR: I/O Address=51H)

bit	7	6	5	4	3	2	1	0
	RMRB7	RMRB6	RMRB5	RMRB4	–	–	–	–
	R/W	R/W	R/W	R/W				

Figure 17-2. RAM Control Register

RMRB7-4: RAM Address Relocation bit (Bit 7-4)
Bits 7-4 relocate internal RAM as shown in figure 17-2.
Bits 7-4 are cleared to 0 during reset.

· RAM addresses

XXXX	1111	1110	0000	0000
		↓		
XXXX	1111	1111	1111	1111

XXXX ·················· RMRB7-RMRB4

SECTION 18. 6800-TYPE BUS INTERFACE

18.1 E Clock Output Timing

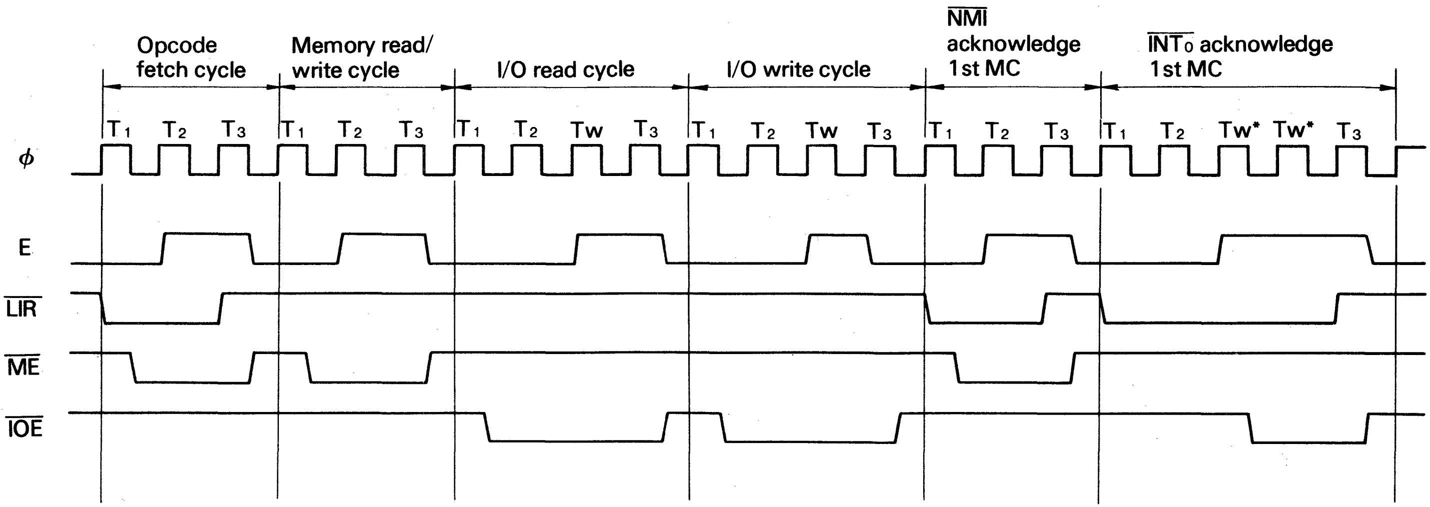
A large selection of 6800-type peripheral devices can be connected to the HD647180X, including the Hitachi 6300 CMOS series (6321 PIA, 6350 ACIA, etc.) as well as 6800 family devices.

These devices require connection with the HD647180X synchronous E clock output. The speed (access time) required for the peripheral device are determined by the HD647180X clock rate. Table 18-1, figure 18-1 and 18-2 define E clock output timing.

Table 18-1. E Clock Timing

Condition	Duration of E Clock Output High	
Opcode Fetch Cycle Memory Read/Write Cycle	$T_2\uparrow$ to $T_3\downarrow$	$(1.5\phi + n_W \cdot \phi)$
I/O Read Cycle	1st $T_W\uparrow$ to $T_3\downarrow$	$(0.5\phi + n_W \cdot \phi)$
I/O Write Cycle	1st $T_W\uparrow$ to $T_3\uparrow$	$(n_W \cdot \phi)$
$\overline{\text{NMI}}$ Acknowledge, 1st MC	$T_2\uparrow$ to $T_3\downarrow$	(1.5ϕ)
$\overline{\text{INT}_0}$, $\overline{\text{INT}_1}$, $\overline{\text{INT}_2}$, and Internal Interrupt Acknowledge, 1st MC	1st $T_W\uparrow$ to $T_3\downarrow$	$(0.5\phi + n_W \cdot \phi)$
Bus Release Mode Sleep Mode System Stop Mode	$\phi\downarrow$ to $\phi\downarrow$	$(2\phi$ or $1\phi)$

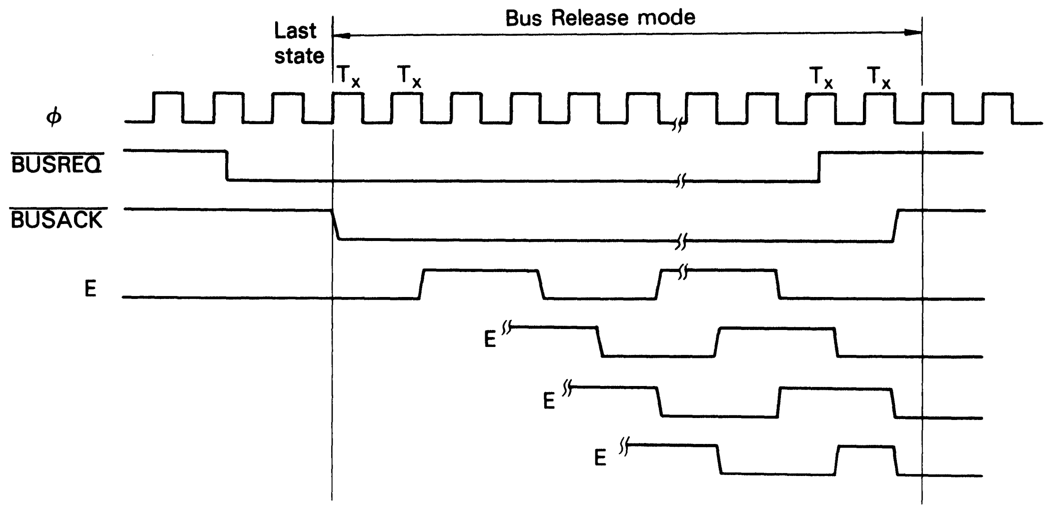
Note: n_W: Number of wait states
MC: Machine Cycle



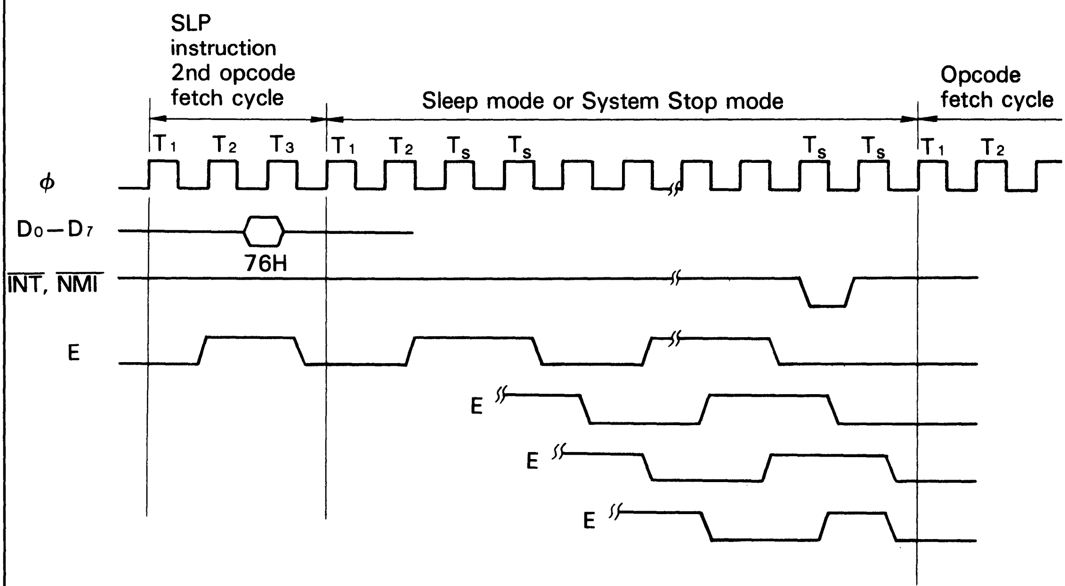
MC: Machine Cycle

* Two wait states are automatically inserted.

Figure 18-1. E Clock Timing (During Read/Write Cycle and Interrupt Acknowledge Cycle)



E Clock Timing in Bus Release Mode



E Clock Timing in Sleep Mode and System Stop Mode

Figure 18-2. E Clock Timing (in Bus Release mode, Sleep mode, System Stop mode)

Wait states inserted in opcode fetch, memory read/write and I/O read/write cycles extend the duration of E clock output high. Note that during I/O read/write cycles with no wait states (only occurs during on-chip I/O register accesses), E will not go high.

The correspondence between the duration of E clock output high and standard peripheral device speed selections is shown in table 18-2.

Table 18-2 Device Speed and E Clock Timing

Device Speed Selection	Required Duration of E Clock Output High
1.0 MHz (ex: HD6321P)	500 ns min
1.5 MHz (ex: HD63A21P)	333 ns min
2.0 MHz (ex: HD63B21P)	230 ns min

18.2 6800-Type Bus Interfacing Note

When the HD647180X is connected to 6800-type peripheral LSIs with E clock, the 6800-type peripheral LSIs should be located in I/O address space.

If the 6800-type peripheral LSIs are located in memory address space, $\overline{WR}$ set-up time and $\overline{WR}$ hold time for E clock won't be guaranteed during memory read/write cycles and 6800-type peripheral LSIs can't be connected correctly.

SECTION 19. ON-CHIP CLOCK GENERATOR

The HD647180X contains a crystal oscillator and system clock (ϕ) generator. A crystal can be directly connected or an external clock input can be provided. In either case, the system clock (ϕ) is equal to one-half the input clock. For example, a crystal or external clock input of 8 MHz corresponds with a system clock rate of ϕ = 4 MHz.

Table 19-1 shows the AT-cut crystal characteristics (Co, Rs) and the load capacitance (CL_1, CL_2) required for various frequencies of HD647180X operation.

Table 19-1. Crystal Characteristics

Item	f = 4 MHz	4 MHz < f ≦ 12 MHz	12 MHz < f ≦ 16 MHz
Co	< 7 pF	< 7 pF	< 7 pF
Rs	< 60 Ω	< 60 Ω	< 35 Ω
CL_1, CL_2	10 to 22 pF ±10%	10 to 22 pF ±10%	10 to 22 pF ±10%

If an external clock input is used instead of a crystal, the waveform (twice the ϕ clock rate) should exhibit a 50% ± 10% duty cycle. Note that the minimum clock input high voltage level is V_{CC} − 0.6 V. The external clock input is connected to the EXTAL pin, while the XTAL pin is left open. Figure 19-1 shows an external clock interface.

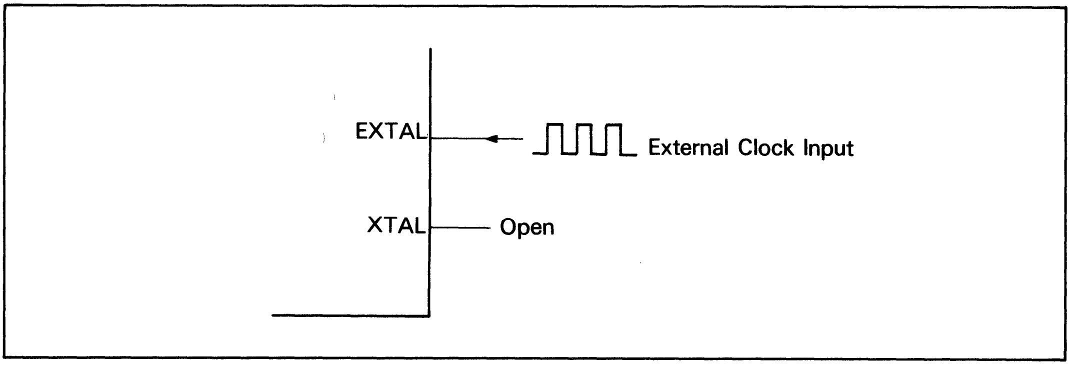


Figure 19-1. External Clock Interface

Figure 19-2 shows the HD647180X clock generator circuit while figures 19-3 and 19-4 specify circuit board design rules.

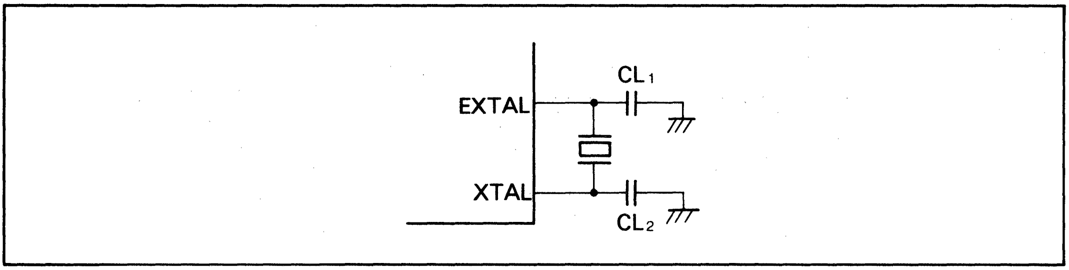


Figure 19-2. Crystal Interface

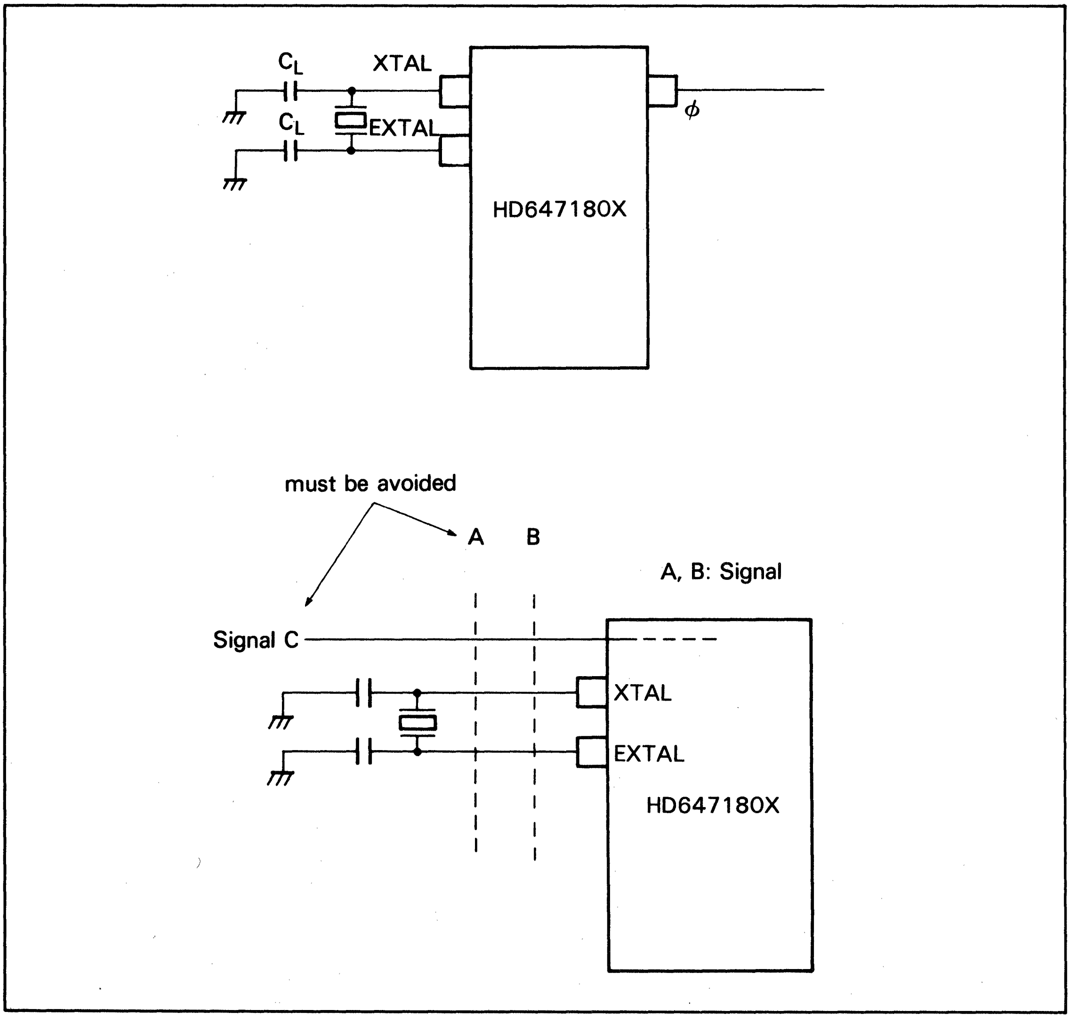


Figure 19-3. Note for Board Design of the Oscillation Circuit

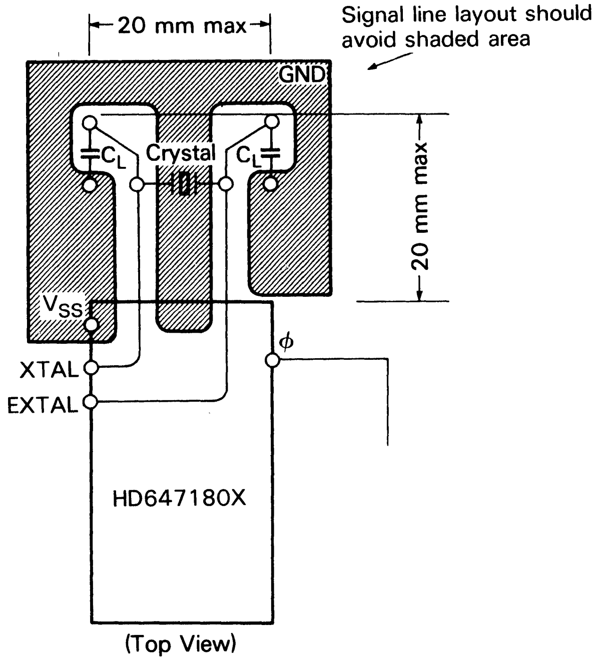


Figure 19-4. Board Design Example

Circuit board design should observe the followings:

1. To prevent induced noise, the crystal and load capacitors should be physically located as close to the LSI as possible.
2. Signal lines should not run parallel to the clock oscillator inputs. In particular, the clock input circuitry and the system clock ϕ output should be separated as much as possible.
3. Similarly, V_{CC} power lines should be separated from the clock oscillator input circuitry.
4. Resistivity between XTAL or EXTAL and the other pins should be greater than 10M ohms.
   Signal line layout should avoid shaded area in figure 19-4.

SECTION 20. FREE-RUNNING COUNTER

Read only 8-bit free-running counter (I/O Address = 18H) has no control registers or status registers. The contents of the 8-bit free-running counter are counted down by 1 every 10 ϕ clock cycles. The free-running counter continues counting down without being affected by the read operation.

If data is written into the free-running counter, we can't guarantee the interval of DRAM refresh cycle and the baud rates of ASCI and CSI/O.

In I/O stop mode, the free-running counter continues counting down. It is initialized to FFH during reset.

SECTION 21. OPERATING MODES AND PROM PROGRAMMING

21.1 Operating Modes

The HD647180X provides four operating modes which are determined by the MP_0 and MP_1 mode programming pins as shown in table 21-1.

Table 21-1. Operating Mode Selection

Mode	MP_1	MP_0	ROM	RAM	Operating Mode
0	Low	Low	Int	Int	Single chip mode
1	Low	High	Ext	Int	Expanded mode 1
2	High	Low	Int	Int	Expanded mode 2
3	High	High	Int	—	PROM programming mode

However, the HD647180X can be fixed in mode 0 independent of the state of MP_0 and MP_1 by using an internal ROM data protection function. For more detailed information, see 21.2, Data Protect Function.

21.1.1 Mode 0 (Single Chip Mode)

In mode 0, ports A to G can be used simultaneously as I/O ports.

21.1.2 Mode 1 (Expanded Mode 1)

In mode 1, port C, port D, and port E_0 to port E_3 pins are configured as address bus, port F as the data bus, and port B and port E_4 to port E_7 pins as the bus state control signal inputs or outputs.

Accordingly, only port A and port G can be used as I/O ports. In addition, the internal ROM is disabled but the HD647180X can provide an external physical space of 1-Mbyte by using MMU.

21.1.3 Mode 2 (Expanded Mode 2)

Mode 2 is almost the same as mode 1. In this mode, however, the internal ROM is enabled, which allows the physical space 00000H to 03FFFH to be provided as internal ROM space. In addition, port D_0 to port D_7 pins and port E_0 to port E_3 pins are initialized as inputs at reset, which disallows high-order addresses to be output. Accordingly, after reset, the CPU must write 1's to the DDR bits corresponding to the address output pins to allow for high-order address output.

If the HD647180X has a small external memory space, the pins which do not output addresses can be used as input ports.

21.1.4 Mode 3 (PROM Programming Mode)

In mode 3, the internal PROM can be programmed via a general-purpose PROM writer. However, an adapter is required to interface the PROM writer to the HD647180X.

21.2 Data Protect Function

The built-in PROM read-prohibition function has the following two modes:

1. Built-in PROM read-prohibition mode (valid in mode 3)
2. Expansion prohibition mode (valid in modes 1 and 2)

   To set the read prohibition mode, write FCH to address 4000H in the PROM mode, and the built-in PROM (addresses 00000H-03FFFH) read in mode 3 is prohibited. In this case, if the built-in PROM is accessed externally, FFH is read irrespective of the internal data.

To set the expansion prohibition mode, write FCH to address 5000H in the PROM mode. The HD647180X will be fixed in mode 0 (single-chip mode) regardless of the setting of MP_0 and MP_1. Accordingly, the expanded mode (modes 1 and 2) cannot be used.

Functions 1 and 2 can be used independently.

Note: The read prohibit function can prevent illegal software access from the outside by using the above functions 1 and 2. However, no countermeasures are taken against illegal hardware access such as probing inside the chip or irradiating the chip with a spot beam.

21.3 Programming the Built-in Programmable ROM

In the PROM mode, the built-in PROM can be programmed by halting the MCU function in PROM.

The PROM mode can be set by setting the MP_0 and MP_1 pins to high (figure 21-1).

The PROM read/write specification is the same as that of the commercially available EPROM 27256. Accordingly, programming can be performed using a general PROM writer and a socket adapter which converts 80 pins to 28 pins (table 21-2). In this case, since the PROM capacity is 16-kbyte, addresses 0000H-3FFFH must be specified.

21.3.1 Write/verifiy

The HD647180X can be programmed by the high-speed programming method.

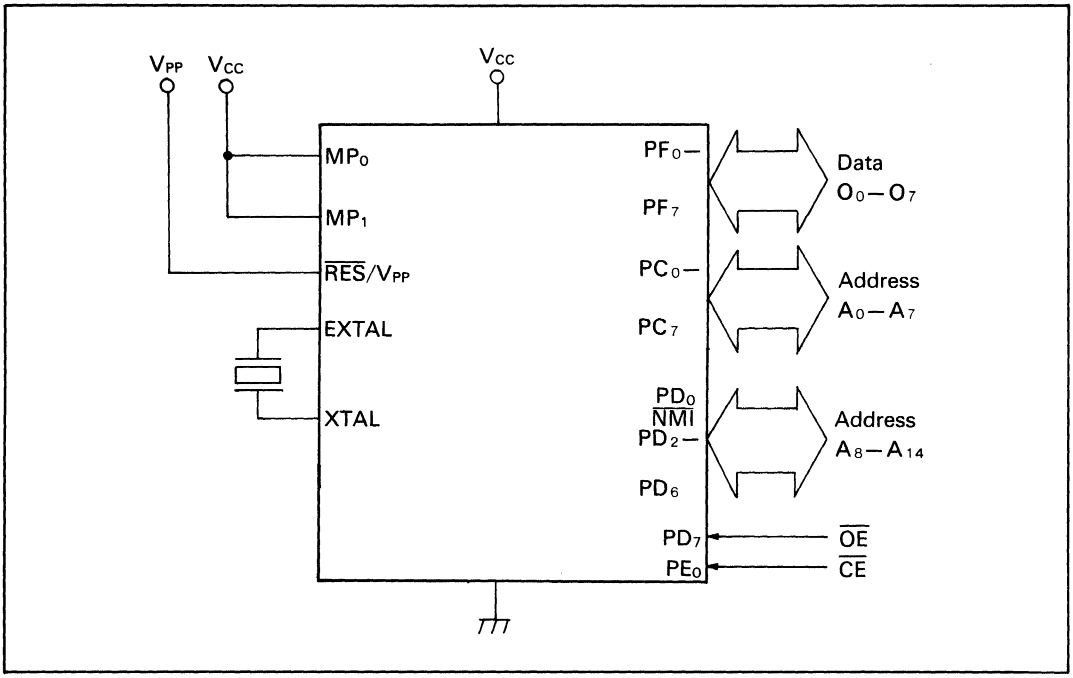


Figure 21-1. PROM Mode

Using this method, high-speed writing is enabled without causing voltage stress to the device and without degrading the reliability of write data.

A basic programming flow chart and timing diagram are shown in figures 21-2 and 21-3.

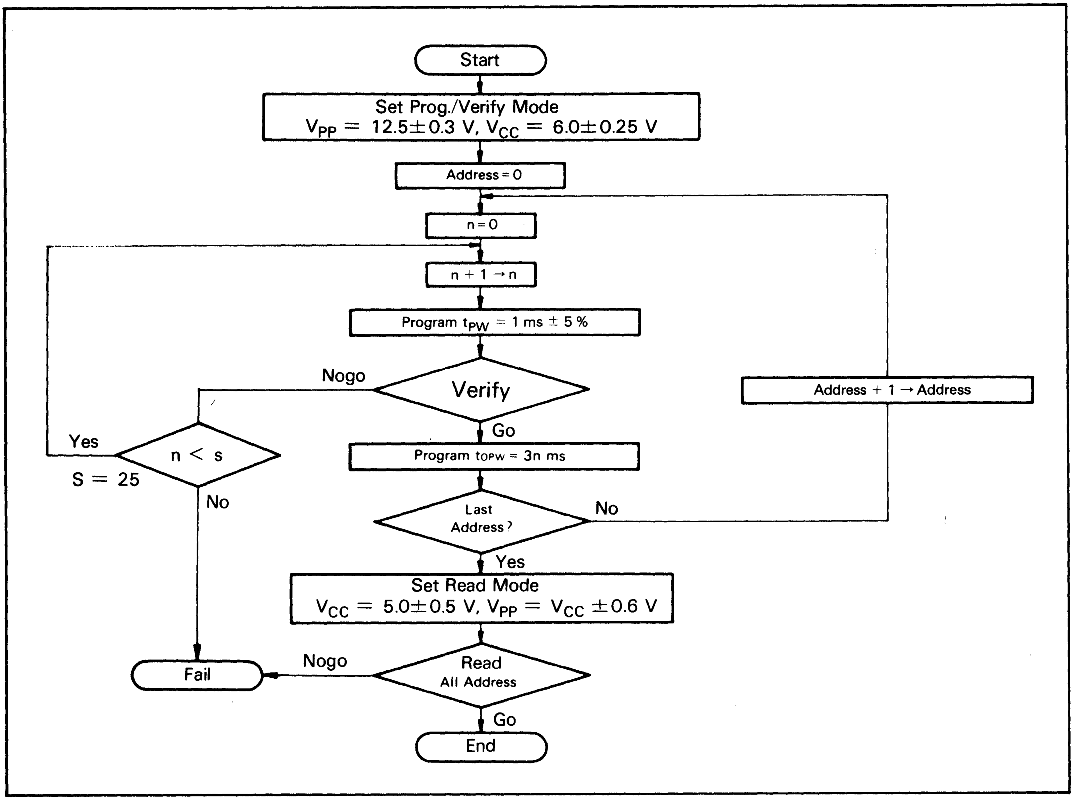


Figure 21-2. High-Speed Programming Flowchart

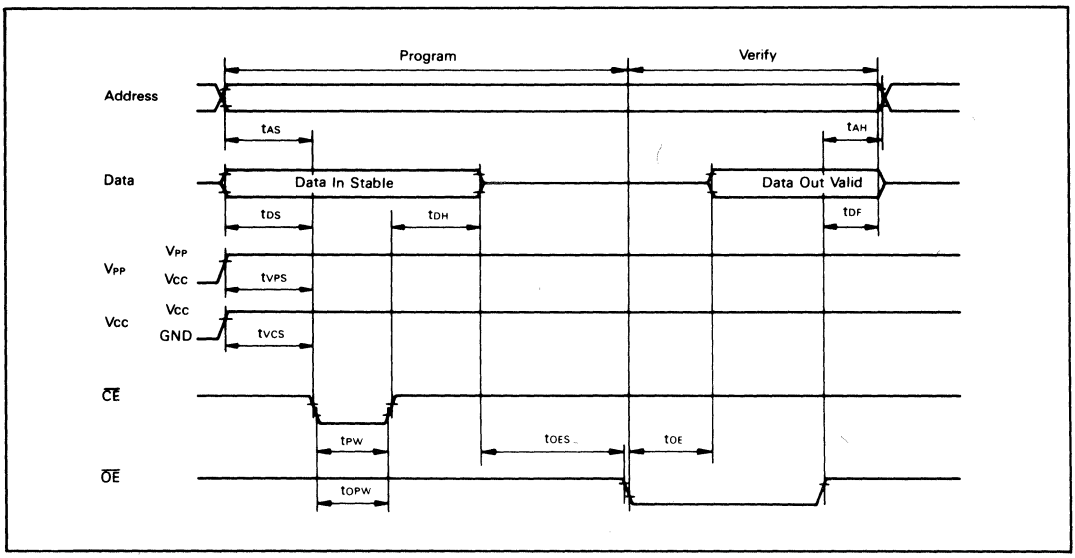


Figure 21-3. PROM Program/Verify Timing

21.3.2 Notes on PROM Programming

When programming using a PROM writer, the addresses must be specified as 0000H-3FFFH. Specify unused address data as FFH.

If there is a discrepancy in the PROM writer's socket, socket adapter, or IC pin index, the product may be damaged due to surplus current. Please ensure that all IC and adaptor pins are correctly inserted into the writer.

For PROM program voltage (Vpp), two voltage levels (12.5 and 21 V) are available. The Vpp of this LSI is 12.5 V. If 21 V is applied, permanent malfunction results. The Vpp of the PROM writer becomes 12.5 V when the chip is set to 27256 Intel specification.

If they are programmed to address 4000H or 5000H, data protect function is enabled.

Table 21-2. PROM Programming Adaptor Pin Assignment

NO	PIN	ASSIGN	NO	PIN	ASSIGN
1	V_{cc}	V_{cc}	41	PF_4/D_4	O_4
2	XTAL	XTAL	42	PF_5/D_5	O_5
3	EXTAL	EXTAL	43	PF_6/D_6	O_6
4	$PE_7/\overline{WAIT}$	Open	44	PF_7/D_7	O_7
5	$PE_6/\overline{BUSACK}$	Open	45	V_{ss}	V_{ss}
6	$PE_5/\overline{BUSREQ}$	Open	46	PG_0/AN_0	V_{ss}
7	$\overline{RES}/V_{pp}$	V_{pp}	47	PG_1/AN_1	V_{ss}
8	$\overline{NMI}/EA_9$	A_9	48	PG_2/AN_2	V_{ss}
9	$\overline{INT_0}$	Pull up	49	PG_3/AN_3	V_{ss}
10	$\overline{INT_1}$	Pull up	50	PG_4/AN_4	V_{ss}
11	$\overline{INT_2}$	Pull up	51	PG_5/AN_5	V_{ss}
12	PE_4/ST	Open	52	$\overline{RTS_0}$	Open
13	PC_0/A_0	A_0	53	$\overline{CTS_0}$	Open
14	PC_1/A_1	A_1	54	$\overline{DCD_0}$	Open
15	PC_2/A_2	A_2	55	TXA_0	Open
16	PC_3/A_3	A_3	56	RXA_0	Open
17	V_{ss}	V_{ss}	57	$CKA_0/\overline{DREQ_0}$	Open
18	PC_4/A_4	A_4	58	TOUT2	Open
19	PC_5/A_5	A_5	59	TOUT3	Open
20	PC_6/A_6	A_6	60	IC	Open
21	PC_7/A_7	A_7	61	PA_0/TXA_1	Open
22	PD_0/A_8	A_8	62	PA_1/RXA_1	Open
23	PD_1/A_9	V_{ss}	63	$PA_2/CKA_1/\overline{TEND_0}$	Open
24	PD_2/A_{10}	A_{10}	64	PA_3/TXS	Open
25	PD_3/A_{11}	A_{11}	65	$PA_4/RXS/\overline{CTS_1}$	Open
26	PD_4/A_{12}	A_{12}	66	PA_5/CKS	Open
27	PD_5/A_{13}	A_{13}	67	$PA_6/\overline{DREQ_1}$	Open
28	PD_6/A_{14}	A_{14}	68	$PA_7/\overline{TEND_1}$	Open
29	PD_7/A_{15}	$\overline{OE}$	69	$PB_7/\overline{HALT}$	Open
30	PE_0/A_{16}	$\overline{CE}$	70	$PB_6/\overline{REF}$	Open
31	PE_1/A_{17}	V_{ss}	71	$PB_5/\overline{IOE}$	Open
32	PE_2/A_{18}	V_{ss}	72	$PB_4/\overline{ME}$	Open
33	TOUT1	Open	73	PB_3/E	Open
34	V_{cc}	V_{cc}	74	$PB_2/\overline{LIR}$	Open
35	PE_3/A_{19}	V_{ss}	75	$PB_1/\overline{WR}$	Open
36	V_{ss}	V_{ss}	76	$PB_0/\overline{RD}$	Open
37	PF_0/D_0	O_0	77	V_{ss}	V_{ss}
38	PF_1/D_1	O_1	78	ϕ	Open
39	PF_2/D_2	O_2	79	MP_1	Pull up
40	PF_3/D_3	O_3	80	MP_0	Pull up

21.3.3 Programming Electrical Characteristics

Table 21-3. DC Characteristics
(Unless specified, V_{cc}=6 V ±0.25 V, V_{pp}=12.5 V ±0.3 V, V_{ss}=0 V, T_a=25°C ±5°C)

Item		Symbol	Min	Typ	Max	Unit	Measurement condition
Input high-level voltage	O_0-O_7, A_0-A_{14}, $\overline{OE}$, $\overline{CE}$	V_{IH}	2.2	—	V_{cc} + 0.3	V	
Input low-level voltage	O_0-O_7, A_0-A_{14}, $\overline{OE}$, $\overline{CE}$	V_{IL}	−0.3	—	0.8	V	
Output high-level voltage	O_0-O_7	V_{OH}	2.4	—	—	V	I_{OH}=−200 μA
Output low-level voltage	O_0-O_7	V_{OL}	—	—	0.45	V	I_{OL}=1.6 mA
Input leak current	O_0-O_7, A_0-A_{14}, $\overline{OE}$, $\overline{CE}$	$\lvert I_{LI} \rvert$	—	—	2	μA	V_{in}=5.25 V/0.5 V
V_{cc} current		I_{cc}	—	—	30	mA	
V_{pp} current		I_{pp}	—	—	40	mA	

Table 21-4. AC characteristics
(Unless specified, V_{cc}=6 V ±0.25 V, V_{pp}=12.5 V ±0.3 V, T_a=25°C ±5°C)

Item	Symbol	Min	Typ	Max	Unit	Measurement condition
Address setup time	t_{AS}	2	—	—	μs	Figure 21-3.
OE setup time	t_{OES}	2	—	—	μs	
Data setup time	t_{DS}	2	—	—	μs	
Address hold time	t_{AH}	0	—	—	μs	
Data hold time	t_{DH}	2	—	—	μs	
Data output disable time	t_{DF}	—	—	130	ns	
V_{pp} setup time	t_{VPS}	2	—	—	μs	
Program pulse time	t_{PW}	0.95	1.0	1.05	ms	
CE pulse width at over programming	t_{OPW}	2.85	—	78.75	ms	
V_{cc} setup time	t_{VCS}	2	—	—	μs	
Data output delay time	t_{OE}	0	—	500	ns	

Input pulse level: 0.8 – 2.2 V
Input rising/falling time: $\leq$ 20ns

Timing reference level Input: 1.0 V, 2.0 V
Output: 0.8 V, 2.0 V

21.4 Characteristics of the ZTAT Microcomputer Built-in Programmable ROM and Application Notes.

21.4.1 Write/Erase Principle

The memory cell structure of the ZTAT microcomputer is the same as that of EPROMs. Accordingly, in the same way as ordinary EPROMs, writing is performed by applying high voltage to the control gate and drain, and by injecting hot electrons in to the floating gate (figure 21-4). Trapped by the energy barrier in SiO_2 film, electrons stored in the floating gate stabilize, and the bit becomes 0 due to the threshold voltage change in the memory device. The bit of a memory cell whose floating gate has no electrons is 1.

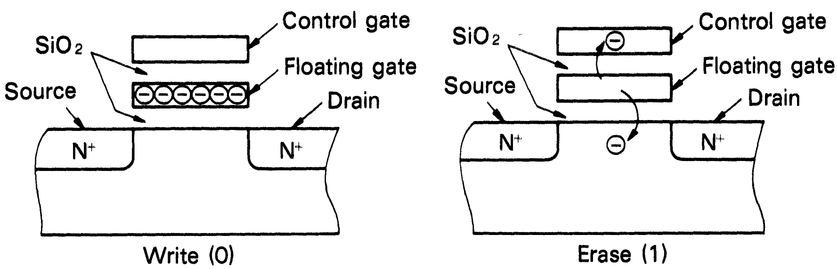


Figure 21-4. PROM Structure

The number of electrons stored in the memory device decreases over time. The causes of electron leakage are as follows:

1. Ultraviolet rays: Electrons are excited and released by ultraviolet rays (the principle of erasure)
2. Heat: Stored electrons escape as a result of thermal excitation
3. Application of high voltage: In some cases, electrons escape due to the high voltage applied to a control gate or drain.

If the oxidized film covering the floating gate has a defect, the loss of electrons becomes noticeable. However, since defective products like this have been eliminated, few or no electrons are lost in the normal memory cell.

21.4.2 Notes on PROM Writing

The higher the program voltage Vpp and the longer the program pulse width t_{PW}, the more electrons are injected, thus ensuring stable writing. However, writing should be performed at a specified voltage and timing. If a voltage greater than the prescribed voltage is applied, the p-n junction can be damaged, inducing permanent breakdown. In particular, care must be taken not to let the PROM writer overshoot. In addition, attention must be paid to negative voltage noise at the terminals since this sometimes induces a parasitic transistor effect which reduces the relative breakdown voltage.

Moreover, since the ZTAT microcomputer is connected to the PROM writer through a socket adapter, the following points should be noted:

1. Before writing, make sure that the socket adapter is correctly attached to the PROM writer.
2. Do not touch the socket adapter and product during writing. This can result in faulty writing due to contact failure.

21.4.3 Reliability after Writing to the Built-In PROM

As a general rule, the reliability of semiconductor products can be assured if faulty devices are identified and rejected early. A screening process is employed to remove initial failures. High-temperature storage is a kind of screeninig whereby the data retention failures in PROM cells can be identified in a short time (refer to 21.4.1 Write/Erase Principle). Since screening is performed during a wafer process for ZTAT microcomputer manufacturing, desirable data retention can be achieved. For further improvements, it is recommended that the user perform high-temperature storage (150°C) after data writing. A recommended screening flow is shown in figure 21-5.

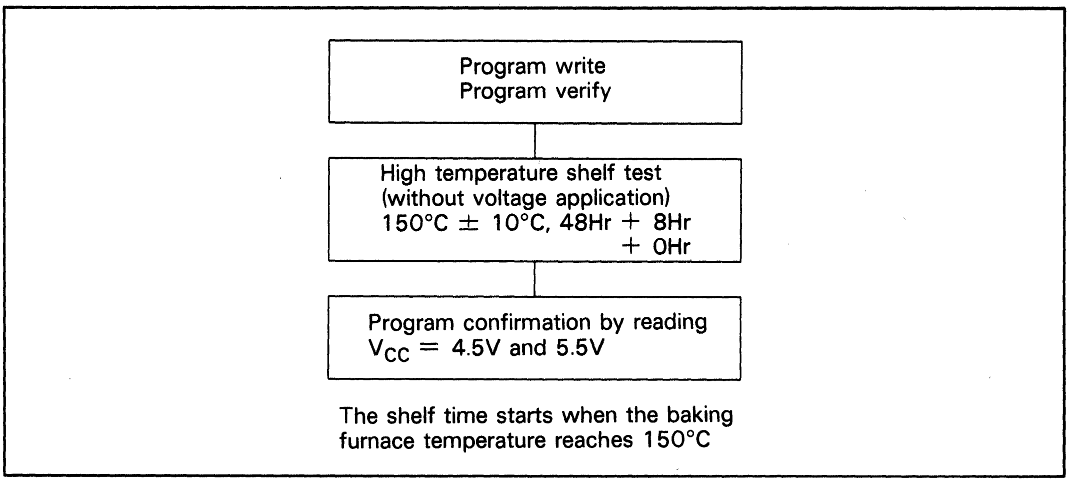


Figure 21-5. Recommended Screening Flow

SECTION 22. HD647180X SOFTWARE ARCHITECTURE

22.1 Instruction Set

The HD647180X is object-code compatible with standard 8-bit operating system and application software. The instruction set also contains a number of new instructions to improve system and software performance, reliability and efficiency (table 22-1).

Table 22-1. Added Instructions

New Instructions	Operation
SLP	Enter sleep mode
MLT	8-bit multiply with 16-bit result
INO g, (m)	Input contents of immediate I/O address into register
OUTO (m), g	Output register contents to immediate I/O address
OTIM	Block output—increment
OTIMR	Block output—increment and repeat
OTDM	Block output—decrement
OTDMR	Block output—decrement and repeat
TSTIO m	Non-destructive AND, I/O port and accumulator
TST g	Non-destructive AND, register and accumulator
TST m	Non-destructive AND, immediate data and accumulator
TST (HL)	Non-destructive AND, memory data and accumulator

22.1.1 SLP—Sleep

The SLP instruction causes the HD647180X to enter sleep low power consumption mode. See section 5 for a complete description of the sleep state.

22.1.2 MLT—Multiply

The MLT performs unsigned multiplication on two 8-bit numbers yielding a 16-bit result. MLT may specify BC, DE, HL, or SP registers. In all cases, the 8-bit operands are loaded into each half of the 16-bit register and the 16-bit result is returned in that register.

22.1.3 INO g, (m)—Input, Immediate I/O Address

The contents of immediately specified 8-bit I/O address are input into the specified register. When I/O is accessed, 00H is output in high-order bits of address automatically.

22.1.4 OUT0 (m), g—Output, Immediate I/O Address

The contents of the specified register are output to the immediately specified 8-bit I/O address. When I/O is accessed, 00H is output in high-order bits of address automatically.

22.1.5 OTIM, OTIMR, OTDM, OTDMR—Block I/O

The contents of memory pointed to by HL are output to the I/O address in (C). The memory address (HL) and I/O address (C) are incremented in OTIM and OTIMR and decremented in OTDM and OTDMR. B register is decremented. The OTIMR and OTDMR variants repeat the above sequence until register B is decremented to 0. Since the I/O address (C) is automatically incremented or decremented, these instructions are useful for block I/O (such as HD647180X on-chip I/O) initialization. When I/O is accessed, 00H is output in high-order bits of address automatically.

22.1.6 TSTIO m—Test I/O Port

The contents of the I/O port addressed by C are ANDed with immediately specified 8-bit data and the status flags are updated. The I/O port contents are not written (non-destructive AND). When I/O is accessed, 00H is output in higher bits of address automatically.

22.1.7 TST g—Test Register

The contents of the specified register are ANDed with the accumulator (A) and the status flags are updated. The accumulator and specified register are not changed (non-destructive AND).

22.1.8 TST m—Test Immediate

The contents of the immediately specified 8-bit data are ANDed with the accumulator (A) and the status flags are updated. The accumulator is not changed (non-destructive AND).

22.1.9 TST (HL)—Test Memory

The contents of memory pointed to by HL are ANDed with the accumulator (A) and the status flags are updated. The memory contents and accumulator are not changed (non-destructive AND).

22.2 CPU Registers

The HD647180X CPU registers consist of register set GR, register set GR' and special registers (figure 22-1).

The register set GR consists of an 8-bit accumulator (A), 8-bit flag register (F), and three general-purpose registers (BC, DE, and HL) which may be treated as 16-bit registers (BC, DE, and HL) or as individual 8-bit registers (B, C, D, E, H, and L) depending on the instruction to be executed. The register set GR' is an alternate register set for register set GR, and also contains an accumulator (A'), flag register (F'), and three general-purpose registers (BC', DE', and HL'). While the alternate register set GR' contents are not directly accessible, the contents can be programmably exchanged at high speed with those of register set GR.

The special registers consist of an 8-bit interrupt vector register (I), an 8-bit R counter (R), two 16-bit index registers (IX and IY), a 16-bit stack pointer (SP), and a 16-bit program counter (PC).

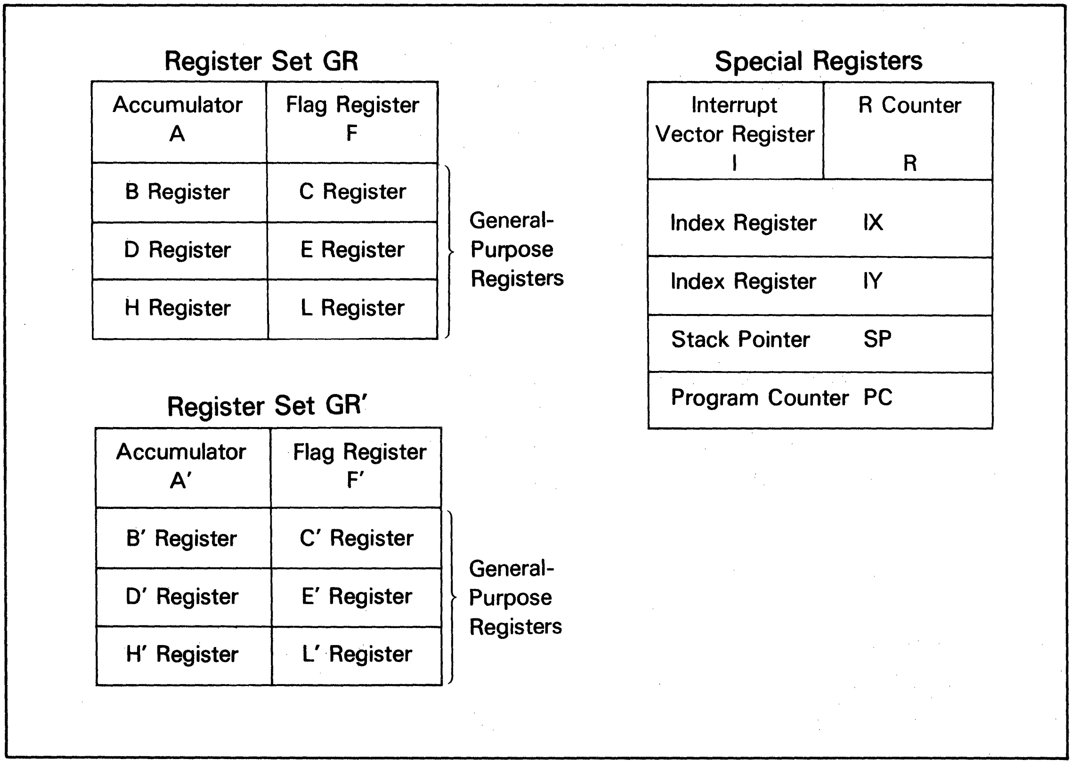


Figure 22-1. CPU Registers

22.2.1 Accumulator (A, A')

The accumulator (A) serves as the primary register used for many arithmetic, logical, and I/O instructions.

22.2.2 Flag Registers (F, F')

The flag register (figure 22-2) stores various status bits which reflect the results of instruction execution. The contents of the flag register are used to control program flow and instruction operation.

bit	7	6	5	4	3	2	1	0	
	S	Z	–	H	–	P/V	N	C	Flag Register (F)

Figure 22-2. Flag Register

S: Sign (Bit 7): S stores the state of the most significant bit (bit 7) of the result. This is useful for operations with signed numbers in which values with bit 7 = 1 are interpreted as negative.

Z: Zero (Bit 6): Z is set to 1 when instruction execution results containing 0. Otherwise, Z is reset to 0.

H: Half Carry (Bit 4): H is used by the DAA (decimal adjust accumulator) instruction to reflect a borrow or carry from the least significant 4 bits and thereby adjust the results of BCD addition and subtraction.

P/V: Parity/Overflow (Bit 2): P/V serves a dual purpose. For logical operations P/V is set to 1 if the number of 1 bit in the result is even and P/V is reset to 0 if the number of 1 bit in the result is odd. For two's complement arithmetic, P/V is set to 1 if the operation produces a result which is outside the allowable range (+127 to −128 for 8-bit operations, +32,767 to −32,768 for 16-bit operations).

N: Negative (Bit 1): N is set to 1 if the last arithmetic instruction was a subtract operation (SUB, DEC, CP, etc.) and N is reset to 0 if the last arithmetic instruction was an addition operation (ADD, INC, etc.).

C: Carry (Bit 0): C is set to 1 when a carry (addition) or borrow (subtraction) from the most significant bit of the result occurs. C is also affected by accumulator logic operations such as shifts and rotates.

22.2.3 General-Purpose Registers (BC, BC', DE, DE', HL, HL')

The general purpose registers are used for both address and data operations. Depending on instruction, each half (8 bits) of these registers (B, C, D, E, H, and L) may also be used.

22.2.4 Interrupt Vector Register (I)

For interrupts which require a vector table address to be calculated ($\overline{INT_0}$ mode 2, $\overline{INT_1}$, $\overline{INT_2}$, and internal interrupts), the interrupt vector register (I) provides the most significant byte of the vector table address. I is cleared to 00H during reset.

22.2.5 R Counter (R)

The least significant seven bits of the R counter (R) serve to count the number of instructions executed by the HD647180X. R is incremented for each CPU opcode fetch cycle (each LIR cycle). R is cleared to 00H during reset.

22.2.6 Index Registers (IX, and IY)

The index registers are used for both address and data operations. For addressing, the contents of a displacement specified in the instruction are added to or subtracted from the index register to determine an effective operand address.

22.2.7 Stack Pointer (SP)

The stack pointer (SP) contains the memory address based LIFO stack. SP is cleared to 0000H during reset.

22.2.8 Program Counter (PC)

The program counter (PC) contains the address of the instruction to be executed and is automatically updated after each instruction fetch. PC is cleared to 0000H during reset.

22.3 Addressing Modes

The HD647180X instruction set includes eight addressing modes:

- Implied register
- Register direct
- Register indirect
- Indexed
- Extended
- Immediate
- Relative
- IO

22.3.1 Implied Register (IMP)

Certain opcodes automatically imply register usage, such as the arithmetic operations which inherently reference the accumulator, index registers, stack pointer, and general-purpose registers.

22.3.2 Register Direct (REG)

Many opcodes contain bit fields specifying registers to be used for the operation. The exact bit field definition vary depending on instruction as in tables 22-2 and 22-3.

Table 22-2. 8-Bit Register Direct Addressing

g or g' Field	Register
0 0 0	B
0 0 1	C
0 1 0	D
0 1 1	E
1 0 0	H
1 0 1	L
1 1 0	—
1 1 1	A

Table 22-3. 16-Bit Register Direct Addressing

zz Field	Register	xx Field	Register
0 0	B C	0 0	B C
0 1	D E	0 1	D E
1 0	H L	1 0	I X
1 1	A F	1 1	S P

ww Field	Register	yy Field	Register
0 0	B C	0 0	B C
0 1	D E	0 1	D E
1 0	H L	1 0	I Y
1 1	S P	1 1	S P

Note: H or L suffixed to ww, xx, yy, zz (ex. wwH,IXL) indicate upper and lower 8 bits of the 16-bit register respectively.

22.3.3 Register Indirect (REG)

The memory operand address is contained in one of the 16-bit general-purpose registers (BC, DE, and HL) (figure 22-3).

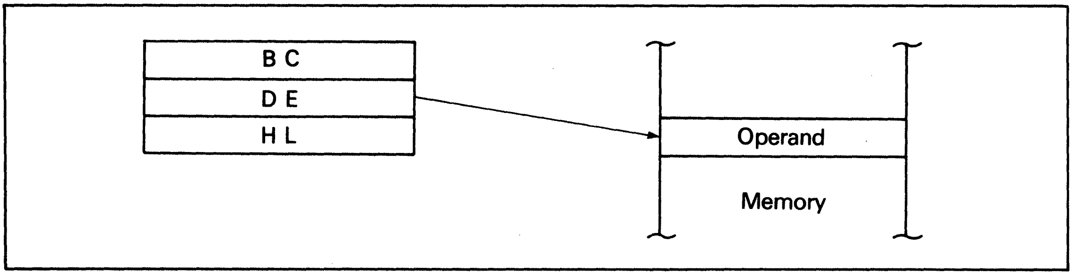


Figure 22-3. Register Indirect Addressing

22.3.4 Indexed (INDX)

The memory operand address is calculated using the contents of an index register (IX or IY) and an 8-bit signed displacement specified in the instruction (figure 22-4).

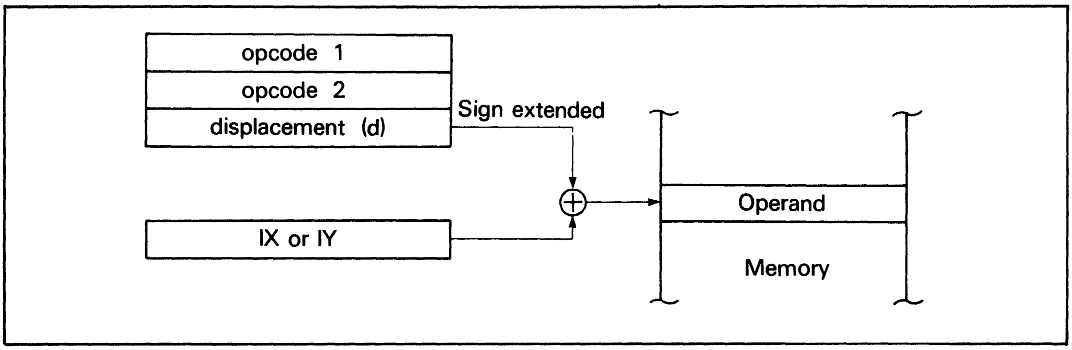


Figure 22-4. Indirect Addressing

22.3.5 Extended (EXT)

The memory operand address is specified by two bytes contained in the instruction (figure 22-5).

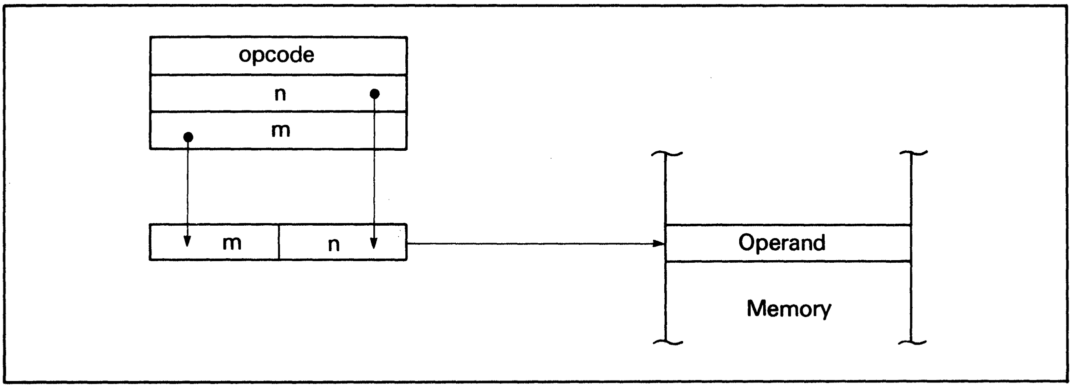


Figure 22-5. Extended Addressing

22.3.6 Immediate (IMMED)

The memory operands are contained within one or two bytes of the instruction (figure 22-6).

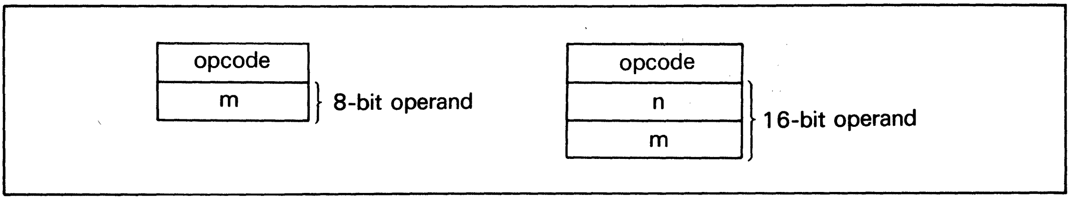


Figure 22-6. Immediate Addressing

22.3.7 Relative (REL)

Relative addressing mode is only used by the conditional and unconditional branch instructions. The branch displacement (relative to the contents of the program counter) is contained in the instruction (figure 22-7).

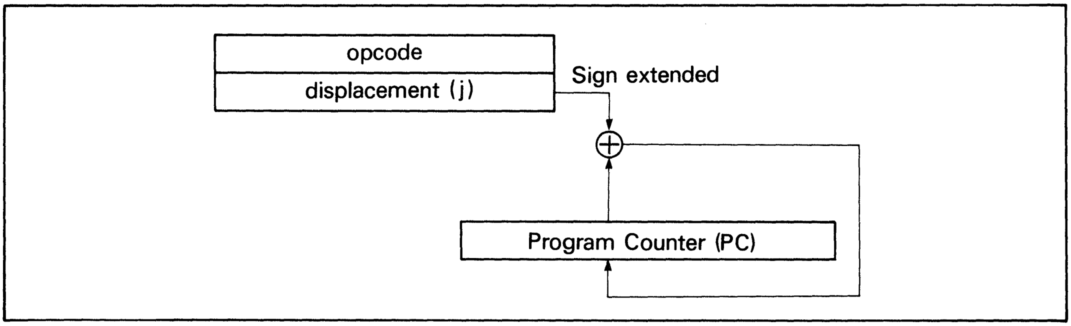


Figure 22-7. Relative Addressing

22.3.8 IO (IO)

IO addressing mode is used only by I/O instructions. This mode specifies I/O address ($\overline{IOE} = 0$) and outputs them as follows:

1. An operand is output to A_0-A_7. The contents of the accumulator is output to A_8-A_{15}.
2. The contents of register B are output to A_0-A_7. The contents of register C are output to A_8-A_{15}.
3. An operand is output to A_0-A_7. 00H is output to A_8-A_{15} (useful for internal I/O register access).
4. The contents of register C are output to A_0-A_7. 00H is output to A_8-A_{15} (useful for internal I/O register access).

SECTION 23. ELECTRICAL CHARACTERISTICS

23.1 Absolute Maximum Ratings

Item	Symbol	Value	Unit
Supply Voltage	V_{CC}	-0.3 to $+7.0$	V
Input Voltage	V_{in}	-0.3 to $V_{CC}+0.3$	V
Operating Temperature	T_{opr}	-20 to $+75$	°C
Storage Temperature	T_{stg}	-55 to $+150$	°C

Note: Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

23.2 DC Characteristics (VCC = 5 V ± 10%, VSS = 0 V, Ta = −20 to +75°C, unless otherwise noted)

Symbol	Item	Min	Typ	Max	Unit	Condition
V_{IH1}	Input High Voltage $\overline{RESET}$, EXTAL, $\overline{NMI}$	$V_{CC}-0.6$	–	$V_{CC}+0.3$	V	
V_{IH2}	Input High Voltage Except $\overline{RESET}$, EXTAL, $\overline{NMI}$	2.0	–	$V_{CC}+0.3$	V	
V_{IL1}	Input Low Voltage $\overline{RESET}$, EXTAL, $\overline{NMI}$	-0.3	–	0.6	V	
V_{IL2}	Input Low Voltage Except $\overline{RESET}$, EXTAL, $\overline{NMI}$	-0.3	–	0.8	V	
V_{OH}	Output High Voltage All outputs	2.4	–	–	V	$I_{OH} = -200\ \mu A$
		$V_{CC}-1.2$	–	–		$I_{OH} = -20\ \mu A$
V_{OL}	Output Low Voltage All Outputs	–	–	0.45	V	$I_{OL} = 2.2$ mA
I_{IL}	Input Leakage Current All Inputs Except XTAL, EXTAL, $\overline{RESET}$	–	–	1.0	μA	Vin=0.5 to $V_{CC} - 0.5$ V
I_{TL}	Three State Leakage Current	–	–	1.0	μA	Vin=0.5 to $V_{CC} - 0.5$ V
I_{CC} (Note)	Power Dissipation (Normal Operation)	–	20	40	mA	f= 4 MHz
		–	25	50		f= 6 MHz
		–	30	60		f= 8 MHz
	Power Dissipation (System Stop Mode)	–	5	10	mA	f= 4 MHz
		–	6.3	12.5		f= 6 MHz
		–	7.5	15		f= 8 MHz
Cp	Pin Capacitance $\overline{RESET}$	–	–	120	pF	Vin=0V, f=1 MHz Ta=25°C
	Except $\overline{RESET}$			20		

Note: $V_{IHmin} = V_{CC} - 1.0$ V, $V_{ILmax} = 0.8$ V (all output terminals are at no load.)

Symbol	Item		Min	Typ	Max	Unit	Condition
V_{IHP}	Input High-Level Voltage		2.2	–	$V_{CC}+0.3$	V	
V_{ILP}	Input Low-Level Voltage		-0.3	–	0.8	V	
V_{OHP}	Output High-Level Voltage		2.4	–	–	V	$I_{OH}=-200\ \mu A$
			$V_{CC}-1.2$	–	–		$I_{OH}=-20\ \mu A$
V_{OLP}	Output Low-Level Voltage		–	–	0.45	V	* $I_{OL}=2.2$ mA
			–	–	1.0		** $I_{OL}=10$ mA
V_{in}	Analog Comparator Input Level Voltage	High level	$V_{ref}+0.1$	–	–	V	
		Low level	–	–	$V_{ref}-0.1$		
V_{ref}		V_{TH}	0	–	$V_{CC}\times 0.8$	V	
I_{ILP}	Input Leak Current		–	–	1.0	μA	$V_{in}=0.5$ to $V_{CC}-0.5$

Note: *: Port A-F
**: Port F only

23.3 AC Characteristics (V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = −20 to +75°C, unless otherwise noted)

		HD647180X-4		HD647180X-6		HD647180X-8		
Symbol	**Item**	**Min**	**Max**	**Min**	**Max**	**Min**	**Max**	**Unit**
t_{cyc}	Clock Cycle Time	250	2000	162	2000	125	2000	ns
t_{CHW}	Clock High Pulse Width	110	—	65	—	50	—	ns
t_{CLW}	Clock Low Pulse Width	110	—	65	—	50	—	ns
t_{cf}	Clock Fall Time	—	15	—	15	—	15	ns
t_{cr}	Clock Rise Time	—	15	—	15	—	15	ns
t_{AD}	Address Delay Time	—	110	—	90	—	80	ns
t_{AS}	Address Set-up Time ($\overline{ME}$ or $\overline{IOE}$ ↓)	50	—	30	—	20	—	ns
t_{MED1}	$\overline{ME}$ Delay Time 1	—	85	—	60	—	50	ns
t_{RDD1}	$\overline{RD}$ Delay Time 1 $\overline{IOC}$=1	—	85	—	60	—	50	ns
	$\overline{IOC}$=0	—	85	—	65	—	60	
t_{LD1}	$\overline{LIR}$ Delay Time 1	—	100	—	80	—	70 (Note)	ns
t_{AH}	Address Hold Time 1 ($\overline{ME}$, $\overline{IOE}$, $\overline{RD}$ or $\overline{WR}$ ↑)	80	—	35	—	20	—	ns
t_{MED2}	$\overline{ME}$ Delay Time 2	—	85	—	60	—	50	ns
t_{RDD2}	$\overline{RD}$ Delay Time 2	—	85	—	60	—	50	ns
t_{LD2}	$\overline{LIR}$ Delay Time 2	—	100	—	80	—	70 (Note)	ns
t_{DRS}	Data Read Set-up Time	50	—	40	—	30	—	ns
t_{DRH}	Data Read Hold Time	0	—	0	—	0	—	ns
t_{STD1}	ST Delay Time 1	—	110	—	90	—	70	ns
t_{STD2}	ST Delay Time 2	—	110	—	90	—	70	ns
t_{WS}	$\overline{WAIT}$ Set-up Time	80	—	40	—	40	—	ns
t_{WH}	$\overline{WAIT}$ Hold Time	70	—	40	—	40	—	ns
t_{WDZ}	Write Data Floating Delay Time	—	100	—	95	—	70	ns
t_{WRD1}	$\overline{WR}$ Delay Time 1	—	90	—	65	—	60	ns
t_{WDD}	Write Data Delay Time	—	110	—	90	—	80	ns
t_{WDS}	Write Data Set-up Time ($\overline{WR}$ ↓)	60	—	40	—	20	—	ns
t_{WRD2}	$\overline{WR}$ Delay Time 2	—	90	—	80	—	60	ns
t_{WRP}	$\overline{WR}$ Pulse Width	280	—	170	—	130	—	ns

Note: For a loading capacitance of less than or equal to 40 picofarads and operating temperature from 0 to 50 degrees, substract 10 nanoseconds from the value given in the maximum columns.

		HD647180X-4		HD647180X-6		HD647180X-8		
Symbol	Item	Min	Max	Min	Max	Min	Max	Unit
t_{WDH}	Write Data Hold Time ($\overline{\text{WR}}$ ↑)	60	—	40	—	15	—	ns
t_{IOD1}	$\overline{\text{IOE}}$ Delay Time 1, $\overline{\text{IOC}}$= 1	—	85	—	60	—	50	ns
	$\overline{\text{IOC}}$= 0	—	85	—	65	—	60	
t_{IOD2}	$\overline{\text{IOE}}$ Delay Time 2	—	85	—	60	—	50	ns
t_{IOD3}	$\overline{\text{IOE}}$ Delay Time 3 ($\overline{\text{LIR}}$ ↓)	540	—	340	—	250	—	ns
t_{INTS}	$\overline{\text{INT}}$ Set-up Time (ϕ ↓)	80	—	40	—	40	—	ns
t_{INTH}	$\overline{\text{INT}}$ Hold Time (ϕ ↓)	70	—	40	—	40	—	ns
t_{NMIW}	$\overline{\text{NMI}}$ Pulse Width	120	—	120	—	100	—	ns
t_{BRS}	$\overline{\text{BUSREQ}}$ Set-up Time (ϕ ↓)	80	—	40	—	40	—	ns
t_{BRH}	$\overline{\text{BUSREQ}}$ Hold Time (ϕ ↓)	70	—	40	—	40	—	ns
t_{BAD1}	$\overline{\text{BUSACK}}$ Delay Time 1	—	100	—	95	—	70	ns
t_{BAD2}	$\overline{\text{BUSACK}}$ Delay Time 2	—	100	—	95	—	70	ns
t_{BZD}	Bus Floating Delay Time	—	130	—	125	—	90	ns
t_{MEWH}	$\overline{\text{ME}}$ Pulse Width (HIGH)	200	—	110	—	90	—	ns
t_{MEWL}	$\overline{\text{ME}}$ Pulse Width (LOW)	210	—	125	—	100	—	ns
t_{RFD1}	$\overline{\text{REF}}$ Delay Time 1	—	110	—	90	—	80	ns
t_{RFD2}	$\overline{\text{REF}}$ Delay Time 2	—	110	—	90	—	80	ns
t_{HAD1}	$\overline{\text{HALT}}$ Delay Time 1	—	110	—	90	—	80	ns
t_{HAD2}	$\overline{\text{HALT}}$ Delay Time 2	—	110	—	90	—	80	ns
t_{DRQS}	$\overline{\text{DREQi}}$ Set-up Time	80	—	40	—	40	—	ns
t_{DRQH}	$\overline{\text{DREQi}}$ Hold Time	70	—	40	—	40	—	ns
t_{TED1}	$\overline{\text{TENDi}}$ Delay Time 1	—	85	—	70	—	60	ns
t_{TED2}	$\overline{\text{TENDi}}$ Delay Time 2	—	85	—	70	—	60	ns
t_{ED1}	Enable Delay Time 1	—	100	—	95	—	70	ns
t_{ED2}	Enable Delay Time 2	—	100	—	95	—	70	ns
P_{WEH}	E Pulse Width (HIGH)	150	—	75	—	65	—	ns
P_{WEL}	E Pulse Width (LOW)	300	—	180	—	130	—	ns

Symbol	Item	HD647180X-4 Min	HD647180X-4 Max	HD647180X-6 Min	HD647180X-6 Max	HD647180X-8 Min	HD647180X-8 Max	Unit
t_{Er}	Enable Rise Time	—	25	—	20	—	20	ns
t_{Ef}	Enable Fall Time	—	25	—	20	—	20	ns
t_{TOD}	Timer Output Delay Time	—	300	—	300	—	200	ns
t_{STDI}	CSI/O Transmit Data Delay Time (Internal Clock Operation)	—	200	—	200	—	200	ns
t_{STDE}	CSI/O Transmit Data Delay Time (External Clock Operation)	—	7.5tcyc +300	—	7.5tcyc +300	—	7.5tcyc +200	ns
t_{SRSI}	CSI/O Receive Data Set-up Time (Internal Clock Operation)	1	—	1	—	1	—	tcyc
t_{SRHI}	CSI/O Receive Data Hold Time (Internal Clock Operation)	1	—	1	—	1	—	tcyc
t_{SRSE}	CSI/O Receive Data Set-up Time (External Clock Operation)	1	—	1	—	1	—	tcyc
t_{SRHE}	CSI/O Receive Data Hold Time (External Clock Operation)	1	—	1	—	1	—	tcyc
t_{RES}	$\overline{\text{RESET}}$ Set-up Time	120	—	120	—	100	—	ns
t_{REH}	$\overline{\text{RESET}}$ Hold Time	80	—	80	—	70	—	ns
t_{OSC}	Oscillator Stabilization Time	—	20	—	20	—	20	ms
t_{EXr}	External Clock Rise Time (EXTAL)	—	25	—	25	—	25	ns
t_{EXf}	External Clock Fall Time (EXTAL)	—	25	—	25	—	25	ns
t_{Rr}	$\overline{\text{RESET}}$ Rise Time	—	50	—	50	—	50	ms
t_{Rf}	$\overline{\text{RESET}}$ Fall Time	—	50	—	50	—	50	ms
t_{Ir}	Input Rise Time (except EXTAL, $\overline{\text{RESET}}$)	—	100	—	100	—	100	ns
t_{If}	Input Fall Time (except EXTAL, $\overline{\text{RESET}}$)	—	100	—	100	—	100	ns
t_{PWD}	Port Data Output Delay Time	—	110	—	90	—	80	ns
t_{PDSU}	Port Data Input Setup Time	80	—	50	—	50	—	ns
t_{PDH}	Port Data Input Hold Time	60	—	40	—	40	—	ns

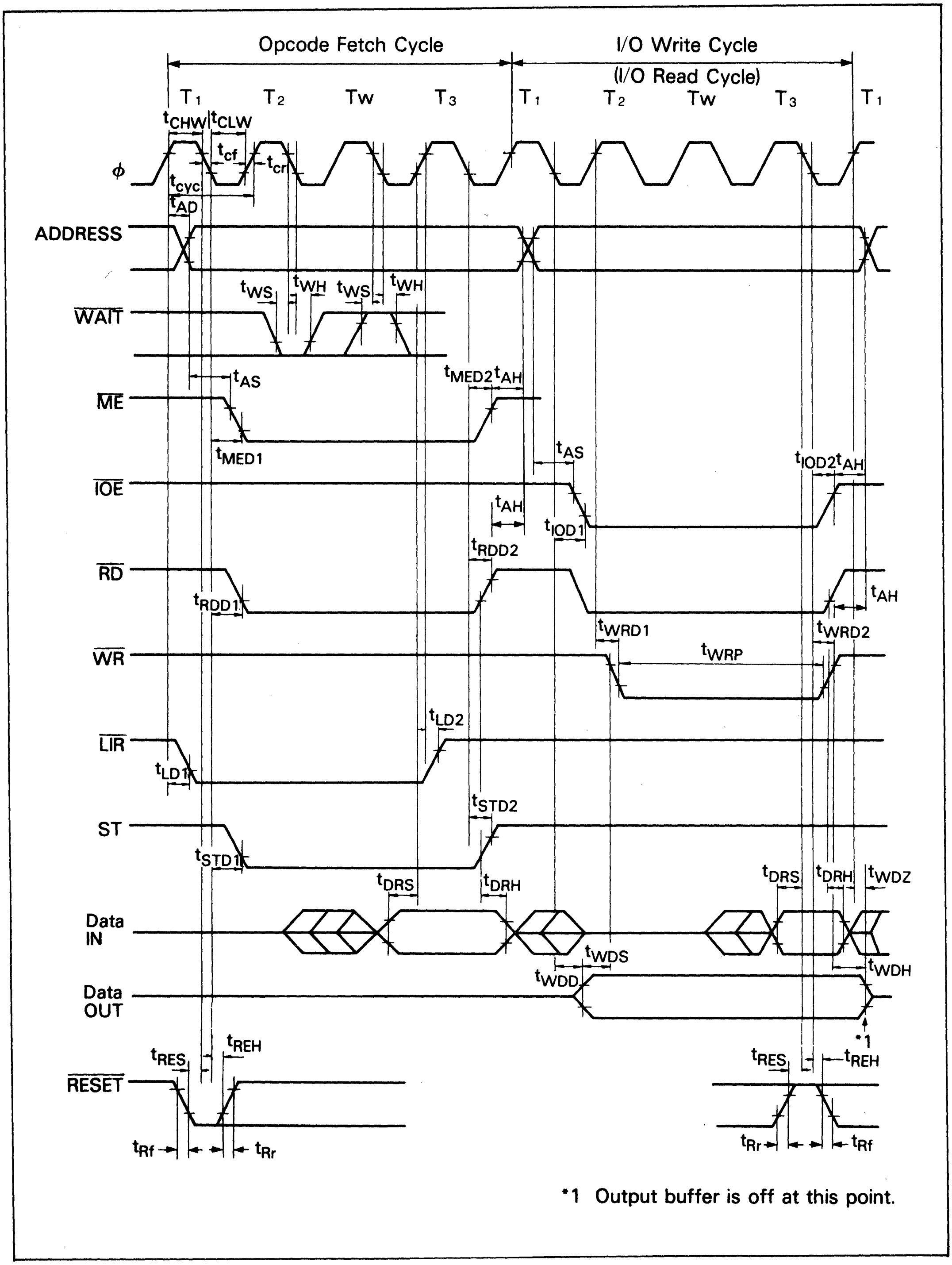


Figure 23-1. CPU Timing (Opcode Fetch Cycle, I/O Write Cycle, I/O Read Cycle)

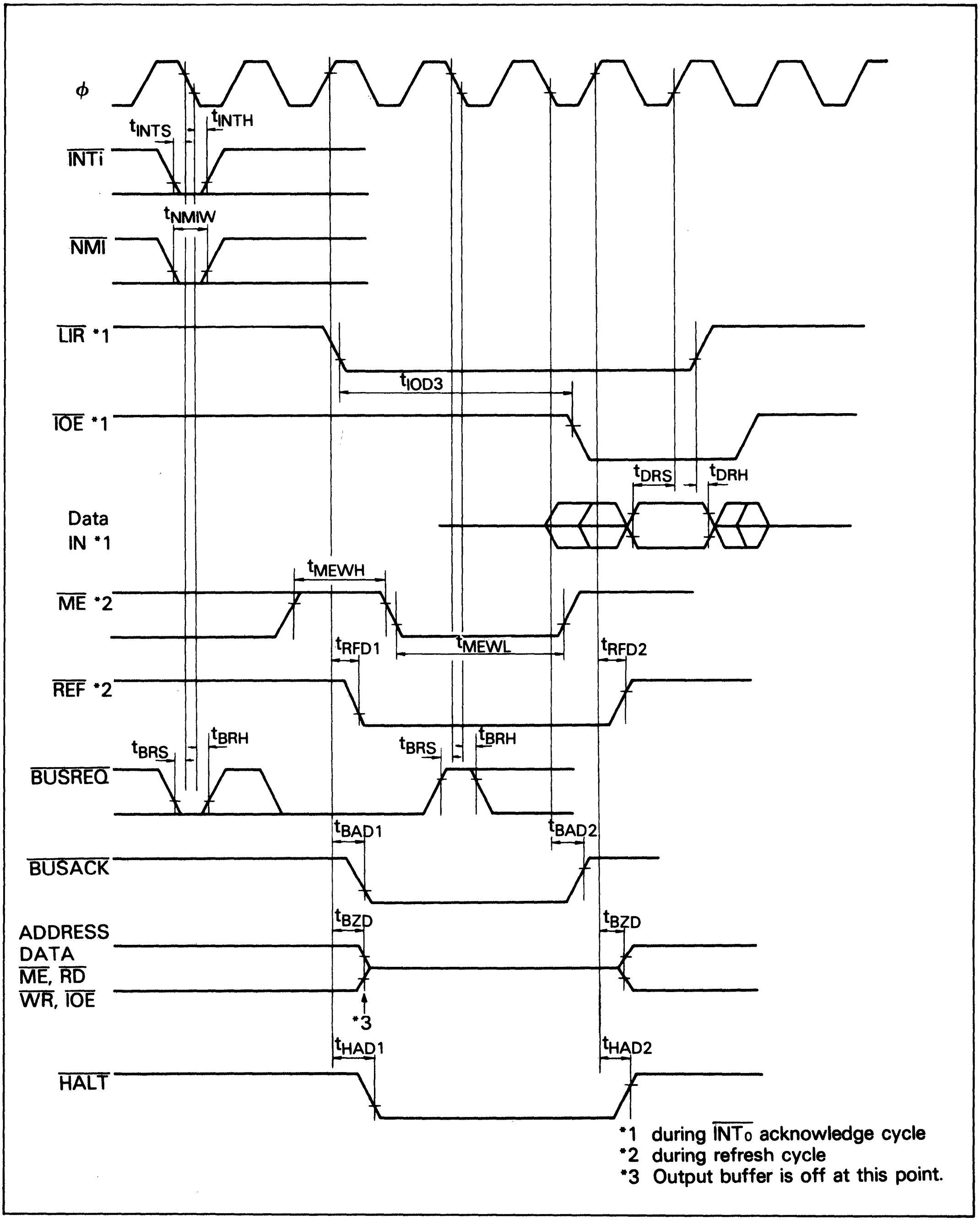


Figure 23-2. CPU Timing ($\overline{INT_0}$ Acknowledge Cycle, Refresh Cycle, Bus Release Mode, Halt Mode, Sleep Mode, System Stop Mode)

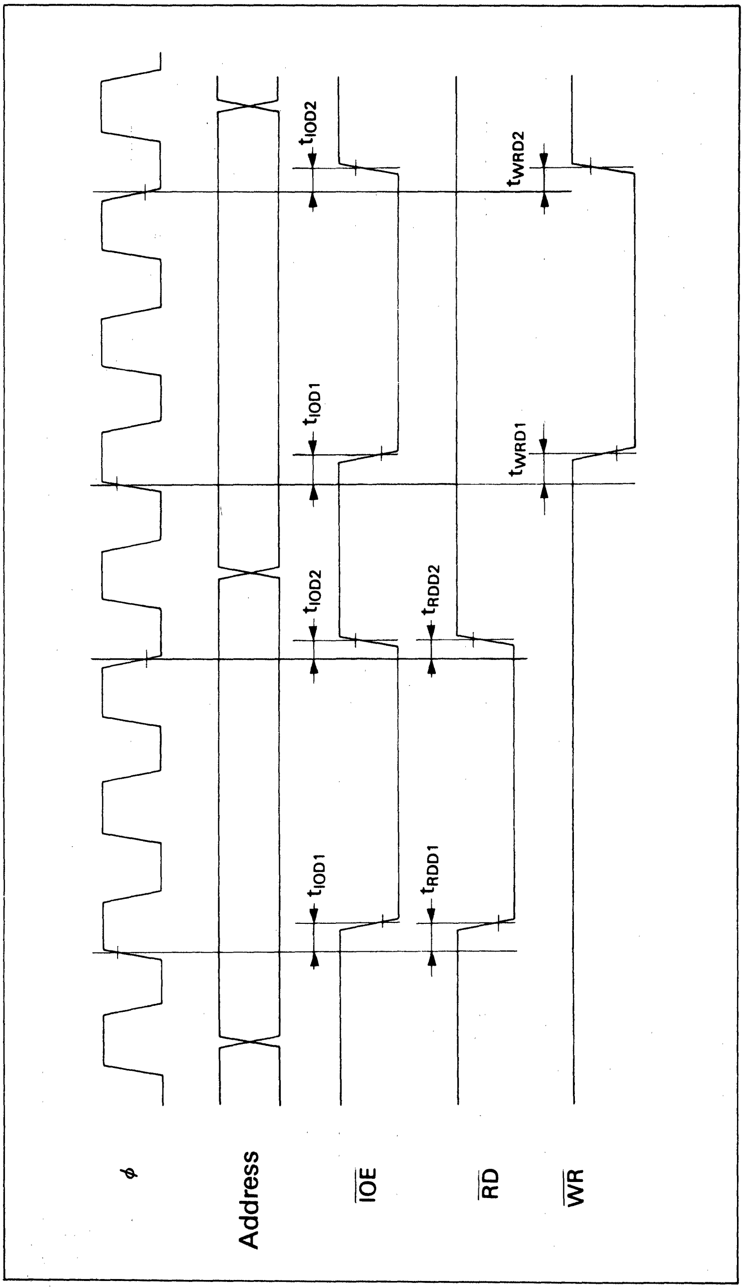
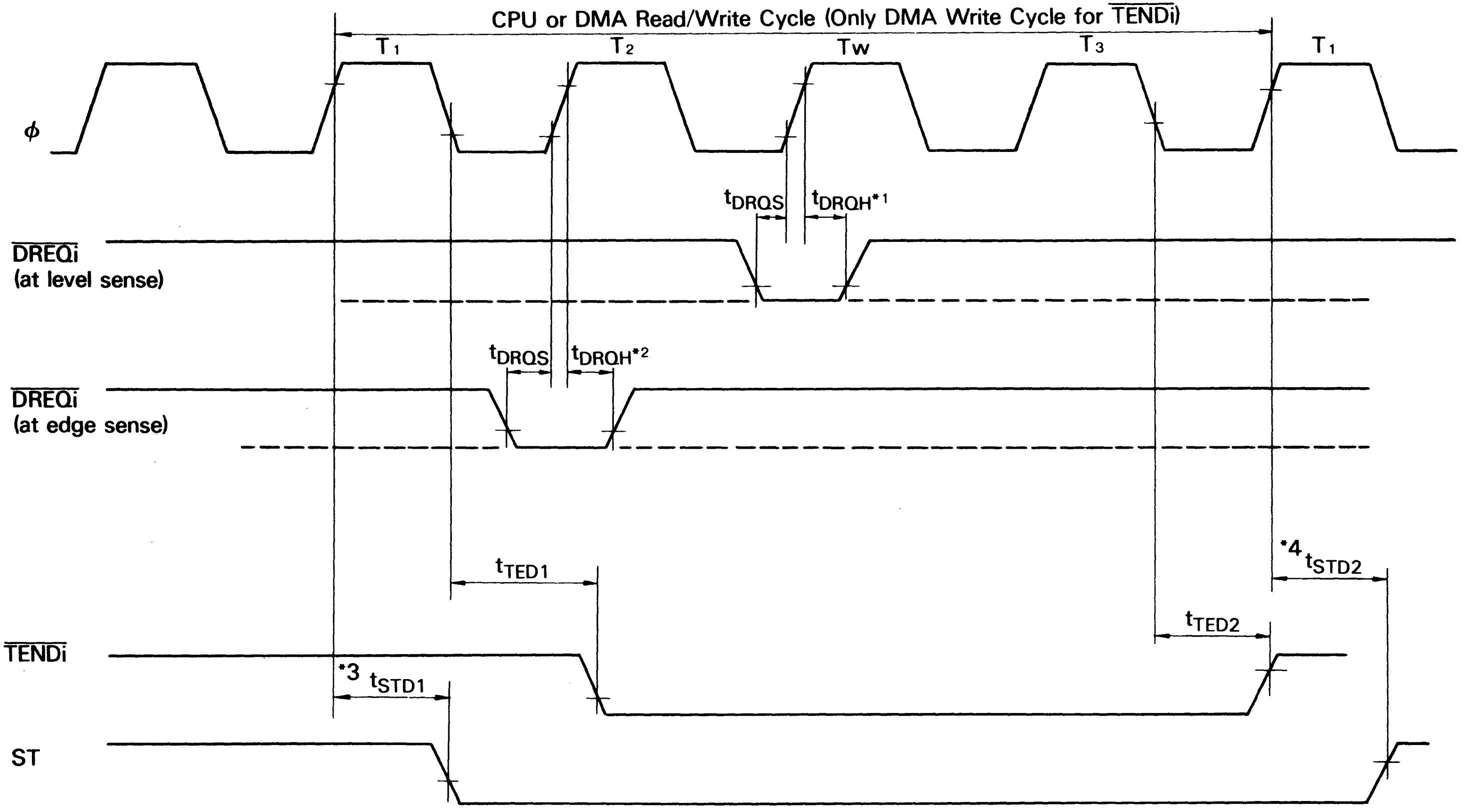


Figure 23-3. CPU Timing ($\overline{IOC}$=0)



*1 t_{DRQS} and t_{DRQH} are specified for the rising edge of clock followed by T_3.
*2 t_{DRQS} and t_{DRQH} are specified for the rising edge of clock.
*3 DMA cycle starts.
*4 CPU cycle starts.

Figure 23-4. DMA Control Signals

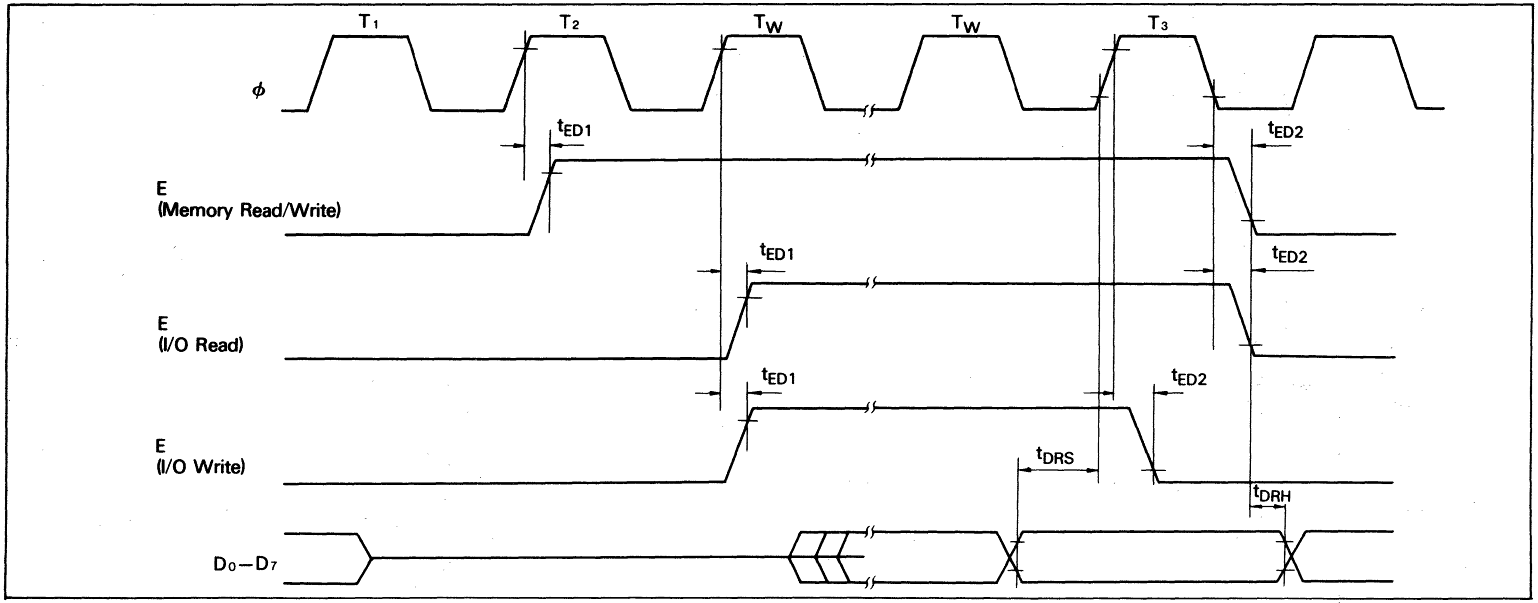


Figure 23-5. E Clock Timing (Memory Read/Write Cycle, I/O Read/Write Cycle)

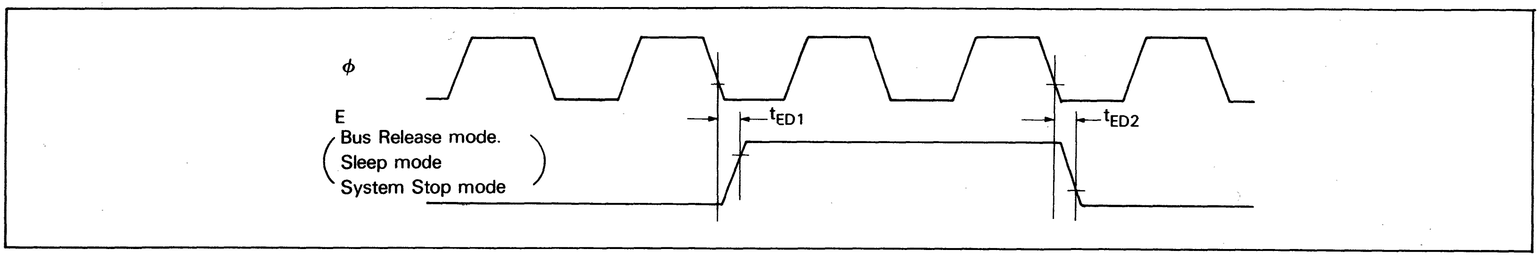


Figure 23-6. E Clock Timing (Bus Release Mode, Sleep Mode, System Stop Mode)

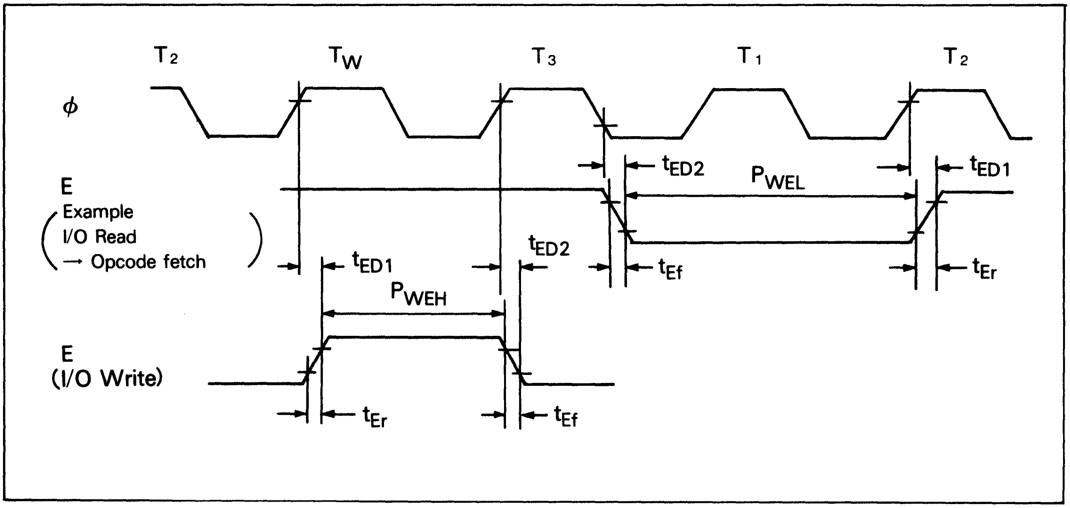


Figure 23-7. E Clock Timing (Minimum Timing Example of P_{WEL} and P_{WEH})

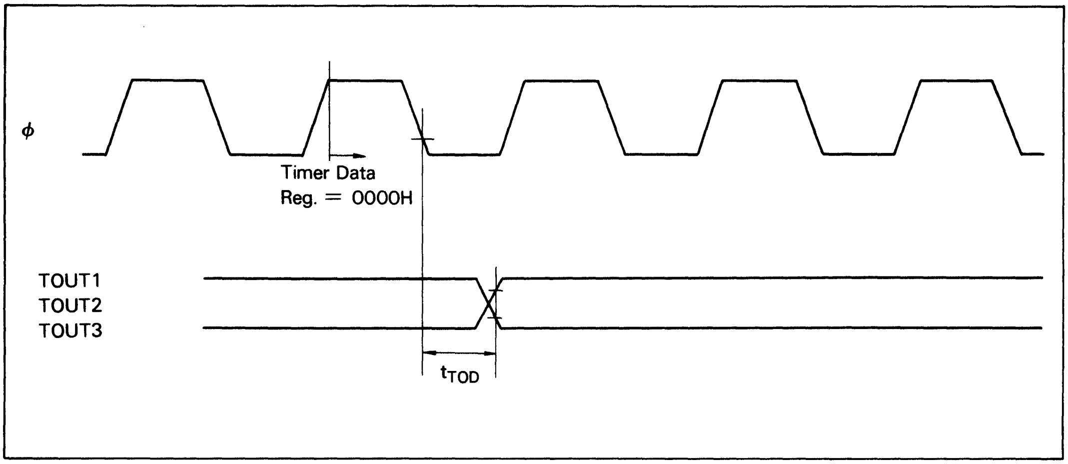


Figure 23-8. Timer Output Timing

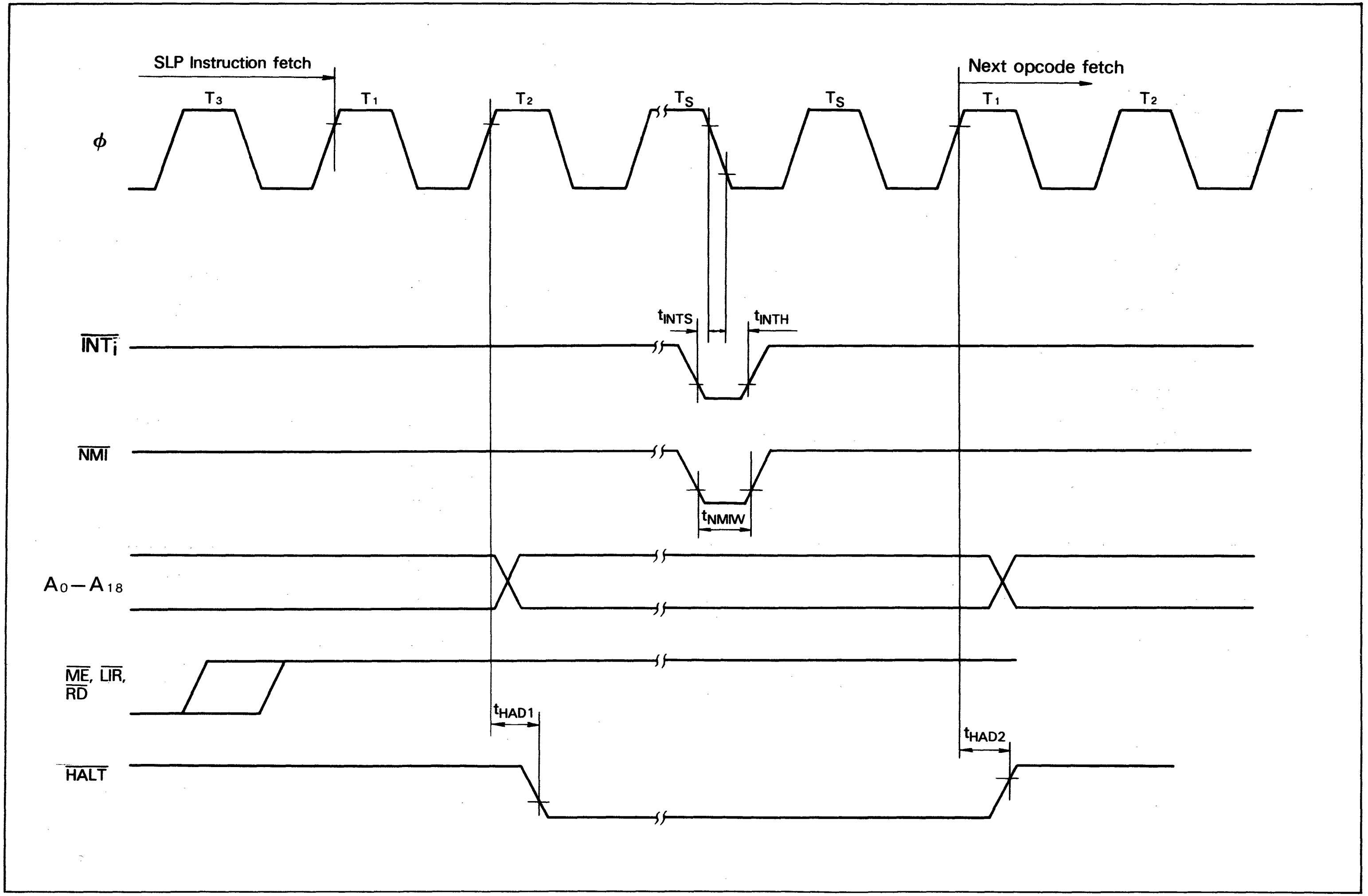


Figure 23-9. SLP Execution Cycle

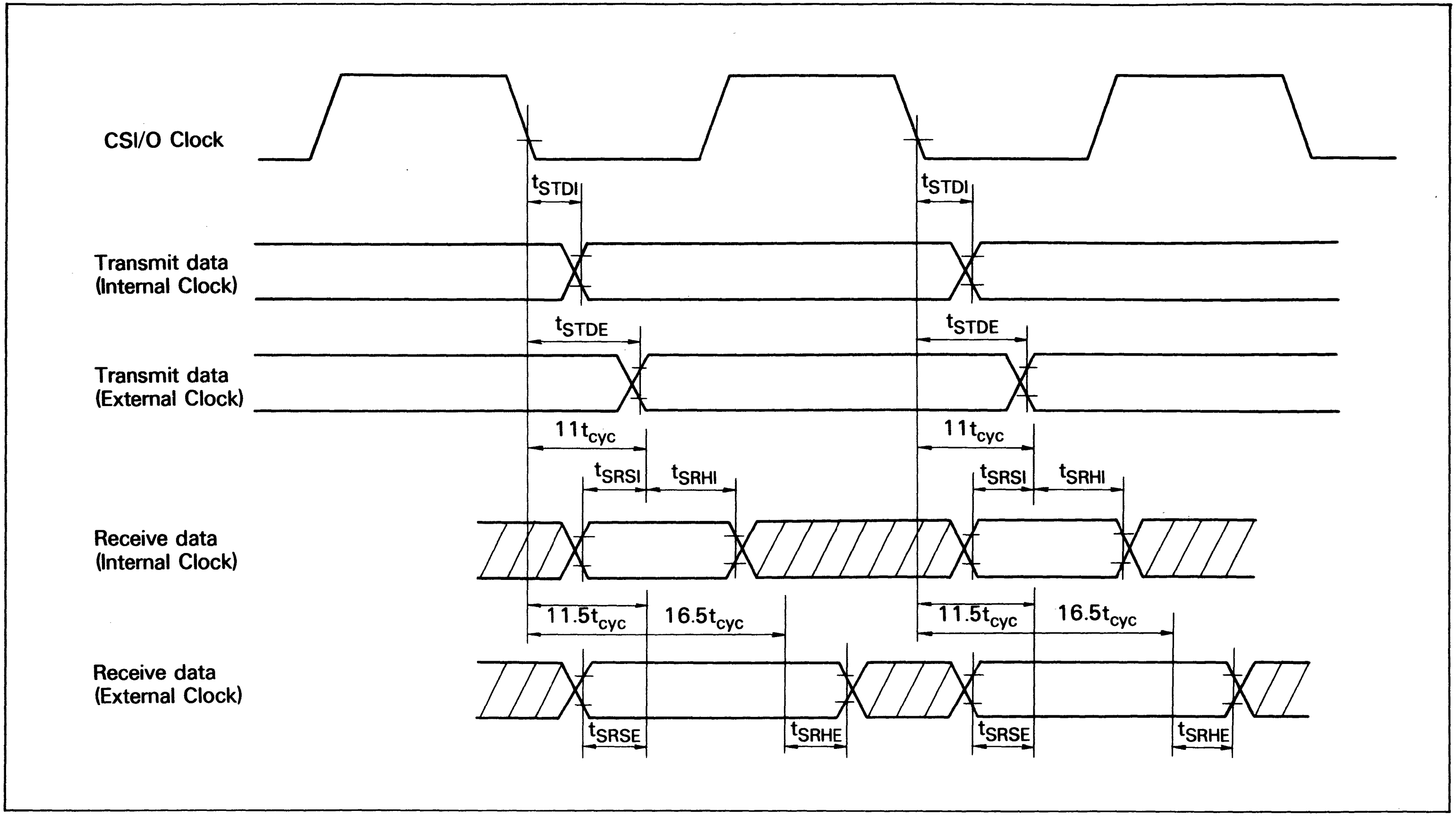


Figure 23-10. CSI/O Receive/Transmit Timing

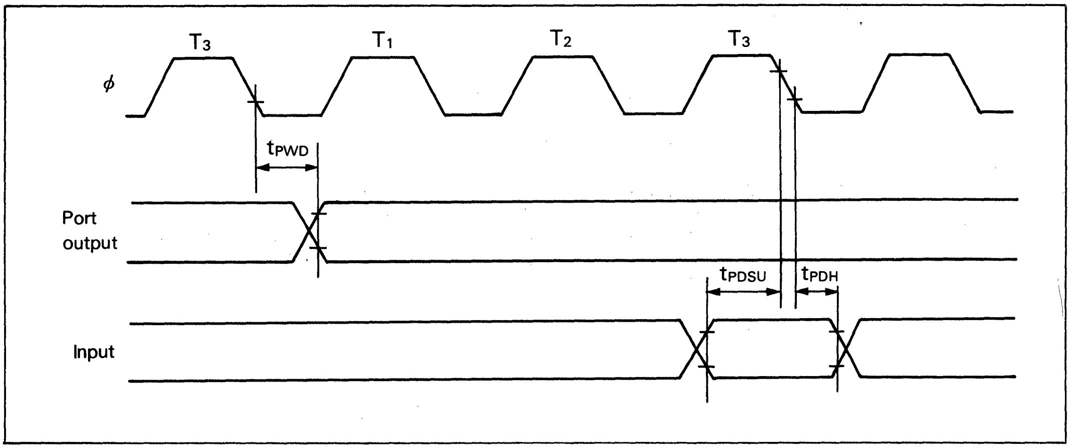


Figure 23-11. Port Input and Output Timing

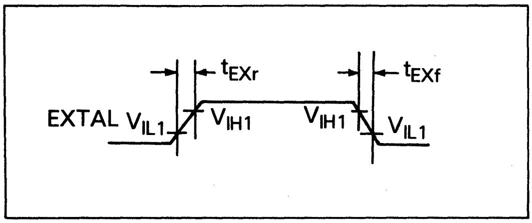


Figure 23-12. External Clock Rise Time and Fall Time

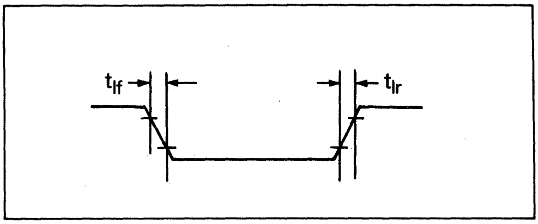


Figure 23-13. Input Rise Time and Fall Time (Except EXTAL, $\overline{\text{RESET}}$)

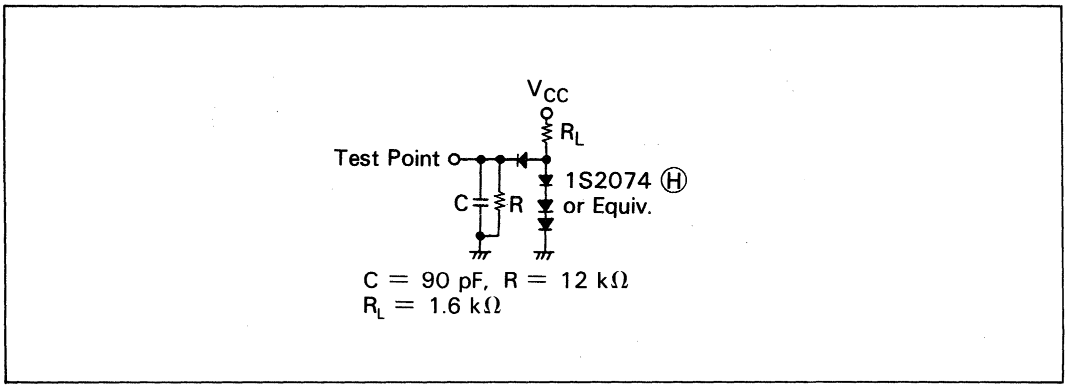


Figure 23-14. Bus Timing Test Load (TTL Load)

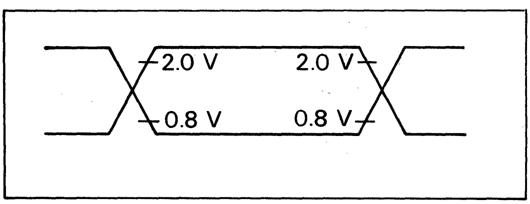


Figure 23-15. Reference Level (Input)

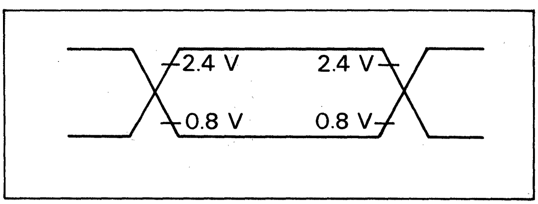
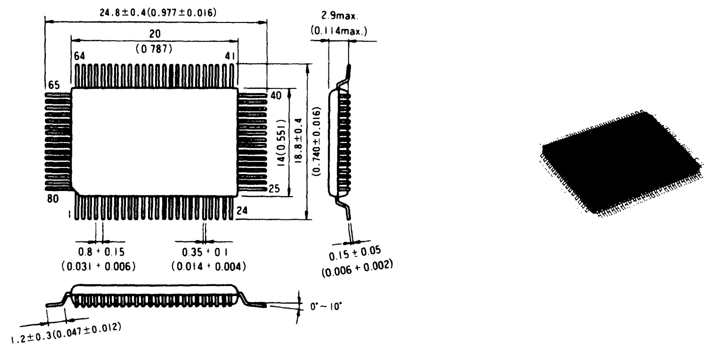


Figure 23-16. Reference Level (Output)

SECTION 24. HD647180X PACKAGE DIMENSIONS

Unit: mm (inch)

FP-80B



CP-84

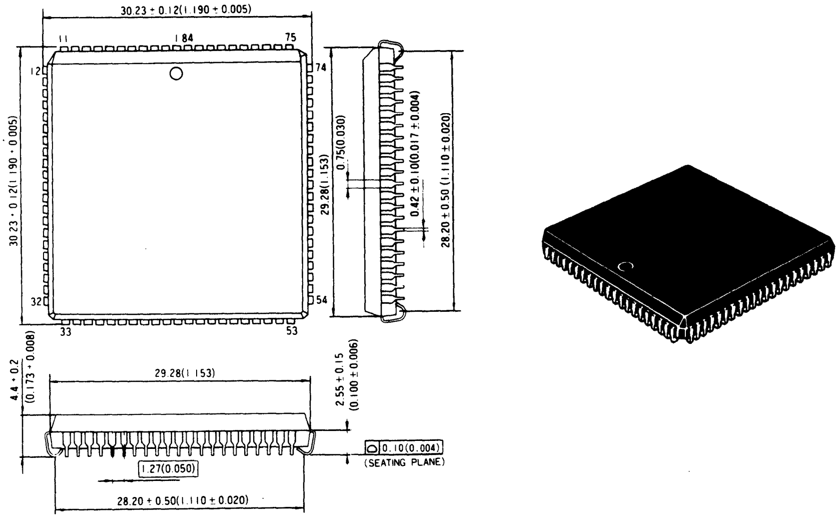


Figure 24-1. Package Dimensions

APPENDIX

APPENDIX A. INSTRUCTION SET

A.1 Symbols

The following explains the symbols in instruction set.

A.1.1 Register

g, g', ww, xx, yy, and zz specify a register to be used. g and g' specify an 8-bit register. ww, xx, yy, and zz specify a 16-bit pair of 8-bit registers. Table A-1 shows the correspondence between symbols and registers.

Table A-1 Register Specification

g,g'	Reg.
000	B
001	C
010	D
011	E
100	H
101	L
111	A

ww	Reg.	xx	Reg.	yy	Reg.	zz	Reg.
00	BC	00	BC	00	BC	00	BC
01	DE	01	DE	01	DE	01	DE
10	HL	10	IX	10	IY	10	HL
11	SP	11	SP	11	SP	11	AF

Note: H and L suffixed to ww,xx,yy,zz (ex. wwH, IXL) indicate upper and lower 8 bits of the 16-bit register, respectively.

A.1.2 Bit

b specifies a bit to be manipulated in the bit manipulation instruction. Table A-2 shows the correspondence between b and bits.

Table A-2 Bit Specification

b	Bit
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

A.1.3 Condition

f specifies the condition in program control instructions. Table A-3 shows the correspondence between f and conditions.

Table A-3 Condition Specification

f	Condition	
000	NZ	non zero
001	Z	zero
010	NC	non carry
011	C	carry
100	PO	parity odd
101	PE	parity even
110	P	sign plus
111	M	sign minus

A.1.4 Restart Address

v specifies a restart address. Table A-4 shows the correspondence between v and restart addresses.

Table A-4 Restart Address Specification

v	Address
000	00H
001	08H
010	10H
011	18H
100	20H
101	28H
110	30H
111	38H

A.1.5 Flag

The following symbols show the flag conditions:

- · : not affected
- ↕ : affected
- × : undefined
- S : set to 1
- R : reset to 0
- P : parity
- V : overflow

A.1.6 Miscellaneous

$(\quad)_M$: Data in the memory address
$(\quad)_I$: Data in the I/O address
m or n : 8-bit data
mn : 16-bit data
r : 8-bit register
R : 16-bit register
b·$(\quad)_M$: Contents of bit b in the memory address
b·gr : Contents of bit b in the register gr
d or j : 8-bit signed displacement
S : Source addressing mode
D : Destination addressing mode
· : AND operation
+ : OR operation
+ : EXCLUSIVE OR operation
** : Added new instructions to Z80

A.2 Instruction Summary

A.2.1 Data Manipulation Instructions

Table A-5 Arithmetic and Logical Instructions (8 Bit)

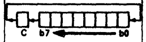
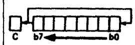
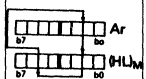
Operation Name	Mnemonics	Opcode	Addressing IMMED	EXT	IND	REG	REGI	IMP	REL	Bytes	States	Operation	Flag 7 S	6 Z	4 H	2 P/V	1 N	0 C
ADD	ADD A,g	10 000 g				S		D		1	4	Ar+gr→Ar	↕	↕	↕	V	R	↕
	ADD A,(HL)	10 000 110					S	D		1	6	Ar+(HL)$_M$→Ar	↕	↕	↕	V	R	↕
	ADD A,m	11 000 110 ⟨ m ⟩	S					D		2	6	Ar+m→Ar	↕	↕	↕	V	R	↕
	ADD A,(IX+d)	11 011 101 10 000 110 ⟨ d ⟩			S			D		3	14	Ar+(IX+d)$_M$→Ar	↕	↕	↕	V	R	↕
	ADD A,(IY+d)	11 111 101 10 000 110 ⟨ d ⟩			S			D		3	14	Ar+(IY+d)$_M$→Ar	↕	↕	↕	V	R	↕
ADC	ADC A,g	10 001 g				S		D		1	4	Ar+gr+c→Ar	↕	↕	↕	V	R	↕
	ADC A,(HL)	10 001 110					S	D		1	6	Ar+(HL)$_M$+c→Ar	↕	↕	↕	V	R	↕
	ADC A,m	11 001 110 ⟨ m ⟩	S					D		2	6	Ar+m+c→Ar	↕	↕	↕	V	R	↕
	ADC A,(IX+d)	11 011 101 10 001 110 ⟨ d ⟩			S			D		3	14	Ar+(IX+d)$_M$+c→Ar	↕	↕	↕	V	R	↕
	ADC A,(IY+d)	11 111 101 10 001 110 ⟨ d ⟩			S			D		3	14	Ar+(IY+d)$_M$+c→Ar	↕	↕	↕	V	R	↕
AND	AND g	10 100 g				S		D		1	4	Ar·gr→Ar	↕	↕	S	P	R	R
	AND (HL)	10 100 110					S	D		1	6	Ar·(HL)$_M$→Ar	↕	↕	S	P	R	R
	AND m	11 100 110 ⟨ m ⟩	S					D		2	6	Ar·m→Ar	↕	↕	S	P	R	R
	AND (IX+d)	11 011 101 10 100 110 ⟨ d ⟩			S			D		3	14	Ar·(IX+d)$_M$→Ar	↕	↕	S	P	R	R
	AND (IY+d)	11 111 101 10 100 110 ⟨ d ⟩			S			D		3	14	Ar·(IY+d)$_M$→Ar	↕	↕	S	P	R	R
Compare	CP g	10 111 g				S		D		1	4	Ar−gr	↕	↕	↕	V	S	↕
	CP (HL)	10 111 110					S	D		1	6	Ar−(HL)$_M$	↕	↕	↕	V	S	↕
	CP m	11 111 110 ⟨ m ⟩	S					D		2	6	Ar−m	↕	↕	↕	V	S	↕
	CP (IX+d)	11 011 101 10 111 110 ⟨ d ⟩			S			D		3	14	Ar−(IX+d)$_M$	↕	↕	↕	V	S	↕
	CP (IY+d)	11 111 101 10 111 110 ⟨ d ⟩			S			D		3	14	Ar−(IY+d)$_M$	↕	↕	↕	V	S	↕
Comple-ment	CPL	00 101 111						S/D		1	3	$\overline{\text{Ar}}$→Ar	·	·	S	·	S	·
DEC	DEC g	00 g 101				S/D				1	4	gr−1→gr	↕	↕	↕	V	S	·
	DEC (HL)	00 110 101					S/D			1	10	(HL)$_M$−1→(HL)$_M$	↕	↕	↕	V	S	·
	DEC (IX+d)	11 011 101 00 110 101 ⟨ d ⟩			S/D					3	18	(IX+d)$_M$−1→ (IX+d)$_M$	↕	↕	↕	V	S	·
	DEC (IY+d)	11 111 101 00 110 101 ⟨ d ⟩			S/D					3	18	(IY+d)$_M$−1→ (IY+d)$_M$	↕	↕	↕	V	S	·
INC	INC g	00 g 100				S/D				1	4	gr+1→gr	↕	↕	↕	V	R	·
	INC (HL)	00 110 100					S/D			1	10	(HL)$_M$+1→(HL)$_M$	↕	↕	↕	V	R	·
	INC (IX+d)	11 011 101 00 110 100 ⟨ d ⟩			S/D					3	18	(IX+d)$_M$+1→ (IX+d)$_M$	↕	↕	↕	V	R	·
	INC (IY+d)	11 111 101 00 110 100 ⟨ d ⟩			S/D					3	18	(IY+d)$_M$+1→ (IY+d)$_M$	↕	↕	↕	V	R	·

(continued)

Table A-5 Arithmetic and Logical Instructions (8 Bit) (cont)

Operation Name	Mnemonics	Opcode	Addressing							Bytes	States	Operation	Flag					
			IMMED	EXT	IND	REG	REGI	IMP	REL				7 S	6 Z	4 H	2 P/V	1 N	0 C
MULT	MLT ww **	11 101 101 01 ww1 100				S/D				2	17	$wwHr \times wwLr \rightarrow ww_r$	•	•	•	•	•	•
Negate	NEG	11 101 101 01 000 100						S/D		2	6	0−Ar→Ar	↕	↕	↕	V	S	↕
OR	OR g	10 110 g				S		D		1	4	Ar+gr→Ar	↕	↕	R	P	R	R
	OR (HL)	10 110 110					S	D		1	6	Ar+(HL)$_M$→Ar	↕	↕	R	P	R	R
	OR m	11 110 110 ⟨ m ⟩	S					D		2	6	Ar+m→Ar	↕	↕	R	P	R	R
	OR (IX+d)	11 011 101 10 110 110 ⟨ d ⟩			S			D		3	14	Ar+(IX+d)$_M$→Ar	↕	↕	R	P	R	R
	OR (IY+d)	11 111 101 10 110 110 ⟨ d ⟩			S			D		3	14	Ar+(IY+d)$_M$→Ar	↕	↕	R	P	R	R
SUB	SUB g	10 010 g				S		D		1	4	Ar−gr→Ar	↕	↕	↕	V	S	↕
	SUB (HL)	10 010 110					S	D		1	6	Ar−(HL)$_M$→Ar	↕	↕	↕	V	S	↕
	SUB m	11 010 110 ⟨ m ⟩	S					D		2	6	Ar−m→Ar	↕	↕	↕	V	S	↕
	SUB (IX+d)	11 011 101 10 010 110 ⟨ d ⟩			S			D		3	14	Ar−(IX+d)$_M$→Ar	↕	↕	↕	V	S	↕
	SUB (IY+d)	11 111 101 10 010 110 ⟨ d ⟩			S			D		3	14	Ar−(IY+d)$_M$→Ar	↕	↕	↕	V	S	↕
SUBC	SBC A,g	10 011 g				S		D		1	4	Ar−gr−c→Ar	↕	↕	↕	V	S	↕
	SBC A,(HL)	10 011 110					S	D		1	6	Ar−(HL)$_M$−c→Ar	↕	↕	↕	V	S	↕
	SBC A,m	11 011 110 ⟨ m ⟩	S					D		2	6	Ar−m−c→Ar	↕	↕	↕	V	S	↕
	SBC A,(IX+d)	11 011 101 10 011 110 ⟨ d ⟩			S			D		3	14	Ar−(IX+d)$_M$−c→Ar	↕	↕	↕	V	S	↕
	SBC A,(IY+d)	11 111 101 10 011 110 ⟨ d ⟩			S			D		3	14	Ar−(IY+d)$_M$−c→Ar	↕	↕	↕	V	S	↕
Test	TST g **	11 101 101 00 g 100				S				2	7	Ar·gr	↕	↕	S	P	R	R
	TST (HL) **	11 101 101 00 110 100					S			2	10	Ar·(HL)$_M$	↕	↕	S	P	R	R
	TST m **	11 101 101 01 100 100 ⟨ m ⟩	S							3	9	Ar·m	↕	↕	S	P	R	R
XOR	XOR g	10 101 g				S		D		1	4	Ar⊕gr→Ar	↕	↕	R	P	R	R
	XOR (HL)	10 101 110					S	D		1	6	Ar⊕(HL)$_M$→Ar	↕	↕	R	P	R	R
	XOR m	11 101 110 ⟨ m ⟩	S					D		2	6	Ar⊕m→Ar	↕	↕	R	P	R	R
	XOR (IX+d)	11 011 101 10 101 110 ⟨ d ⟩			S			D		3	14	Ar⊕(IX+d)$_M$→Ar	↕	↕	R	P	R	R
	XOR (IY+d)	11 111 101 10 101 110 ⟨ d ⟩			S			D		3	14	Ar⊕(IY+d)$_M$→Ar	↕	↕	R	P	R	R

Table A-6 Rotate and Shift Instructions

Operation Name	Mnemonics	Opcode	Addressing							Bytes	States	Operation	Flag					
													7	6	4	2	1	0
			IMMED	EXT	IND	REG	REGI	IMP	REL				S	Z	H	P/V	N	C
Rotate	RLA	00 010 111						S/D		1	3		·	·	R	·	R	↕
and	RL g	11 001 011 00 010 g				S/D				2	7		↕	↕	R	P	R	↕
Shift Data	RL (HL)	11 001 011 00 010 110					S/D			2	13		↕	↕	R	P	R	↕
	RL (IX+d)	11 011 101 11 001 011 〈 d 〉 00 010 110			S/D					4	19		↕	↕	R	P	R	↕
	RL (IY+d)	11 111 101 11 001 011 〈 d 〉 00 010 110			S/D					4	19		↕	↕	R	P	R	↕
	RLCA	00 000 111						S/D		1	3		·	·	R	·	R	↕
	RLC g	11 001 011 00 000 g				S/D				2	7		↕	↕	R	P	R	↕
	RLC (HL)	11 001 011 00 000 110					S/D			2	13		↕	↕	R	P	R	↕
	RLC (IX+d)	11 011 101 11 001 011 〈 d 〉 00 000 110			S/D					4	19		↕	↕	R	P	R	↕
	RLC (IY+d)	11 111 101 11 001 011 〈 d 〉 00 000 110			S/D					4	19		↕	↕	R	P	R	↕
	RLD	11 101 101 01 101 111						S/D		2	16		↕	↕	R	P	R	·
	RRA	00 011 111						S/D		1	3		·	·	R	·	R	↕
	RR g	11 001 011 00 011 g				S/D				2	7		↕	↕	R	P	R	↕
	RR (HL)	11 001 011 00 011 110					S/D			2	13		↕	↕	R	P	R	↕
	RR (IX+d)	11 011 101 11 001 011 〈 d 〉 00 011 110			S/D					4	19		↕	↕	R	P	R	↕
	RR (IY+d)	11 111 101 11 001 011 〈 d 〉 00 011 110			S/D					4	19		↕	↕	R	P	R	↕
	RRCA	00 001 111						S/D		1	3		·	·	R	·	R	↕
	RRC g	11 001 011 00 001 g				S/D				2	7		↕	↕	R	P	R	↕
	RRC (HL)	11 001 011 00 001 110					S/D			2	13		↕	↕	R	P	R	↕
	RRC (IX+d)	11 011 101 11 001 011 〈 d 〉 00 001 110			S/D					4	19		↕	↕	R	P	R	↕
	RRC (IY+d)	11 111 101 11 001 011 〈 d 〉 00 001 110			S/D					4	19		↕	↕	R	P	R	↕

(continued)

Table A-6 Rotate and Shift Instructions (cont)

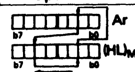
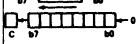
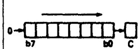
Operation Name	Mnemonics	Opcode	Addressing: IMMED	EXT	IND	REG	REGI	IMP	REL	Bytes	States	Operation	Flag 7 S	6 Z	4 H	2 P/V	1 N	0 C
Rotate and Shift Data	RRD	11 101 101 01 100 111						S/D		2	16		↕	↕	R	P	R	·
	SLA g	11 001 011 00 100 g				S/D				2	7		↕	↕	R	P	R	↕
	SLA (HL)	11 001 011 00 100 110					S/D			2	13		↕	↕	R	P	R	↕
	SLA (IX+d)	11 011 101 11 001 011 < d > 00 100 110			S/D					4	19		↕	↕	R	P	R	↕
	SLA (IY+d)	11 111 101 11 001 011 < d > 00 100 110			S/D					4	19		↕	↕	R	P	R	↕
	SRA g	11 001 011 00 101 g				S/D				2	7		↕	↕	R	P	R	↕
	SRA (HL)	11 001 011 00 101 110					S/D			2	13		↕	↕	R	P	R	↕
	SRA (IX+d)	11 011 101 11 001 011 < d > 00 101 110			S/D					4	19		↕	↕	R	P	R	↕
	SRA (IY+d)	11 111 101 11 001 011 < d > 00 101 110			S/D					4	19		↕	↕	R	P	R	↕
	SRL g	11 001 011 00 111 g				S/D				2	7		↕	↕	R	P	R	↕
	SRL (HL)	11 001 011 00 111 110					S/D			2	3		↕	↕	R	P	R	↕
	SRL (IX+d)	11 011 101 11 001 011 < d > 00 111 110			S/D					4	19		↕	↕	R	P	R	↕
	SRL (IY+d)	11 111 101 11 001 011 < d > 00 111 110			S/D					4	19		↕	↕	R	P	R	↕

Table A-7 Bit Manipulation Instructions

Operation Name	Mnemonics	Opcode	Addressing							Bytes	States	Operation	Flag					
													7	6	4	2	1	0
			IMMED	EXT	IND	REG	REGI	IMP	REL				S	Z	H	P/V	N	C
Bit Set	SET b,g	11 001 011 11 b g				S/D				2	7	$1 \rightarrow b \cdot gr$	·	·	·	·	·	·
	SET b,(HL)	11 001 011 11 b 110					S/D			2	13	$1 \rightarrow b \cdot (HL)_M$	·	·	·	·	·	·
	SET b,(IX+d)	11 011 101 11 001 011 ‹ d › 11 b 110			S/D					4	19	$1 \rightarrow b \cdot (IX+d)_M$	·	·	·	·	·	·
	SET b,(IY+d)	11 111 101 11 001 011 ‹ d › 11 b 110			S/D					4	19	$1 \rightarrow b \cdot (IY+d)_M$	·	·	·	·	·	·
Bit Reset	RES b,g	11 001 011 10 b g				S/D				2	7	$0 \rightarrow b \cdot gr$	·	·	·	·	·	·
	RES b,(HL)	11 001 011 10 b 110					S/D			2	13	$0 \rightarrow b \cdot (HL)_M$	·	·	·	·	·	·
	RES b,(IX+d)	11 011 101 11 001 011 ‹ d › 10 b 110			S/D					4	19	$0 \rightarrow b \cdot (IX+d)_M$	·	·	·	·	·	·
	RES b,(IY+d)	11 111 101 11 001 011 ‹ d › 10 b 110			S/D					4	19	$0 \rightarrow b \cdot (IY+d)_M$	·	·	·	·	·	·
Bit Test	BIT b,g	11 001 011 01 b g				S				2	6	$\overline{b \cdot gr} \rightarrow z$	X	↕	S	X	R	·
	BIT b,(HL)	11 001 011 01 b 110					S			2	9	$\overline{b \cdot (HL)_M} \rightarrow z$	X	↕	S	X	R	·
	BIT b,(IX+d)	11 011 101 11 001 011 ‹ d › 01 b 110			S					4	15	$\overline{b \cdot (IX+d)_M} \rightarrow z$	X	↕	S	X	R	·
	BIT b,(IY+d)	11 111 101 11 001 011 ‹ d › 01 b 110			S					4	15	$\overline{b \cdot (IY+d)_M} \rightarrow z$	X	↕	S	X	R	·

Table A-8 Arithmetic Instructions (16 Bit)

			Addressing										Flag					
													7	6	4	2	1	0
Operation Name	Mnemonics	Opcode	IMMED	EXT	IND	REG	REGI	IMP	REL	Bytes	States	Operation	S	Z	H	P/V	N	C
ADD	ADD HL,ww	00 ww1 001				S		D		1	7	$HL_R+ww_R \rightarrow HL_R$	·	·	X	·	R	↕
	ADD IX,xx	11 011 101				S		D		2	10	$IX_R+xx_R \rightarrow IX_R$	·	·	X	·	R	↕
		00 xx1 001																
	ADD IY,yy	11 111 101				S		D		2	10	$IY_R+yy_R \rightarrow IY_R$	·	·	X	·	R	↕
		00 yy1 001																
ADC	ADC HL,ww	11 101 101				S		D		2	10	$HL_R+ww_R+c \rightarrow HL_R$	↕	↕	X	V	R	↕
		01 ww1 010																
DEC	DEC ww	00 ww1 011				S/D				1	4	$ww_R-1 \rightarrow ww_R$	·	·	·	·	·	·
	DEC IX	11 011 101						S/D		2	7	$IX_R-1 \rightarrow IX_R$	·	·	·	·	·	·
		00 101 011																
	DEC IY	11 111 101						S/D		2	7	$IY_R-1 \rightarrow IY_R$	·	·	·	·	·	·
		00 101 011																
INC	INC ww	00 ww0 011				S/D				1	4	$ww_R+1 \rightarrow ww_R$	·	·	·	·	·	·
	INC IX	11 011 101						S/D		2	7	$IX_R+1 \rightarrow IX_R$	·	·	·	·	·	·
		00 100 011																
	INC IY	11 111 101						S/D		2	7	$IY_R+1 \rightarrow IY_R$	·	·	·	·	·	·
		00 100 011																
SBC	SBC HL,ww	11 101 101				S		D		2	10	$HL_R-ww_R-c \rightarrow HL_R$	↕	↕	X	V	S	↕
		01 ww0 010																

A.2.2 Data Transfer Instructions

Table A-9 8-Bit Load

Operation Name	Mnemonics	Opcode	Addressing							Bytes	States	Operation	Flag					
													7	6	4	2	1	0
			IMMED	EXT	IND	REG	REGI	IMP	REL				S	Z	H	P/V	N	C
Load 8-Bit Data	LD A,I	11 101 101 01 010 111						S/D		2	6	Ir→Ar	$\updownarrow$	$\updownarrow$	R	IEF_2	R	·
	LD A,R	11 101 101 01 011 111						S/D		2	6	Rr→Ar	$\updownarrow$	$\updownarrow$	R	IEF_2	R	·
	LD A,(BC)	00 001 010					S	D		1	6	$(BC)_M \to Ar$ (Note 1)	·	·	·	·	·	·
	LD A,(DE)	00 011 010					S	D		1	6	$(DE)_M \to Ar$	·	·	·	·	·	·
	LD A,(mn)	00 111 010 ⟨ n ⟩ ⟨ m ⟩		S				D		3	12	$(mn)_M \to Ar$	·	·	·	·	·	·
	LD I,A	11 101 101 01 000 111						S/D		2	6	Ar→Ir	·	·	·	·	·	·
	LD R,A	11 101 101 01 001 111						S/D		2	6	Ar→Rr	·	·	·	·	·	·
	LD (BC),A	00 000 010					D	S		1	7	$Ar \to (BC)_M$	·	·	·	·	·	·
	LD (DE),A	00 010 010					D	S		1	7	$Ar \to (DE)_M$	·	·	·	·	·	·
	LD (mn),A	00 110 010 ⟨ n ⟩ ⟨ m ⟩		D				S		3	13	$Ar \to (mn)_M$	·	·	·	·	·	·
	LD g,g'	01 g g'				S/D				1	4	gr'→gr	·	·	·	·	·	·
	LD g,(HL)	01 g 110				D	S			1	6	$(HL)_M \to gr$	·	·	·	·	·	·
	LD g,m	00 g 110 ⟨ m ⟩	S			D				2	6	m→gr	·	·	·	·	·	·
	LD g,(IX+d)	11 011 101 01 g 110 ⟨ d ⟩			S	D				3	14	$(IX+d)_M \to gr$	·	·	·	·	·	·
	LD g,(IY+d)	11 111 101 01 g 110 ⟨ d ⟩			S	D				3	14	$(IY+d)_M \to gr$	·	·	·	·	·	·
	LD (HL),m	00 110 110 ⟨ m ⟩	S				D			2	9	$m \to (HL)_M$	·	·	·	·	·	·
	LD (IX+d),m	11 011 101 00 110 110 ⟨ d ⟩ ⟨ m ⟩	S		D					4	15	$m \to (IX+d)_M$	·	·	·	·	·	·
	LD (IY+d),m	11 111 101 00 110 110 ⟨ d ⟩ ⟨ m ⟩	S		D					4	15	$m \to (IY+d)_M$	·	·	·	·	·	·
	LD (HL),g	01 110 g				S	D			1	7	$gr \to (HL)_M$	·	·	·	·	·	·
	LD (IX+d),g	11 011 101 01 110 g ⟨ d ⟩			D	S				3	15	$gr \to (IX+d)_M$	·	·	·	·	·	·
	LD (IY+d),g	11 111 101 01 110 g ⟨ d ⟩			D	S				3	15	$gr \to (IY+d)_M$	·	·	·	·	·	·

Note: 1 Interrupts are not sampled at the end of LD A, I or LD A, R.

Table A-10 16-Bit Load

Operation Name	Mnemonics	Opcode	Addressing							Bytes	States	Operation	Flag					
													7	6	4	2	1	0
			IMMED	EXT	IND	REG	REGI	IMP	REL				S	Z	H	P/V	N	C
Load 16-Bit Data	LD ww,mn	00 ww0 001 ⟨ n ⟩ ⟨ m ⟩	S			D				3	9	$mn \rightarrow ww_R$	·	·	·	·	·	·
	LD IX,mn	11 011 101 00 100 001 ⟨ n ⟩ ⟨ m ⟩	S					D		4	12	$mn \rightarrow IX_R$	·	·	·	·	·	·
	LD IY,mn	11 111 101 00 100 001 ⟨ n ⟩ ⟨ m ⟩	S					D		4	12	$mn \rightarrow IY_R$	·	·	·	·	·	·
	LD SP,HL	11 111 001						S/D		1	4	$HL_R \rightarrow SP_R$	·	·	·	·	·	·
	LD SP,IX	11 011 101 11 111 001						S/D		2	7	$IX_R \rightarrow SP_R$	·	·	·	·	·	·
	LD SP,IY	11 111 101 11 111 001						S/D		2	7	$IY_R \rightarrow SP_R$	·	·	·	·	·	·
	LD ww,(mn)	11 101 101 01 ww1 011 ⟨ n ⟩ ⟨ m ⟩		S		D				4	18	$(mn+1)_M \rightarrow wwHr$ $(mn)_M \rightarrow wwLr$	·	·	·	·	·	·
	LD HL,(mn)	00 101 010 ⟨ n ⟩ ⟨ m ⟩		S				D		3	15	$(mn+1)_M \rightarrow Hr$ $(mn)_M \rightarrow Lr$	·	·	·	·	·	·
	LD IX,(mn)	11 011 101 00 101 010 ⟨ n ⟩ ⟨ m ⟩		S				D		4	18	$(mn+1)_M \rightarrow IXHr$ $(mn)_M \rightarrow IXLr$	·	·	·	·	·	·
	LD IY,(mn)	11 111 101 00 101 010 ⟨ n ⟩ ⟨ m ⟩		S				D		4	18	$(mn+1)_M \rightarrow IYHr$ $(mn)_M \rightarrow IYLr$	·	·	·	·	·	·
	LD (mn),ww	11 101 101 01 ww0 011 ⟨ n ⟩ ⟨ m ⟩		D		S				4	19	$wwHr \rightarrow (mn+1)_M$ $wwLr \rightarrow (mn)_M$	·	·	·	·	·	·
	LD (mn),HL	00 100 010 ⟨ n ⟩ ⟨ m ⟩		D				S		3	16	$Hr \rightarrow (mn+1)_M$ $Lr \rightarrow (mn)_M$	·	·	·	·	·	·
	LD (mn),IX	11 011 101 00 100 010 ⟨ n ⟩ ⟨ m ⟩		D				S		4	19	$IXHr \rightarrow (mn+1)_M$ $IXLr \rightarrow (mn)_M$	·	·	·	·	·	·
	LD (mn),IY	11 111 101 00 100 010 ⟨ n ⟩ ⟨ m ⟩		D				S		4	19	$IYHr \rightarrow (mn+1)_M$ $IYLr \rightarrow (mn)_M$	·	·	·	·	·	·

Table A-11 Block Transfer

Operation Name	Mnemonics	Opcode	Addressing: IMMED	EXT	IND	REG	REGI	IMP	REL	Bytes	States	Operation	Flag 7 S	6 Z	4 H	2 P/V	1 N	0 C
Block Transfer Search Data	CPD	11 101 101 10 101 001					S	S		2	12	$Ar-(HL)_M$ $BC_R-1 \rightarrow BC_R$ $HL_R-1 \rightarrow HL_R$	↕	↕[3]	↕	↕[2]	S	·
	CPDR	11 101 101 10 111 001					S	S		2	14 12	$BC_R \neq 0$ $Ar \neq (HL)_M$ $BC_R=0$ or $Ar=(HL)_M$ Q { $Ar-(HL)_M$ $BC_R-1 \rightarrow BC_R$ $HL_R-1 \rightarrow HL_R$ } Repeat Q until $Ar=(HL)_M$ or $BC_R=0$	↕	↕[3]	↕	↕[2]	S	·
	CPI	11 101 101 10 100 001					S	S		2	12	$Ar-(HL)_M$ $BC_R-1 \rightarrow BC_R$ $HL_R+1 \rightarrow HL_R$	↕	↕[3]	↕	↕[2]	S	·
	CPIR	11 101 101 10 110 001					S	S		2	14 12	$BC_R \neq 0$ $Ar \neq (HL)_M$ $BC_R=0$ or $Ar=(HL)_M$ Q { $Ar-(HL)_M$ $BC_R-1 \rightarrow BC_R$ $HL_R+1 \rightarrow HL_R$ } Repeat Q until $Ar=(HL)_M$ or $BC_R=0$	↕	↕[3]	↕	↕[2]	S	·
	LDD	11 101 101 10 101 000					S/D			2	12	$(HL)_M \rightarrow (DE)_M$ $BC_R-1 \rightarrow BC_R$ $DE_R-1 \rightarrow DE_R$ $HL_R-1 \rightarrow HL_R$	·	·	R	↕[2]	R	·
	LDDR	11 101 101 10 111 000					S/D			2	14 ($BC_R \neq 0$) 12 ($BC_R=0$)	Q { $(HL)_M \rightarrow (DE)_M$ $BC_R-1 \rightarrow BC_R$ $DE_R-1 \rightarrow DE_R$ $HL_R-1 \rightarrow HL_R$ } Repeat Q until $BC_R=0$	·	·	R	R	R	·
	LDI	11 101 101 10 100 000					S/D			2	12	$(HL)_M \rightarrow (DE)_M$ $BC_R-1 \rightarrow BC_R$ $DE_R+1 \rightarrow DE_R$ $HL_R+1 \rightarrow HL_R$	·	·	R	↕[2]	R	·
	LDIR	11 101 101 10 110 000					S/D			2	14 ($BC_R \neq 0$) 12 ($BC_R=0$)	Q { $(HL)_M \rightarrow (DE)_M$ $BC_R-1 \rightarrow BC_R$ $DE_R+1 \rightarrow DE_R$ $HL_R+1 \rightarrow HL_R$ } Repeat Q until $BC_R=0$	·	·	R	R	R	·

Note: 2 $P/V = 0$: $BC_R - 1 = 0$
$P/V = 1$: $BC_R - 1 \neq 0$
3 $Z = 1$: $Ar = (HL)_M$
$Z = 0$: $Ar \neq (HL)_M$

Table A-12 Stack and Exchange

Operation Name	Mnemonics	Opcode	Addressing							Bytes	States	Operation	Flag					
													7	6	4	2	1	0
			IMMED	EXT	IND	REG	REGI	IMP	REL				S	Z	H	P/V	N	C
PUSH	PUSH zz	11 zz0 101				S		D		1	11	zzLr→$(SP-2)_M$ zzHr→$(SP-1)_M$ $SP_R-2→SP_R$	·	·	·	·	·	·
	PUSH IX	11 011 101 11 100 101						S/D		2	14	IXLr→$(SP-2)_M$ IXHr→$(SP-1)_M$ $SP_R-2→SP_R$	·	·	·	·	·	·
	PUSH IY	11 111 101 11 100 101						S/D		2	14	IYLr→$(SP-2)_M$ IYHr→$(SP-1)_M$ $SP_R-2→SP_R$	·	·	·	·	·	·
POP	POP zz	11 zz0 001				D		S		1	9	$(SP+1)_M$→zzHr 4 $(SP)_M$→zzLr $SP_R+2→SP_R$	·	·	·	·	·	·
	POP IX	11 011 101 11 100 001						S/D		2	12	$(SP+1)_M$→IXHr $(SP)_M$→IXLr $SP_R+2→SP_R$	·	·	·	·	·	·
	POP IY	11 111 101 11 100 001						S/D		2	12	$(SP+1)_M$→IYHr $(SP)_M$→IYLr $SP_R+2→SP_R$	·	·	·	·	·	·
Exchange	EX AF,AF'	00 001 000						S/D		1	4	$AF_R↔AF_R'$	·	·	·	·	·	·
	EX DE,HL	11 101 011						S/D		1	3	$DE_R↔HL_R$	·	·	·	·	·	·
	EXX	11 011 001						S/D		1	3	$BC_R↔BC_R'$ $DE_R↔DE_R'$ $HL_R↔HL_R'$	·	·	·	·	·	·
	EX (SP),HL	11 100 011						S/D		1	16	Hr↔$(SP+1)_M$ Lr↔$(SP)_M$	·	·	·	·	·	·
	EX (SP),IX	11 011 101 11 100 011						S/D		2	19	IXHr↔$(SP+1)_M$ IXLr↔$(SP)_M$	·	·	·	·	·	·
	EX (SP),IY	11 111 101 11 100 011						S/D		2	19	IYHr↔$(SP+1)_M$ IYLr↔$(SP)_M$	·	·	·	·	·	·

Note: 4 In the case of POP AF, Flag is written a current contents of the stack.

A.2.3 Program Control Instructions

Table A-13 Program Control

Operation Name	Mnemonics	Opcode	Addressing							Bytes	States	Operation	Flag					
													7	6	4	2	1	0
			IMMED	EXT	IND	REG	REGI	IMP	REL				S	Z	H	P/V	N	C
Call	CALL mn	11 001 101		D						3	16	$PCHr \rightarrow (SP-1)_M$	·	·	·	·	·	·
		⟨ n ⟩										$PCLr \rightarrow (SP-2)_M$						
		⟨ m ⟩										$mn \rightarrow PC_R$						
												$SP_R - 2 \rightarrow SP_R$						
	CALL f,mn	11 f 100		D						3	6 (f : false)	continue : f is false	·	·	·	·	·	·
		⟨ n ⟩									16 (f : true)	CALL mn : f is true						
		⟨ m ⟩																
Jump	DJNZ j	00 010 000							D	2	9 ($Br \neq 0$)	$Br - 1 \rightarrow Br$	·	·	·	·	·	·
		⟨j-2⟩								2	7 ($Br = 0$)	continue : $Br = 0$						
												$PC_R + j \rightarrow PC_R$: $Br \neq 0$						
	JP f,mn	11 f 010		D						3	6 (f : false)	$mn \rightarrow PC_R$: f is true	·	·	·	·	·	·
		⟨ n ⟩								3	9 (f : true)	continue : f is false						
		⟨ m ⟩																
	JP mn	11 000 011		D						3	9	$mn \rightarrow PC_R$	·	·	·	·	·	·
		⟨ n ⟩																
		⟨ m ⟩																
	JP (HL)	11 101 001					D			1	3	$HL_R \rightarrow PC_R$	·	·	·	·	·	·
	JP (IX)	11 011 101					D			2	6	$IX_R \rightarrow PC_R$	·	·	·	·	·	·
		11 101 001																
	JP (IY)	11 111 101					D			2	6	$IY_R \rightarrow PC_R$	·	·	·	·	·	·
		11 101 001																
	JR j	00 011 000							D	2	8	$PC_R + j \rightarrow PC_R$	·	·	·	·	·	·
		⟨j-2⟩																
	JR C,j	00 111 000							D	2	6	continue : $C = 0$	·	·	·	·	·	·
		⟨j-2⟩								2	8	$PC_R + j \rightarrow PC_R$: $C = 1$						
	JR NC,j	00 110 000							D	2	6	continue : $C = 1$	·	·	·	·	·	·
		⟨j-2⟩								2	8	$PC_R + j \rightarrow PC_R$: $C = 0$						
	JR Z,j	00 101 000							D	2	6	continue : $Z = 0$	·	·	·	·	·	·
		⟨j-2⟩								2	8	$PC_R + j \rightarrow PC_R$: $Z = 1$						
	JR NZ,j	00 100 000							D	2	6	continue : $Z = 1$	·	·	·	·	·	·
		⟨j-2⟩								2	8	$PC_R + j \rightarrow PC_R$: $Z = 0$						
Return	RET	11 001 001						D		1	9	$(SP)_M \rightarrow PCLr$	·	·	·	·	·	·
												$(SP+1)_M \rightarrow PCHr$						
												$SP_R + 2 \rightarrow SP_R$						
	RET f	11 f 000							D	1	5 (f : false)	continue : f is false	·	·	·	·	·	·
										1	10 (f : true)	RET : f is true						
	RETI	11 101 101						D		2	22	$(SP)_M \rightarrow PCLr$	·	·	·	·	·	·
		01 001 101										$(SP+1)_M \rightarrow PCHr$						
												$SP_R + 2 \rightarrow SP_R$						
	RETN	11 101 101						D		2	12	$(SP)_M \rightarrow PCLr$	·	·	·	·	·	·
		01 000 101										$(SP+1)_M \rightarrow PCHr$						
												$SP_R + 2 \rightarrow SP_R$						
												$IEF_2 \rightarrow IEF_1$						
Restart	RST v	11 v 111						D		1	11	$PCHr \rightarrow (SP-1)_M$	·	·	·	·	·	·
												$PCLr \rightarrow (SP-2)_M$						
												$0 \rightarrow PCHr$						
												$v \rightarrow PCLr$						
												$SP_R - 2 \rightarrow SP_R$						

A.2.4 I/O Instructions

Table A-14 I/O

Operation Name	Mnemonics	Opcode	Addressing							Bytes	States	Operation	Flag					
													7	6	4	2	1	0
			IMMED	EXT	IND	REG	REGI	IMP	I/O				S	Z	H	P/V	N	C
Input	IN A,(m)	11 011 011 ⟨ m ⟩						D	S	2	9	$(Am)_I \to Ar$ $m \to A_0 \sim A_7$ $Ar \to A_8 \sim A_{15}$	·	·	·	·	·	·
	IN g,(C)	11 101 101 01 g 000				D			S	2	9	$(BC)_I \to gr$ g=110 : Only the flags will change. $Cr \to A_0 \sim A_7$ $Br \to A_8 \sim A_{15}$	$\updownarrow$	$\updownarrow$	R	P	R	·
	IN0 g,(m) **	11 101 101 00 g 000 ⟨ m ⟩				D			S	3	12	$(00m)_I \to gr$ g=110 : Only the flags will change. $m \to A_0 \sim A_7$ $00 \to A_8 \sim A_{15}$	$\updownarrow$	$\updownarrow$	R	P	R	·
	IND	11 101 101 10 101 010					D		S	2	12	$(BC)_I \to (HL)_M$ $HL_R - 1 \to HL_R$ $Br - 1 \to Br$ $Cr \to A_0 \sim A_7$ $Br \to A_8 \sim A_{15}$	X	$\updownarrow$[5]	X	X	$\updownarrow$[6]	X
	INDR	11 101 101 10 111 010					D		S	2	14(Br≠0) 12(Br=0)	Q { $(BC)_I \to (HL)_M$ $HL_R - 1 \to HL_R$ $Br - 1 \to Br$ } Repeat Q until Br=0 $Cr \to A_0 \sim A_7$ $Br \to A_8 \sim A_{15}$	X	S	X	X	$\updownarrow$[6]	X
	INI	11 101 101 10 100 010					D		S	2	12	$(BC)_I \to (HL)_M$ $HL_R + 1 \to HL_R$ $Br - 1 \to Br$ $Cr \to A_0 \sim A_7$ $Br \to A_8 \sim A_{15}$	X	$\updownarrow$[5]	X	X	$\updownarrow$[6]	X
	INIR	11 101 101 10 110 010					D		S	2	14(Br≠0) 12(Br=0)	Q { $(BC)_I \to (HL)_M$ $HL_R + 1 \to HL_R$ $Br - 1 \to Br$ } Repeat Q until Br=0 $Cr \to A_0 \sim A_7$ $Br \to A_8 \sim A_{15}$	X	S	X	X	$\updownarrow$[6]	X

(continued)

Note: 5 Z = 1: Br − 1 = 0
Z = 0: Br − 1 ≠ 0
6 N = 1: MSB of Data = 1
N = 0: MSB of Data = 0

Table A-14 I/O (cont)

Operation Name	Mnemonics	Opcode	IMMED	EXT	IND	REG	REGI	IMP	I/O	Bytes	States	Operation	7 S	6 Z	4 H	2 P/V	1 N	0 C
Output	OUT (m),A	11 010 011 ⟨ m ⟩						S	D	2	10	$Ar \rightarrow (Am)_1$ $m \rightarrow A_0 \sim A_7$ $Ar \rightarrow A_8 \sim A_{15}$	·	·	·	·	·	·
	OUT (C),g	11 101 101 01 g 001				S			D	2	10	$gr \rightarrow (BC)_1$ $Cr \rightarrow A_0 \sim A_7$ $Br \rightarrow A_8 \sim A_{15}$	·	·	·	·	·	·
	OUT0 (m),g **	11 101 101 00 g 001 ⟨ m ⟩				S			D	3	13	$gr \rightarrow (00m)_1$ $m \rightarrow A_0 \sim A_7$ $00 \rightarrow A_8 \sim A_{15}$	·	·	·	·	·	·
	OTDM **	11 101 101 10 001 011					S		D	2	14	$(HL)_M \rightarrow (00C)_1$ $HL_R - 1 \rightarrow HL_R$ $Cr - 1 \rightarrow Cr$ $Br - 1 \rightarrow Br$ $Cr \rightarrow A_0 \sim A_7$ $00 \rightarrow A_8 \sim A_{15}$	↕	↕ (5)	↕	P	↕ (6)	↕
	OTDMR **	11 101 101 10 011 011					S		D	2	16(Br≠0) 14(Br=0)	Q: $(HL)_M \rightarrow (00C)_1$ $HL_R - 1 \rightarrow HL_R$ $Cr - 1 \rightarrow Cr$ $Br - 1 \rightarrow Br$ Repeat Q until $Br = 0$ $Cr \rightarrow A_0 \sim A_7$ $00 \rightarrow A_8 \sim A_{15}$	R	S	R	S	↕ (6)	R
	OTDR	11 101 101 10 111 011					S		D	2	14(Br≠0) 12(Br=0)	Q: $(HL)_M \rightarrow (BC)_1$ $HL_R - 1 \rightarrow HL_R$ $Br - 1 \rightarrow Br$ Repeat Q until $Br = 0$ $Cr \rightarrow A_0 \sim A_7$ $Br \rightarrow A_8 \sim A_{15}$	X	S	X	X	↕ (6)	X
	OUTI	11 101 101 10 100 011					S		D	2	12	$(HL)_M \rightarrow (BC)_1$ $HL_R + 1 \rightarrow HL_R$ $Br - 1 \rightarrow Br$ $Cr \rightarrow A_0 \sim A_7$ $Br \rightarrow A_8 \sim A_{15}$	X	↕ (5)	X	X	↕ (6)	X
	OTIR	11 101 101 10 110 011					S		D	2	14(Br≠0) 12(Br=0)	Q: $(HL)_M \rightarrow (BC)_1$ $HL_R + 1 \rightarrow HL_R$ $Br - 1 \rightarrow Br$ Repeat Q until $Br = 0$ $Cr \rightarrow A_0 \sim A_7$ $Br \rightarrow A_8 \sim A_{15}$	X	S	X	X	↕ (6)	X
	TSTIO m **	11 101 101 01 110 100 ⟨ m ⟩	S						S	3	12	$(00C)_1 \cdot m$ $Cr \rightarrow A_0 \sim A_7$ $00 \rightarrow A_8 \sim A_{15}$	↕	↕	S	P	R	R
	OTIM **	11 101 101 10 000 011					S		D	2	14	$(HL)_M \rightarrow (00C)_1$ $HL_R + 1 \rightarrow HL_R$ $Cr + 1 \rightarrow Cr$ $Br - 1 \rightarrow Br$ $Cr \rightarrow A_0 \sim A_7$ $00 \rightarrow A_8 \sim A_{15}$	↕	↕ (5)	↕	P	↕ (6)	↕
	OTIMR **	11 101 101 10 010 011					S		D	2	16(Br≠0) 14(Br=0)	Q: $(HL)_M \rightarrow (00C)_1$ $HL_R + 1 \rightarrow HL_R$ $Cr + 1 \rightarrow Cr$ $Br - 1 \rightarrow Br$ Repeat Q until $Br = 0$ $Cr \rightarrow A_0 \sim A_7$ $00 \rightarrow A_8 \sim A_{15}$	R	S	R	S	↕ (6)	R
	OUTD	11 101 101 10 101 011					S		D	2	12	$(HL)_M \rightarrow (BC)_1$ $HL_R - 1 \rightarrow HL_R$ $Br - 1 \rightarrow Br$ $Cr \rightarrow A_0 \sim A_7$ $Br \rightarrow A_8 \sim A_{15}$	X	↕ (5)	X	X	↕ (6)	X

Note: 5 $Z = 1$: $Br - 1 = 0$
$Z = 0$: $Br - 1 \neq 0$
6 $N = 1$: MSB of Data = 1
$N = 0$: MSB of Data = 0

A.2.5 Special Control Instructions

Table A-15 Special Control

Operation Name	Mnemonics	Opcode	Addressing							Bytes	States	Operation	Flag					
			IMMED	EXT	IND	REG	REGI	IMP	REL				7 S	6 Z	4 H	2 P/V	1 N	0 C
Special Function	DAA	00 100 111						S/D		1	4	Decimal Adjust Accumulator	↕	↕	↕	P	·	↕
Carry Control	CCF	00 111 111								1	3	$\overline{C}$→C	·	·	R	·	R	↕
	SCF	00 110 111								1	3	1→C	·	·	R	·	R	S
CPU Control	DI	11 110 011								1	3	0→IEF_1, 0→IEF_2 7	·	·	·	·	·	·
	EI	11 111 011								1	3	1→IEF_1, 1→IEF_2 7	·	·	·	·	·	·
	HALT	01 110 110								1	3	CPU halted	·	·	·	·	·	·
	IM 0	11 101 101 01 000 110								2	6	Interrupt mode 0	·	·	·	·	·	·
	IM 1	11 101 101 01 010 110								2	6	Interrupt mode 1	·	·	·	·	·	·
	IM 2	11 101 101 01 011 110								2	6	Interrupt mode 2	·	·	·	·	·	·
	NOP	00 000 000								1	3	No operation	·	·	·	·	·	·
	SLP **	11 101 101 01 110 110								2	8	Sleep	·	·	·	·	·	·

Note: 7 Interrupts are not sampled at the end of DI or EI.

APPENDIX B. INSTRUCTION SUMMARY IN ALPHABETICAL ORDER

MNEMONICS	Bytes	Machine Cycles	States
ADC A,m	2	2	6
ADC A,g	1	2	4
ADC A, (HL)	1	2	6
ADC A, (IX+d)	3	6	14
ADC A, (IY+d)	3	6	14
ADD A,m	2	2	6
ADD A,g	1	2	4
ADD A, (HL)	1	2	6
ADD A, (IX+d)	3	6	14
ADD A, (IY+d)	3	6	14
ADC HL,ww	2	6	10
ADD HL,ww	1	5	7
ADD IX,xx	2	6	10
ADD IY,yy	2	6	10
AND m	2	2	6
AND g	1	2	4
AND (HL)	1	2	6
AND (IX+d)	3	6	14
AND (IY+d)	3	6	14
BIT b, (HL)	2	3	9
BIT b, (IX+d)	4	5	15
BIT b, (IY+d)	4	5	15
BIT b,g	2	2	6
CALL f,mn	3	2	6 (If condition is false)
	3	6	16 (If condition is true)

Note ** : New instructions added to Z80

(continued)

MNEMONICS	Bytes	Machine Cycles	States
CALL mn	3	6	16
CCF	1	1	3
CPD	2	6	12
CPDR	2	8	14
			(If $BC_R \neq 0$ and $Ar \neq (HL)_M$)
	2	6	12
			(If $BC_R = 0$ or $Ar = (HL)_M$)
CP (HL)	1	2	6
CPI	2	6	12
CPIR	2	8	14
			(If $BC_R \neq 0$ and $Ar \neq (HL)_M$)
	2	6	12
			(If $BC_R = 0$ or $Ar = (HL)_M$)
CP (IX+d)	3	6	14
CP (IY+d)	3	6	14
CPL	1	1	3
CP m	2	2	6
CP g	1	2	4
DAA	1	2	4
DEC (HL)	1	4	10
DEC IX	2	3	7
DEC IY	2	3	7
DEC (IX+d)	3	8	18
DEC (IY+d)	3	8	18
DEC g	1	2	4
DEC ww	1	2	4
DI	1	1	3

(continued)

MNEMONICS	Bytes	Machine Cycles	States
DJNZ j	2	5	9 (If Br≠0)
	2	3	7 (If Br=0)
EI	1	1	3
EX AF,AF'	1	2	4
EX DE,HL	1	1	3
EX (SP),HL	1	6	16
EX (SP),IX	2	7	19
EX (SP),IY	2	7	19
EXX	1	1	3
HALT	1	1	3
IM 0	2	2	6
IM 1	2	2	6
IM 2	2	2	6
INC g	1	2	4
INC (HL)	1	4	10
INC (IX+d)	3	8	18
INC (IY+d)	3	8	18
INC ww	1	2	4
INC IX	2	3	7
INC IY	2	3	7
IN A,(m)	2	3	9
IN g,(C)	2	3	9
INI	2	4	12
INIR	2	6	14 (If Br≠0)
	2	4	12 (If Br=0)
IND	2	4	12
INDR	2	6	14 (If Br≠0)

(continued)

MNEMONICS	Bytes	Machine Cycles	States
INDR	2	4	12 (If Br=0)
INO g,(m)**	3	4	12
JP f,mn	3	2	6 (If f is false)
	3	3	9 (If f is true)
JP (HL)	1	1	3
JP (IX)	2	2	6
JP (IY)	2	2	6
JP mn	3	3	9
JR j	2	4	8
JR C,j	2	2	6 (If condition is false)
	2	4	8 (If condition is true)
JR NC,j	2	2	6 (If condition is false)
	2	4	8 (If condition is true)
JR Z,j	2	2	6 (If condition is false)
	2	4	8 (If condition is true)
JR NZ,j	2	2	6 (If condition is false)
	2	4	8 (If condition is true)

(continued)

MNEMONICS	Bytes	Machine Cycles	States
LD A, (BC)	1	2	6
LD A, (DE)	1	2	6
LD A,I	2	2	6
LD A, (mn)	3	4	12
LD A,R	2	2	6
LD (BC),A	1	3	7
LDD	2	4	12
LD (DE),A	1	3	7
LD ww,mn	3	3	9
LD ww,(mn)	4	6	18
LDDR	2	6	14 (If $BC_R \neq 0$)
	2	4	12 (If $BC_R = 0$)
LD (HL),m	2	3	9
LD HL,(mn)	3	5	15
LD (HL),g	1	3	7
LDI	2	4	12
LD I,A	2	2	6
LDIR	2	6	14 (If $BC_R \neq 0$)
	2	4	12 (If $BC_R = 0$)
LD IX,mn	4	4	12
LD IX,(mn)	4	6	18
LD (IX+d),m	4	5	15
LD (IX+d),g	3	7	15
LD IY,mn	4	4	12
LD IY,(mn)	4	6	18
LD (IY+d),m	4	5	15
LD (IY+d),g	3	7	15

(continued)

MNEMONICS	Bytes	Machine Cycles	States
LD (mn),A	3	5	13
LD (mn),ww	4	7	19
LD (mn),HL	3	6	16
LD (mn),IX	4	7	19
LD (mn),IY	4	7	19
LD R,A	2	2	6
LD g,(HL)	1	2	6
LD g,(IX+d)	3	6	14
LD g,(IY+d)	3	6	14
LD g,m	2	2	6
LD g,g'	1	2	4
LD SP,HL	1	2	4
LD SP,IX	2	3	7
LD SP,IY	2	3	7
MLT ww**	2	13	17
NEG	2	2	6
NOP	1	1	3
OR (HL)	1	2	6
OR (IX+d)	3	6	14
OR (IY+d)	3	6	14
OR m	2	2	6
OR g	1	2	4
OTDM**	2	6	14
OTDMR**	2	8	16 (If Br≠0)
	2	6	14 (If Br=0)
OTDR	2	6	14 (If Br≠0)
	2	4	12 (If Br=0)

(continued)

MNEMONICS	Bytes	Machine Cycles	States
OTIM**	2	6	14
OTIMR**	2	8	16 (If Br≠0)
	2	6	14 (If Br=0)
OTIR	2	6	14 (If Br≠0)
	2	4	12 (If Br=0)
OUTD	2	4	12
OUTI	2	4	12
OUT (m),A	2	4	10
OUT (C),g	2	4	10
OUT0 (m),g **	3	5	13
POP IX	2	4	12
POP IY	2	4	12
POP zz	1	3	9
PUSH IX	2	6	14
PUSH IY	2	6	14
PUSH zz	1	5	11
RES b,(HL)	2	5	13
RES b,(IX+d)	4	7	19
RES b,(IY+d)	4	7	19
RES b,g	2	3	7
RET	1	3	9
RET f	1	3	5
			(If condition is false)
	1	4	10
			(If condition is true)
RETI	2	10	22
RETN	2	4	12

(continued)

MNEMONICS	Bytes	Machine Cycles	States
RLA	1	1	3
RLCA	1	1	3
RLC (HL)	2	5	13
RLC (IX+d)	4	7	19
RLC (IY+d)	4	7	19
RLC g	2	3	7
RLD	2	8	16
RL (HL)	2	5	13
RL (IX+d)	4	7	19
RL (IY+d)	4	7	19
RL g	2	3	7
RRA	1	1	3
RRCA	1	1	3
RRC (HL)	2	5	13
RRC (IX+d)	4	7	19
RRC (IY+d)	4	7	19
RRC g	2	3	7
RRD	2	8	16
RR (HL)	2	5	13
RR (IX+d)	4	7	19
RR (IY+d)	4	7	19
RR g	2	3	7
RST v	1	5	11
SBC A,(HL)	1	2	6
SBC A,(IX+d)	3	6	14
SBC A,(IY+d)	3	6	14
SBC A,m	2	2	6

(continued)

MNEMONICS	Bytes	Machine Cycles	States
SBC A,g	1	2	4
SBC HL,ww	2	6	10
SCF	1	1	3
SET b,(HL)	2	5	13
SET b,(IX+d)	4	7	19
SET b,(IY+d)	4	7	19
SET b,g	2	3	7
SLA (HL)	2	5	13
SLA (IX+d)	4	7	19
SLA (IY+d)	4	7	19
SLA g	2	3	7
SLP**	2	2	8
SRA (HL)	2	5	13
SRA (IX+d)	4	7	19
SRA (IY+d)	4	7	19
SRA g	2	3	7
SRL (HL)	2	5	13
SRL (IX+d)	4	7	19
SRL (IY+d)	4	7	19
SRL g	2	3	7
SUB (HL)	1	2	6
SUB (IX+d)	3	6	14
SUB (IY+d)	3	6	14
SUB m	2	2	6
SUB g	1	2	4
**TSTIO m	3	4	12
**TST g	2	3	7

(continued)

MNEMONICS	Bytes	Machine Cycles	States
TST m**	3	3	9
TST (HL)**	2	4	10
XOR (HL)	1	2	6
XOR (IX+d)	3	6	14
XOR (IY+d)	3	6	14
XOR m	2	2	6
XOR g	1	2	4

APPENDIX C. OPCODE MAP

Table C-1 First Opcode Map

Instruction format: XX

				0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
			ww(Lo = All)	BC	DE	HL	SP									Lo = 0 – 7				
			zz													BC	DE	HL	AF	zz
			g (Lo = 0 – 7)	B	D	H	(HL)	B	D	H	(HL)					NZ	NC	PO	P	f
			v													00H	10H	20H	30H	v
			HI	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111	
		LO		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
S (Hi = All)	B	0000	0	NOP	DJNZ j	JR NZ, j	JR NC, j									RET f				0
	C	0001	1	LD ww, mn							(Note 1)					POP zz				1
	D	0010	2	LD (ww), A		LD (mn), HL	LD (mn), A									JP f, mn				2
																JP mn	OUT (m), A	EX (SP), HL	DI	3
	E	0011	3	INC ww				LD g, s				ADD A, s	SUB s	AND s	OR s					
	H	0100	4	INC g			(Note 1)									CALL f, mn				4
	L	0101	5	DEC g			(Note 1)									PUSH zz				5
	(HL)	0110	6	LD g, m			(Note 1)	(Note 2)			HALT	(Note 2)	(Note 2)	(Note 2)	(Note 2)	ADD A, m	SUB m	AND m	OR m	6
	A	0111	7	RLCA	RLA	DAA	SCF									RST v				7
	B	1000	8	EXAF, AF'	JR j	JR Z, j	JR C, j									RET f				8
	C	1001	9	ADD HL, ww												RET	EXX	JP (HL)	LD SP, HL	9
	D	1010	A	LD A, (ww)		LD HL, (mn)	LD A, (mn)									JP f, mn				A
	E	1011	B	DEC ww				LD g, s				ADC A, s	SBC A, s	XOR s	CP s	Table 2	IN A, (m)	EX DE, HL	EI	B
	H	1100	C	INC g												CALL f, mn				C
	L	1101	D	DEC g												CALL mn	(Note 3)	Table 3	(Note 3)	D
	(HL)	1110	E	LD g, m				(Note 2)				(Note 2)	(Note 2)	(Note 2)	(Note 2)	ADC A, m	SBC A, m	XOR m	CP m	E
	A	1111	F	RRCA	RRA	CPL	CCF									RST v				F
				0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
			g (Lo = 8 – F)	C	E	L	A	C	E	L	A					Z	C	PE	M	f
																08H	18H	28H	38H	v
																Lo = 8 – F				

Notes:
1. (HL) replaces g.
2. (HL) replaces s.
3. If DDH is added as first opcode for the instructions which have HL or (HL) as an operand in table 1, the instructions are executed replacing HL with IX and (HL) with (IX + d).

   ex: 22H: LD (mn), HL
   DDH 22H: LD (mn), IX

   If FDH is added as first opcode for the instructions which have HL or (HL) as an operand in table 1, the instructions are executed replacing HL with IY and (HL) with (IY + d).

   ex: 34H: INC (HL)
   FDH 34H: INC (IY + d)

   However, JP (HL) and EX DE, HL are exceptions. Note the followings:
   If DDH is added as first opcode for JP (HL), (IX) replaces (HL) as operand and JP (IX) is executed.
   If FDH is added as first opcode for JP (HL), (IY) replaces (HL) as operand and JP (IY) is executed.
   Even if DDH or FDH is added as first opcode for EX DE, HL, HL is not replaced and the instruction is regarded as illegal.

Table C-2 Second Opcode Map
Instruction format: CB XX

g (Hi = All)	LO		0000 (0)	0001 (1)	0010 (2)	0011 (3)	0100 (4)	0101 (5)	0110 (6)	0111 (7)	1000 (8)	1001 (9)	1010 (A)	1011 (B)	1100 (C)	1101 (D)	1110 (E)	1111 (F)	
		b (Lo = 0 – 7)					0	2	4	6	0	2	4	6	0	2	4	6	
B	0000	0																	0
C	0001	1																	1
D	0010	2																	2
E	0011	3																	3
H	0100	4	RLC g	RL g	SLA g		BIT b,g				RES b,g				SET b,g				4
L	0101	5																	5
(HL)	0110	6	(Note 1)	(Note 1)	(Note 1)		(Note 1)				(Note 1)				(Note 1)				6
A	0111	7																	7
B	1000	8																	8
C	1001	9																	9
D	1010	A																	A
E	1011	B																	B
H	1100	C	RRC g	RR g	SRA g	SRL g	BIT b,g				RES b,g				SET b,g				C
L	1101	D																	D
(HL)	1110	E	(Note 1)	(Note 1)	(Note 1)	(Note 1)	(Note 1)				(Note 1)				(Note 1)				E
A	1111	F																	F
			0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
		b (Lo = 8 – nF)					1	3	5	7	1	3	5	7	1	3	5	7	

Note: 1. If DDH is added as first opcode for the instructions which have (HL) as operand in table 2, the instructions are executed replacing (HL) with (IX + d).
If FDH is added as first opcode for the instructions which have (HL) as operand in table 2, the instructions are executed replacing (HL) with (IY + d).

Table C-3 Second Opcode Map
Instruction format: ED XX

Lo		0000 (0)	0001 (1)	0010 (2)	0011 (3)	0100 (4)	0101 (5)	0110 (6)	0111 (7)	1000 (8)	1001 (9)	1010 (A)	1011 (B)	1100 (C)	1101 (D)	1110 (E)	1111 (F)	
ww (Lo = All)						BC	DE	HL	SP									
g (Lo = 0 – 7)		B	D	H		B	D	H										
0000	0	IN0 g, (m)				IN g, (C)						LDI	LDIR					0
0001	1	OUT0 (m),g				OUT (C),g						CPI	CPIR					1
0010	2					SBC HL, ww						INI	INIR					2
0011	3					LD (mn), ww				OTIM	OTIMR	OUTI	OTIR					3
0100	4	TST g			TST (HL)	NEG		TST m	TSTIO m									4
0101	5					RETN												5
0110	6					IM 0	IM 1		SLP									6
0111	7					LD I,A	LD A,I	RRD										7
1000	8	IN0 g, (m)				IN g, (C)						LDD	LDDR					8
1001	9	OUT0 (m),g				OUT (C),g						CPD	CPDR					9
1010	A					ADC HL, ww						IND	INDR					A
1011	B					LD ww, (mn)				OTDM	OTDMR	OUTD	OTDR					B
1100	C	TST g				MLT ww												C
1101	D					RETI												D
1110	E						IM 2											E
1111	F					LD R,A	LD A,R	RLD										F
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
g (Lo = 8 – F)		C	E	L	A	C	E	L	A									

APPENDIX D. BUS AND CONTROL SIGNAL CONDITION IN EACH MACHINE CYCLE

Instruction	Machine Cycle	States	Address	Data	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
ADD HL,ww	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2– MC_5	TiTiTiTi	•	Z	1	1	1	1	1	1	1
ADD IX,xx ADD IY,yy	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3– MC_6	TiTiTiTi	•	Z	1	1	1	1	1	1	1
ADC HL,ww SBC HL,ww	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3– MC_6	TiTiTiTi	•	Z	1	1	1	1	1	1	1
ADD A,g ADC A,g SUB g	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
SBC A,g AND g OR g XOR g CP g	MC_2	Ti	•	Z	1	1	1	1	1	1	1
ADD A,m ADC A,m SUB m	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
SBC A,m AND m OR m XOR m CP m	MC_2	$T_1T_2T_3$	1st operand Address	m	0	1	0	1	1	1	1
ADD A, (HL) ADC A, (HL) SUB (HL)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
SBC A, (HL) AND (HL) OR (HL) XOR (HL) CP (HL)	MC_2	$T_1T_2T_3$	HL	Data	0	1	0	1	1	1	1

ADD A, (IX+d)
ADD A, (IY+d)
ADC A, (IX+d)
	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0

ADC A, (IY+d) SUB (IX+d) SUB (IY+d) SBC A, (IX+d)	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1

(continued)

Note: • (Address): Invalid
Z (Data): High impedance.
**: New instructions added to Z80

Instruction	Machine Cycle	States	Address	Data	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
SBC A, (IY+d) AND (IX+d)	MC_3	$T_1T_2T_3$	1st operand Address	d	0	1	0	1	1	1	1
AND (IY+d) OR (IX+d) OR (IY+d) XOR (IX+d)	MC_4–MC_5	TiTi	*	Z	1	1	1	1	1	1	1
XOR (IY+d) CP (IX+d) CP (IY+d)	MC_6	$T_1T_2T_3$	IX+d IY+d	Data	0	1	0	1	1	1	1
BIT b,g	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
BIT b, (HL)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	Data	0	1	0	1	1	1	1
BIT b, (IX+d) BIT b, (IY+d)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	d	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	3rd opcode Address	3rd opcode	0	1	0	1	0	1	1
	MC_5	$T_1T_2T_3$	IX+d IY+d	Data	0	1	0	1	1	1	1
CALL mn	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
	MC_4	Ti	*	Z	1	1	1	1	1	1	1
	MC_5	$T_1T_2T_3$	SP−1	PCH	1	0	0	1	1	1	1
	MC_6	$T_1T_2T_3$	SP−2	PCL	1	0	0	1	1	1	1
CALL f,mn (If condition	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
is false)	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1

(continued)

Instruction	Machines Cycle	States	Address	Data	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
CALL f,mn (If condition is true)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
	MC_4	Ti	*	Z	1	1	1	1	1	1	1
	MC_5	$T_1T_2T_3$	SP−1	PCH	1	0	0	1	1	1	1
	MC_6	$T_1T_2T_3$	SP−2	PCL	1	0	0	1	1	1	1
CCF	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
CPI CPD	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	Data	0	1	0	1	1	1	1
	MC_4− MC_6	TiTiTi	*	Z	1	1	1	1	1	1	1
CPIR CPDR (If $BC_R \neq 0$ and $Ar \neq (HL)_M$)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	Data	0	1	0	1	1	1	1
	MC_4− MC_8	TiTiTiTiTi	*	Z	1	1	1	1	1	1	1
CPIR CPDR (If $BC_R = 0$ or $Ar = (HL)_M$)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	Data	0	1	0	1	1	1	1
	MC_4− MC_6	TiTiTi	*	Z	1	1	1	1	1	1	1
CPL	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
DAA	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	Ti	*	Z	1	1	1	1	1	1	1
DI (Note 1)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0

(continued)

Note: 1. Interrupt request is not sampled.

Instruction	Machine Cycle	States	Address	Data	RD	WR	ME	IOE	LIR	HALT	ST
DJNZ j (If Br≠0)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	Ti (Note 2)	*	Z	1	1	1	1	1	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	j-2	0	1	0	1	1	1	1
	MC_4– MC_5	TiTi	*	Z	1	1	1	1	1	1	1
DJNZ j (If Br=0)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	Ti (Note 1)	*	Z	1	1	1	1	1	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	j-2	0	1	0	1	1	1	1
EI (Note 3)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
EX DE, HL EXX	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
EX AF, AF'	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	Ti	*	Z	1	1	1	1	1	1	1
EX (SP), HL	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	SP	Data	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	SP+1	Data	0	1	0	1	1	1	1
	MC_4	Ti	*	Z	1	1	1	1	1	1	1
	MC_5	$T_1T_2T_3$	SP+1	H	1	0	0	1	1	1	1
	MC_6	$T_1T_2T_3$	SP	L	1	0	0	1	1	1	1
EX (SP),IX EX (SP),IY	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	SP	Data	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	SP+1	Data	0	1	0	1	1	1	1
	MC_5	Ti	*	Z	1	1	1	1	1	1	1

(continued)

Note: 2 DMA, refresh, or bus release cannot be executed after this state. (Request is ignored.)
3 Interrupt request is not sampled.

Instruction	Machine Cycle	States	Address	Data	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
EX (SP), IX EX (SP), IY	MC_6	$T_1T_2T_3$	SP+1	IXH IYH	1	0	0	1	1	1	1
	MC_7	$T_1T_2T_3$	SP	IXL IYL	1	0	0	1	1	1	1
HALT	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	----	----	Next opcode Address	Next opcode	0	1	0	1	0	0	0
IM 0 IM 1 IM 2	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
INC g DEC g	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	Ti	*	Z	1	1	1	1	1	1	1
INC (HL) DEC (HL)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	HL	Data	0	1	0	1	1	1	1
	MC_3	Ti	*	Z	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	HL	Data	1	0	0	1	1	1	1
INC (IX+d) INC (IY+d)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
DEC (IX+d) DEC (IY+d)	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	d	0	1	0	1	1	1	1
	MC_4– MC_5	TiTi	*	Z	1	1	1	1	1	1	1
	MC_6	$T_1T_2T_3$	IX+d IY+d	Data	0	1	0	1	1	1	1
	MC_7	Ti	*	Z	1	1	1	1	1	1	1
	MC_8	$T_1T_2T_3$	IX+d IY+d	Data	1	0	0	1	1	1	1
INC ww DEC ww	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	Ti	*	Z	1	1	1	1	1	1	1
INC IX INC IY	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
DEC IX DEC IY	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	Ti	*	Z	1	1	1	1	1	1	1

(continued)

Instruction	Machine Cycle	States	Address	Data	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
IN A,(m)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	m	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	m to A_0-A_7 A to A_8-A_{15}	Data	0	1	1	0	1	1	1
IN g,(C)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	BC	Data	0	1	1	0	1	1	1
IN0 g,(m)**	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	m	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	m to A_0-A_7 00H to A_8-A_{15}	Data	0	1	1	0	1	1	1
INI IND	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	BC	Data	0	1	1	0	1	1	1
	MC_4	$T_1T_2T_3$	HL	Data	1	0	0	1	1	1	1
INIR INDR (If $Br \neq 0$)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	BC	Data	0	1	1	0	1	1	1
	MC_4	$T_1T_2T_3$	HL	Data	1	0	0	1	1	1	1
	MC_5-MC_6	TiTi	*	Z	1	1	1	1	1	1	1
INIR INDR (If $Br = 0$)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	BC	Data	0	1	1	0	1	1	1
	MC_4	$T_1T_2T_3$	HL	Data	1	0	0	1	1	1	1

(continued)

Instruction	Machine Cycle	States	Address	Data	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
JP mn	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
JP f,mn (If f is false)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
JP f,mn (If f is true)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
JP (HL)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
JP (IX) JP (IY)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
JR j	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	j-2	0	1	0	1	1	1	1
	MC_3– MC_4	TiTi	*	Z	1	1	1	1	1	1	1
JR C,j JR NC,j JR Z,j JR NZ,j (If condition is false)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	j-2	0	1	0	1	1	1	1
JR C,j JR NC,j JR Z,j JR NZ,j (If condition is true)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	j-2	0	1	0	1	1	1	1
	MC_3– MC_4	TiTi	*	Z	1	1	1	1	1	1	1
LD g,g'	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	Ti	*	Z	1	1	1	1	1	1	1
LD g,m	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	m	0	1	0	1	1	1	1

(continued)

Instruction	Machine Cycle	States	Address	Data	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
LD g, (HL)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	HL	Data	0	1	0	1	1	1	1
LD g, (IX+d) LD g, (IY+d)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	d	0	1	0	1	1	1	1
	MC_4– MC_5	TiTi	*	Z	1	1	1	1	1	1	1
	MC_6	$T_1T_2T_3$	IX+d IY+d	Data	0	1	0	1	1	1	1
LD (HL),g	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	Ti	*	Z	1	1	1	1	1	1	1
	MC_3	$T_1T_2T_3$	HL	g	1	0	0	1	1	1	1
LD (IX+d),g LD (IY+d),g	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	d	0	1	0	1	1	1	1
	MC_4– MC_6	TiTiTi	*	Z	1	1	1	1	1	1	1
	MC_7	$T_1T_2T_3$	IX+d IY+d	g	1	0	0	1	1	1	1
LD (HL),m	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	m	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	HL	Data	1	0	0	1	1	1	1
LD (IX+d),m LD (IY+d),m	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	d	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
	MC_5	$T_1T_2T_3$	IX+d IY+d	Data	1	0	0	1	1	1	1
LD A, (BC) LD A, (DE)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0

(continued)

Instruction	Machine Cycle	States	Address	Data	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
LD A, (BC) LD A, (DE)	MC_2	$T_1T_2T_3$	BC DE	Data	0	1	0	1	1	1	1
LD A,(mn)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	mn	Data	0	1	0	1	1	1	1
LD (BC),A LD (DE),A	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	Ti	*	Z	1	1	1	1	1	1	1
	MC_3	$T_1T_2T_3$	BC DE	A	1	0	0	1	1	1	1
LD (mn),A	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
	MC_4	Ti	*	Z	1	1	1	1	1	1	1
	MC_5	$T_1T_2T_3$	mn	A	1	0	0	1	1	1	1
LD A,I (Note 4) LD A,R	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
LD I,A LD R,A	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
LD ww, mn	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
LD IX,mn LD IY,mn	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
LD HL, (mn)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1

(continued)

Note: 4 Interrupt request is not sampled.

Instruction	Machine Cycle	States	Address	Data	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
LD HL, (mn)	MC_3	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	mn	Data	0	1	0	1	1	1	1
	MC_5	$T_1T_2T_3$	mn+ 1	Data	0	1	0	1	1	1	1
LD ww,(mn)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
	MC_5	$T_1T_2T_3$	mn	Data	0	1	0	1	1	1	1
	MC_6	$T_1T_2T_3$	mn+ 1	Data	0	1	0	1	1	1	1
LD IX,(mn) LD IY,(mn)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
	MC_5	$T_1T_2T_3$	mn	Data	0	1	0	1	1	1	1
	MC_6	$T_1T_2T_3$	mn+ 1	Data	0	1	0	1	1	1	1
LD (mn),HL	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
	MC_4	Ti	*	Z	1	1	1	1	1	1	1
	MC_5	$T_1T_2T_3$	mn	L	1	0	0	1	1	1	1
	MC_5	$T_1T_2T_3$	mn+ 1	H	1	0	0	1	1	1	1

(continued)

Instruction	Machine Cycle	States	Address	Data	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
LD (mn),ww	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
	MC_5	Ti	*	Z	1	1	1	1	1	1	1
	MC_6	$T_1T_2T_3$	mn	wwL	1	0	0	1	1	1	1
	MC_7	$T_1T_2T_3$	mn+1	wwH	1	0	0	1	1	1	1
LD (mn),IX LD (mn),IY	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	n	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	2nd operand Address	m	0	1	0	1	1	1	1
	MC_5	Ti	*	Z	1	1	1	1	1	1	1
	MC_6	$T_1T_2T_3$	mn	IXL IYL	1	0	0	1	1	1	1
	MC_7	$T_1T_2T_3$	mn+1	IXH IYH	1	0	0	1	1	1	1
LD SP, HL	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	Ti	*	Z	1	1	1	1	1	1	1
LD SP,IX LD SP,IY	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	Ti	*	Z	1	1	1	1	1	1	1
LDI LDD	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	Data	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	DE	Data	1	0	0	1	1	1	1

(continued)

Instruction	Machine Cycle	States	Address	Data	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
LDIR LDDR (If $BC_R \neq 0$)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	Data	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	DE	Data	1	0	0	1	1	1	1
	MC_5– MC_6	TiTi	*	Z	1	1	1	1	1	1	1
LDIR LDDR (If $BC_R = 0$)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	Data	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	DE	Data	1	0	0	1	1	1	1
MLT ww**	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3– MC_{13}	TiTiTiTi TiTiTiTi TiTiTi	*	Z	1	1	1	1	1	1	1
NEG	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
NOP	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
OUT (m),A	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	1st operand Address	m	0	1	0	1	1	1	1
	MC_3	Ti	*	Z	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	m to A_0–A_7 A to A_8–A_{15}	A	1	0	1	0	1	1	1

(continued)

Instruction	Machine Cycle	States	Address	Data	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
OUT (C),g	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	Ti	*	Z	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	BC	g	1	0	1	0	1	1	1
OUT0 (m),g**	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	m	0	1	0	1	1	1	1
	MC_4	Ti	*	Z	1	1	1	1	1	1	1
	MC_5	$T_1T_2T_3$	m to A_0-A_7 00H to A_8-A_{15}	g	1	0	1	0	1	1	1
OTIM** OTDM**	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	Ti	*	Z	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	HL	Data	0	1	0	1	1	1	1
	MC_5	$T_1T_2T_3$	C to A_0-A_7 00H to A_8-A_{15}	Data	1	0	1	0	1	1	1
	MC_6	Ti	*	Z	1	1	1	1	1	1	1
OTIMR** OTDMR** (If Br≠0)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	Ti	*	Z	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	HL	Data	0	1	0	1	1	1	1
	MC_5	$T_1T_2T_3$	C to A_0-A_7 00H to A_8-A_{15}	Data	1	0	1	0	1	1	1
	MC_6-MC_8	TiTiTi	*	Z	1	1	1	1	1	1	1
OTIMR** OTDMR** (If Br=0)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	Ti	*	Z	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	HL	Data	0	1	0	1	1	1	1
	MC_5	$T_1T_2T_3$	C to A_0-A_7 00H to A_8-A_{15}	Data	1	0	1	0	1	1	1
	MC_6	Ti	*	Z	1	1	1	1	1	1	1

(continued)

Instruction	Machine Cycle	States	Address	Data	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
OUTI OUTD	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	Data	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	BC	Data	1	0	1	0	1	1	1
OTIR OTDR (If $Br \neq 0$)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	Data	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	BC	Data	1	0	1	0	1	1	1
	MC_5– MC_6	TiTi	*	Z	1	1	1	1	1	1	1
OTIR OTDR (If $Br = 0$)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	Data	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	BC	Data	1	0	1	0	1	1	1
POP zz	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	SP	Data	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	SP+1	Data	0	1	0	1	1	1	1
POP IX POP IY	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0

(continued)

Instruction	Machine Cycle	States	Address	Data	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
POP IX POP IY	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	SP	Data	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	SP+1	Data	0	1	0	1	1	1	1
PUSH zz	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2– MC_3	TiTi	*	Z	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	SP−1	zzH	1	0	0	1	1	1	1
	MC_5	$T_1T_2T_3$	SP−2	zzL	1	0	0	1	1	1	1
PUSH IX PUSH IY	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3– MC_4	TiTi	*	Z	1	1	1	1	1	1	1
	MC_5	$T_1T_2T_3$	SP−1	IXH IYH	1	0	0	1	1	1	1
	MC_6	$T_1T_2T_3$	SP−2	IXL IYL	1	0	0	1	1	1	1
RET	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	SP	Data	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	SP+1	Data	0	1	0	1	1	1	1
RET f (If condition is false)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2– MC_3	TiTi	*	Z	1	1	1	1	1	1	1
RET f (If condition is true)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	Ti	*	Z	1	1	1	1	1	1	1
	MC_3	$T_1T_2T_3$	SP	Data	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	SP+1	Data	0	1	0	1	1	1	1

(continued)

Instruction	Machine Cycle	States	Address	Data	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
RETI	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0 [5] 1	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0 [5] 1	1	1
	MC_3 – MC_5	TiTiTi	•	Z	1	1	1	1	1 [5] 1	1	1
	MC_6	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0 [5] 0	1	1
	MC_7	Ti	•	Z	1	1	1	1	1 [5] 1	1	1
	MC_8	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0 [5] 1	1	1
	MC_9	$T_1T_2T_3$	SP	Data	0	1	0	1	1 [5] 1	1	1
	MC_{10}	$T_1T_2T_3$	SP+1	Data	0	1	0	1	1 [5] 1	1	1
RLCA RLA RRCA RRA	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
RLC g RL g RRC g RR g SLA g SRA g SRL g	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	Ti	•	Z	1	1	1	1	1	1	1
RLC (HL) RL (HL) RRC (HL) RR (HL) SLA (HL) SRA (HL) SRL (HL)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	Data	0	1	0	1	1	1	1
	MC_4	Ti	•	Z	1	1	1	1	1	1	1
	MC_5	$T_1T_2T_3$	HL	Data	1	0	0	1	1	1	1

(continued)

Note: 5 The upper and lower data show the state of LIR when IOC = 1 and IOC = 0 respectively.

Instruction	Machine Cycle	States	Address	Data	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
RLC (IX + d) RLC (IY + d) RL (IX + d) RL (IY + d) RRC (IX + d) RRC (IY + d) RR (IX + d) RR (IY + d) SLA (IX + d) SLA (IY + d) SRA (IX + d) SRA (IY + d) SRL (IX + d) SRL (IY + d)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	d	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	3rd opcode Address	3rd opcode	0	1	0	1	0	1	1
	MC_5	$T_1T_2T_3$	IX+d IY+d	Data	0	1	0	1	1	1	1
	MC_6	Ti	*	Z	1	1	1	1	1	1	1
	MC_7	$T_1T_2T_3$	IX+d IY+d	Data	1	0	0	1	1	1	1
RLD RRD	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	Data	0	1	0	1	1	1	1
	MC_4–MC_7	TiTiTiTi	*	Z	1	1	1	1	1	1	1
	MC_8	$T_1T_2T_3$	HL	Data	1	0	0	1	1	1	1
RST v	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2–MC_3	TiTi	*	Z	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	SP−1	PCH	1	0	0	1	1	1	1
	MC_5	$T_1T_2T_3$	SP−2	PCL	1	0	0	1	1	1	1
SCF	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
SET b,g RES b,g	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	Ti	*	Z	1	1	1	1	1	1	1
SET b, (HL) RES b, (HL)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	HL	Data	0	1	0	1	1	1	1
	MC_4	Ti	*	Z	1	1	1	1	1	1	1
	MC_5	$T_1T_2T_3$	HL	Data	1	0	0	1	1	1	1

(continued)

Instruction	Machine Cycle	States	Address	Data	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
SET b, (IX+d) SET b, (IY+d) RES b, (IX+d) RES b, (IY+d)	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	d	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	3rd opcode Address	3rd opcode	0	1	0	1	0	1	1
	MC_5	$T_1T_2T_3$	IX+d IY+d	Data	0	1	0	1	1	1	1
	MC_6	Ti	*	Z	1	1	1	1	1	1	1
	MC_7	$T_1T_2T_3$	IX+d IY+d	Data	1	0	0	1	1	1	1
SLP**	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	—	—	7FFFFH	Z	1	1	1	1	1	0	1
TSTIO m**	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$ Address	2nd opcode opcode	2nd	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	m	0	1	0	1	1	1	1
	MC_4	$T_1T_2T_3$	C to A_0-A_7 00H to A_8-A_{15}	Data	0	1	1	0	1	1	1
TST g**	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	Ti	*	Z	1	1	1	1	1	1	1
TST m**	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	$T_1T_2T_3$	1st operand Address	m	0	1	0	1	1	1	1
TST (HL)**	MC_1	$T_1T_2T_3$	1st opcode Address	1st opcode	0	1	0	1	0	1	0
	MC_2	$T_1T_2T_3$	2nd opcode Address	2nd opcode	0	1	0	1	0	1	1
	MC_3	Ti	*	Z	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	HL	Data	0	1	0	1	1	1	1

(continued)

Instruction	Machine Cycle	States	Address	Data	$\overline{RD}$	$\overline{WR}$	$\overline{ME}$	$\overline{IOE}$	$\overline{LIR}$	$\overline{HALT}$	ST
$\overline{NMI}$	MC_1	$T_1T_2T_3$	Next opcode Address (PC)		0	1	0	1	0	1	0
	MC_2– MC_3	TiTi	*	Z	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	SP−1	PCH	1	0	0	1	1	1	1
	MC_5	$T_1T_2T_3$	SP−2	PCL	1	0	0	1	1	1	1
$\overline{INT_0}$ Mode 0 (RST Inserted)	MC_1	$T_1T_2T_W$ T_WT_3	Next opcode Address (PC)	1st opcode	1	1	1	0	0	1	0
	MC_2– MC_3	TiTi	*	Z	1	1	1	1	1	1	1
	MC_4	$T_1T_2T_3$	SP−1	PCH	1	0	0	1	1	1	1
	MC_5	$T_1T_2T_3$	SP−2	PCL	1	0	0	1	1	1	1
$\overline{INT_0}$ Mode 0 (CALL Inserted)	MC_1	$T_1T_2T_W$ T_WT_3	Next opcode Address (PC)	1st opcode	1	1	1	0	0	1	0
	MC_2	$T_1T_2T_3$	PC	n	0	1	0	1	1	1	1
	MC_3	$T_1T_2T_3$	PC+1	m	0	1	0	1	1	1	1
	MC_4	Ti	*	Z	1	1	1	1	1	1	1
	MC_5	$T_1T_2T_3$	SP−1	PC+2(H)	1	0	0	1	1	1	1
	MC_6	$T_1T_2T_3$	SP−2	PC+2(L)	1	0	0	1	1	1	1
$\overline{INT_0}$ Mode 1	MC_1	$T_1T_2T_W$ T_WT_3	Next opcode Address (PC)		1	1	1	0	0	1	0
	MC_2	$T_1T_2T_3$	SP−1	PCH	1	0	0	1	1	1	1
	MC_3	$T_1T_2T_3$	SP−2	PCL	1	0	0	1	1	1	1
$\overline{INT_0}$ Mode 2	MC_1	$T_1T_2T_W$ T_WT_3	Next opcode Address (PC)	Vector	1	1	1	0	0	1	0
	MC_2	Ti	*	Z	1	1	1	1	1	1	1
	MC_3	$T_1T_2T_3$	SP−1	PCH	1	0	0	1	1	1	1
	MC_4	$T_1T_2T_3$	SP−2	PCL	1	0	0	1	1	1	1
	MC_5	$T_1T_2T_3$	I, Vector	Data	0	1	0	1	1	1	1
	MC_6	$T_1T_2T_3$	I, Vector+1	Data	0	1	0	1	1	1	1
$\overline{INT_1}$ $\overline{INT_2}$ Internal Interrupts	MC_1	$T_1T_2T_W$ T_WT_3	Next opcode Address (PC)		1	1	1	1	1	1	0
	MC_2	Ti	*	Z	1	1	1	1	1	1	1
	MC_3	$T_1T_2T_3$	SP−1	PCH	1	0	0	1	1	1	1
	MC_4	$T_1T_2T_3$	SP−2	PCL	1	0	0	1	1	1	1
	MC_5	$T_1T_2T_3$	I, Vector	Data	0	1	0	1	1	1	1
	MC_6	$T_1T_2T_3$	I, Vector+1	Data	0	1	0	1	1	1	1

APPENDIX E. OPERATING MODES

E.1 Request Acceptance in Each Operating Mode

Table E-1 Request Acceptance

Request		Normal Operation (CPU mode) (I/O Stop mode)	Wait State	Refresh Cycle	Interrupt Acknowledge Cycle	DMA Cycle	Bus Release Mode	Sleep mode	System Stop Mode
$\overline{WAIT}$		Accepted	Accepted	Not accepted	Accepted	Accepted	Not accepted	Not accepted	Not accepted
Refresh Request (Request of Refresh by the on-chip Refresh Controller)		Refresh cycle begins at the end of MC	Not accepted	Not accepted	Refresh cycle begins at the end of MC	Refresh cycle begins at the end of MC	Not accepted	Not accepted	Not accepted
$\overline{DREQ_0}$ $\overline{DREQ_1}$		DMA cycle begins at the end of MC	DMA cycle begins at the end of MC	Accepted If refresh cycle precedes: DMA cycle begins at the end of one MC	Accepted DMA cycle begins at the end of MC	Accepted Refer to Section 10 "DMA Controller" for details.	Accepted *, After bus frelease cycle, DMA cycle begins at the end of one MC	Not accepted	Not accepted
$\overline{BUSREQ}$		Bus is released at the end of MC	Not accepted	Not accepted	Bus is released at the end of MC	Bus is released at the end of MC	Continue bus release mode.	Accepted	Accepted
Interrupt	$\overline{INT_0}$, $\overline{INT_1}$, $\overline{INT_2}$	Accepted after executing the current instruction.	Accepted after executing the current instruction	Not accepted	Not accepted	Not accepted	Not accepted	Accepted Return from sleep mode to normal operation.	Accepted Return from system stop mode to normal operation.
	Internal I/O Interrupt	Accepted after executing the current instruction.	Accepted after executing the current instruction	Not accepted	Not accepted	Not accepted	Not accepted	Accepted Return from sleep mode to normal operation.	Not accepted
	$\overline{NMI}$	Accepted after executing the current instruction.	Accepted after executing the current instruction	Not accepted	Not accepted Interrupt acknowledge cycle precedes. $\overline{NMI}$ is accepted after executing the next instruction.	Accepted DMA cycle stops.	Not accepted	Accepted Return from sleep mode to normal operation.	Acceptable Return from system stop mode to normal operation.

Notes *: not acceptable when DMA Request is in level sense.
MC: Machine Cycle

E.2 Request Priority

The HD647180X has the following three types of requests.

Type 1: To be accepted in specified state .... WAIT

Type 2: To be accepted in each machine cycle .... Refresh Req.
DMA Req.
Bus Req.

Type 3: To be accepted in each instruction .... Interrupt Req.

Type 1, type 2, and type 3 request priority is as follows:

Highest priority Type 1 > Type 2 > Type 3 Lowest priority

Type 2 request priority is as follows:

Highest priority Bus Req. > Refresh Req. > DMA Req. Lowest priority

Note : If Bus Req. and Refresh Req. occurs simultaneously, Bus Req. is accepted but Refresh Req. is cleared.

Refer to "Section 8, Interrupts" for type 3 request priority.

E.3 Operation Mode Transition

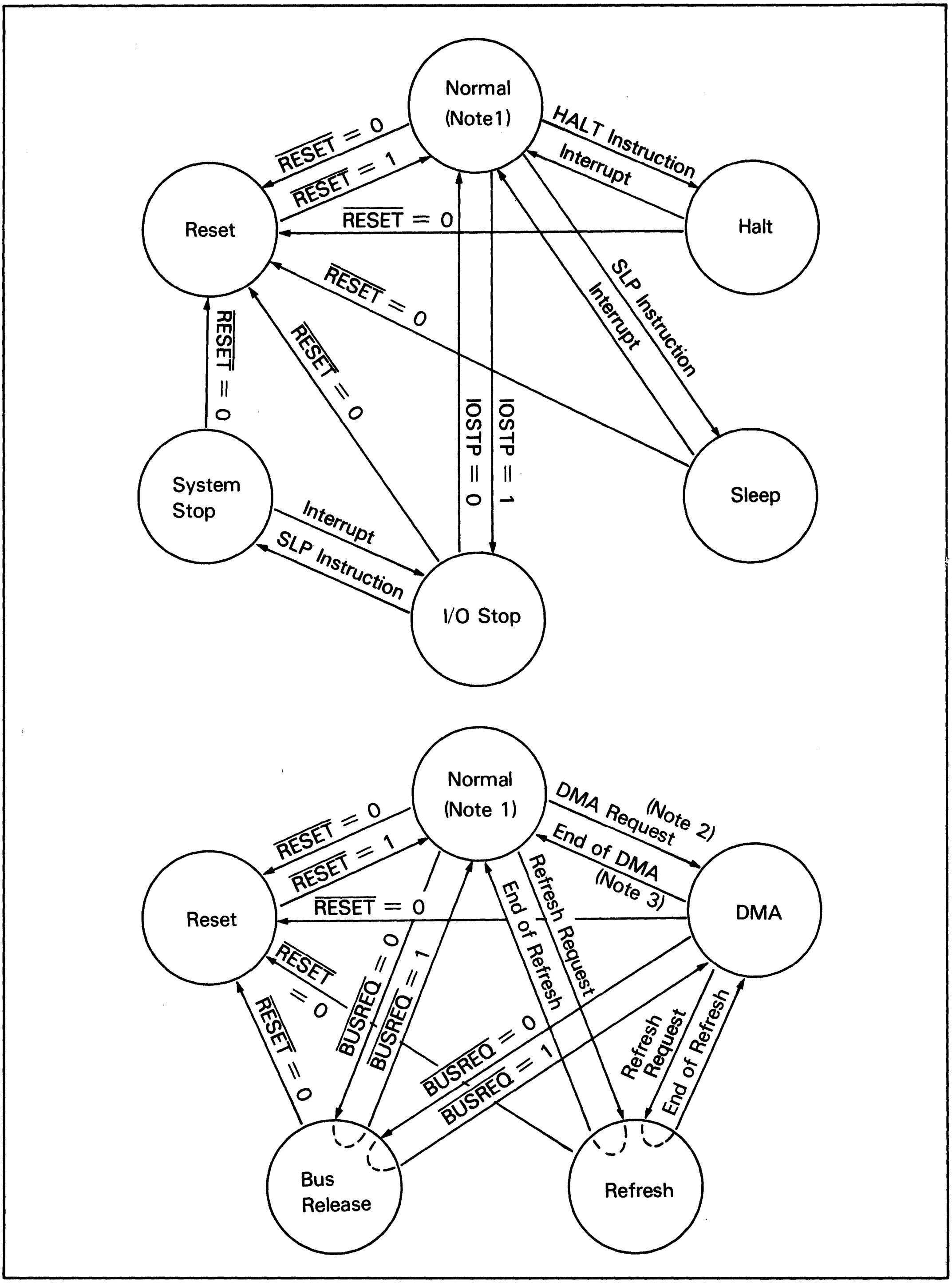


Figure E-1. Operation Mode Transitions

Notes : 1. Normal: CPU executes instructions normally in normal mode.

2. DMA request: DMA is requested in the following cases.
   (1) $\overline{DREQ_0}$, $\overline{DREQ_1}$ = 0 (memory to/from (memory-mapped) I/O DMA transfer)
   (2) DE0 = 1 (memory to/from memory DMA transfer)

3. DMA end: DMA ends in the following cases.
   (1) $\overline{DREQ_0}$, $\overline{DREQ_1}$ = 1 (memory to/from (memory-mapped) I/O DMA transfer)
   (2) BCR0, BCR1 = 0000H (all DMA transfers)
   (3) $\overline{NMI}$ = 0 (all DMA transfers)

The following operation mode transitions are also possible.

$$\text{Halt} \rightleftarrows \left\{ \begin{array}{l} \text{DMA} \\ \text{Refresh} \\ \text{Bus Release} \end{array} \right\}$$

$$\text{I/O Stop} \rightleftarrows \left\{ \begin{array}{l} \text{DMA} \\ \text{Refresh} \\ \text{Bus Release} \end{array} \right\}$$

$$\text{Sleep} \rightleftarrows \text{Bus Release}$$

$$\text{System Stop} \rightleftarrows \text{Bus Release}$$

E.4 Status Signals

Table E-2. shows pin outputs in each operating mode.

Table E-2 Pin Outputs

Mode		$\overline{\mathrm{LIR}}$	$\overline{\mathrm{ME}}$	$\overline{\mathrm{IOE}}$	$\overline{\mathrm{RD}}$	$\overline{\mathrm{WR}}$	$\overline{\mathrm{REF}}$	$\overline{\mathrm{HALT}}$	$\overline{\mathrm{BUSACK}}$	ST	Address Bus	Data Bus
CPU operation	Opcode Fetch (1st opcode)	0	0	1	0	1	1	1	1	0	A	In
	Opcode Fetch (except 1st opcode)	0	0	1	0	1	1	1	1	1	A	In
	Memory Read	1	0	1	0	1	1	1	1	1	A	In
	Memory Write	1	0	1	1	0	1	1	1	1	A	Out
	I/O Read	1	1	0	0	1	1	1	1	1	A	In
	I/O Write	1	1	0	1	0	1	1	1	1	A	Out
	Internal Operation	1	1	1	1	1	1	1	1	1	A	In
Refresh		1	0	1	1	1	0	1	1	*	A	In
Interrupt Acknow-ledge Cycle (1st machine cycle)	$\overline{\mathrm{NMI}}$	0	0	1	0	1	1	1	1	0	A	In
	$\overline{\mathrm{INT}}_0$	0	1	0	1	1	1	1	1	0	A	In
	$\overline{\mathrm{INT}}_1$, $\overline{\mathrm{INT}}_2$ & Internal Interrupts	1	1	1	1	1	1	1	1	0	A	In
Bus Release		1	Z	Z	Z	Z	1	1	0	*	Z	In
Halt		0	0	1	0	1	1	0	1	0	A	In
Sleep		1	1	1	1	1	1	0	1	1	1	In
Internal DMA	Memory Read	1	0	1	0	1	1	*	1	0	A	IN
	Memory Write	1	0	1	1	0	1	*	1	0	A	Out
	I/O Read	1	1	0	0	1	1	*	1	0	A	In
	I/O Write	1	1	0	1	0	1	*	1	0	A	Out
Reset		1	1	1	1	1	1	1	1	1	Z	In

Note 1 : High
0 : Low
A : Programmable
Z : High Impedance
In : Input
Out : Output
* : Invalid

APPENDIX F. INTERNAL I/O REGISTERS

By programming IOA7 in the I/O control register, internal I/O register addresses are relocatable within ranges from 0000H to 00FFH in the I/O address space.

Register	Mnemonic	Address	Remarks
ASCI Control Register A Channel 0	(CNTLA0)	0 0	see below
ASCI Control Register A Channel 1	(CNTLA1)	0 1	see below
ASCI Control Register B Channel 0	(CNTLB0)	0 2	see below

ASCI Control Register A Channel 0 (CNTLA0) — 0 0

bit	MPE	RE	TE	$\overline{RTS0}$	MPBR/EFR	MOD2	MOD1	MOD0
During reset	0	0	0	1	invalid	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- MOD2, MOD1, MOD0 — Mode Selection
- MPBR/EFR — Multi Processor Bit Receive/ Error Flag Reset
- $\overline{RTS0}$ — Request To Send
- TE — Transmit Enable
- RE — Receive Enable
- MPE — Multi Processor Enable

ASCI Control Register A Channel 1 (CNTLA1) — 0 1

bit	MPE	RE	TE	CKA1D	MPBR/EFR	MOD2	MOD1	MOD0
During reset	0	0	0	1	invalid	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- MOD2, MOD1, MOD0 — Mode Selection
- MPBR/EFR — Multi Processor Bit Receive/ Error Flag Reset
- CKA1D — CKA1 Disable
- TE — Transmit Enable
- RE — Receive Enable
- MPE — Multi Processor Enable

MOD2, 1, 0:

0 0 0	Start + 7 bit Data + 1 Stop
0 0 1	Start + 7 bit Data + 2 Stop
0 1 0	Start + 7 bit Data + Parity + 1 Stop
0 1 1	Start + 7 bit Data + Parity + 2 Stop
1 0 0	Start + 8 bit Data + 1 Stop
1 0 1	Start + 8 bit Data + 2 Stop
1 1 0	Start + 8 bit Data + Parity + 1 Stop
1 1 1	Start + 8 bit Data + Parity + 2 Stop

ASCI Control Register B Channel 0 (CNTLB0) — 0 2

bit	MPBT	MP	$\overline{CTS}$/PS	PEO	DR	SS2	SS1	SS0
During reset	invalid	0	*	0	0	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- SS2, SS1, SS0 — Clock Source and Speed Select
- DR — Divide Ratio
- PEO — Parity Even or Odd
- $\overline{CTS}$/PS — Clear To Send/Prescale
- MP — Multi Processor
- MPBT — Multi Processor Bit Transmit

* $\overline{CTS}$: Depends on the condition of $\overline{CTS}$ Pin.
PS: Cleared to 0.

(continued)

Register	Mnemonic	Address	Remarks
ASCI Control Register B Channel 1	(CNTLB1)	0 3	see below
ASCI Status Register Channel 0	(STAT0)	0 4	see below
ASCI Status Register Channel 1	(STAT1)	0 5	see below

ASCI Control Register B Channel 1 (CNTLB1)

bit	MPBT	MP	$\overline{CTS}$/PS	PEO	DR	SS2	SS1	SS0
During reset	invalid	0	0	0	0	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SS2, SS1, SS0 — Clock Source and Speed Select
DR — Divide Ratio
PEO — Parity Even or Odd
$\overline{CTS}$/PS — Clear To Send/Prescale
MP — Multi Processor
MPBT — Multi Processor Bit Transmit

General divide ratio	PS = 0 (divide ratio = 10)		PS = 1 (divide ratio = 30)	
SS2,1,0	DR = 0(× 16)	DR = 1 (× 64)	DR = 0(× 16)	DR = 1(× 64)
0 0 0	ϕ÷ 160	ϕ÷ 640	ϕ÷ 480	ϕ÷ 1920
0 0 1	÷ 320	÷ 1280	÷ 960	÷ 3840
0 1 0	÷ 640	÷ 2560	÷ 1920	÷ 7680
0 1 1	÷ 1280	÷ 5120	÷ 3840	÷ 15360
1 0 0	÷ 2560	÷ 10240	÷ 7680	÷ 30720
1 0 1	÷ 5120	÷ 20480	÷ 15360	÷ 61440
1 1 0	÷ 10240	÷ 40960	÷ 30720	÷ 122880
1 1 1	External clock (frequency < ϕ ÷ 40)			

ASCI Status Register Channel 0 (STAT0)

bit	RDRF	OVRN	PE	FE	RIE	$\overline{DCD0}$	TDRE	TIE
During reset	0	0	0	0	0	*	**	0
R/W	R	R	R	R	R/W	R	R	R/W

TIE — Transmit Interrupt Enable
TDRE — Transmit Data Register Empty
$\overline{DCD0}$ — Data Carrier Detect
RIE — Receive Interrupt Enable
FE — Framing Error
PE — Parity Error
OVRN — Over Run Error
RDRF — Receive Data Register Full

**

$\overline{CTS_0}$ Pin	TDRE
L	1
H	0

* $\overline{DCD_0}$: Depends on the condition of $\overline{DCD_0}$ Pin.

ASCI Status Register Channel 1 (STAT1)

bit	RDRF	OVRN	PE	FE	RIE	CTS1E	TDRE	TIE
During reset	0	0	0	0	0	0	1	0
R/W	R	R	R	R	R/W	R/W	R	R/W

TIE — Transmit Interrupt Enable
TDRE — Transmit Data Register Empty
CTS1E — $\overline{CTS1}$ Enable
RIE — Receive Interrupt Enable
FE — Framing Error
PE — Parity Error
OVRN — Over Run Error
RDRF — Receive Data Register Full

(continued)

Register	Mnemonic	Address	Remarks
ASCI Transmit Data Register Channel 0	(TDR0)	0 6	
ASCI Transmit Data Register Channel 1	(TDR1)	0 7	
ASCI Receive Data Register Channel 0	(TSR0)	0 8	
ASCI Receive Data Register Channel 1	(TSR1)	0 9	
CSI/O Control Register	(CNTR)	0 A	

bit	EF	EIE	RE	TE	—	SS2	SS1	SS0
During reset	0	0	0	0	1	1	1	1
R/W	R	R/W	R/W	R/W		R/W	R/W	R/W

SS2, SS1, SS0 — Speed Select
TE — Transmit Enable
RE — Receive Enable
EIE — End Interrupt Enable
EF — End Flag

SS2,1,0	Baud Rate	SS2,1,0	Baud Rate
000	$\phi \div 20$	100	$\phi \div 320$
001	$\div 40$	101	$\div 640$
010	$\div 80$	110	$\div 1280$
011	$\div 160$	111	External (frequency $< \div 20$)

Register	Mnemonic	Address	Remarks
CSI/O Transmit/Receive Data Register	(TRDR)	0 B	
Timer Data Register Channel 0L	(TMDR0L)	0 C	
Timer Data Register Channel 0H	(TMDR0H)	0 D	
Timer Reload Register Channel 0L	(RLDR0L)	0 E	
Timer Reload Register Channel 0H	(RLDR0H)	0 F	
Timer Control Register	(TCR)	1 0	

bit	TIF1	TIF0	TIE1	TIE0	TOC1	TOC0	TDE1	TDE0
During reset	0	0	0	0	0	0	0	0
R/W	R	R	R/W	R/W	R/W	R/W	R/W	R/W

TDE1, TDE0 — Timer Down Count Enable 1,0
TOC1, TOC0 — Timer Output Control 1,0
TIE1, TIE0 — Timer Interrupt Enable 1,0
TIF1, TIF0 — Timer Interrupt Flag 1,0

TOC1,0	TOUT1
00	1
01	Toggle
10	0
11	1

(continued)

<table>
<tr><th>Register</th><th>Mnemonic</th><th>Address</th><th>Remarks</th></tr>
<tr><td>Timer Data Register Channel 1L</td><td>(TMDR1L)</td><td>1 4</td><td></td></tr>
<tr><td>Timer Data Register Channel 1H</td><td>(TMDR1H)</td><td>1 5</td><td></td></tr>
<tr><td>Timer Reload Register Channel 1L</td><td>(RLDR1L)</td><td>1 6</td><td></td></tr>
<tr><td>Timer Reload Register Channel 1H</td><td>(RLDR1H)</td><td>1 7</td><td></td></tr>
<tr><td>Free Running Counter</td><td>(FRC)</td><td>1 8</td><td>Read only</td></tr>
<tr><td>DMA Source Address Register Channel 0L</td><td>(SAR0L)</td><td>2 0</td><td></td></tr>
<tr><td>DMA Source Address Register Channel 0H</td><td>(SAR0H)</td><td>2 1</td><td></td></tr>
<tr><td>DMA Source Address Register Channel 0B</td><td>(SAR0B)</td><td>2 2</td><td>Bits 0-3 are used for SAR0B.<table>
<tr><th>A_{19}</th><th>A_{18}</th><th>A_{17}</th><th>A_{16}</th><th>DMA Transfer Request</th></tr>
<tr><td>X</td><td>X</td><td>0</td><td>0</td><td>$\overline{DREQ_0}$ (external)</td></tr>
<tr><td>X</td><td>X</td><td>0</td><td>1</td><td>RDR0 (ASCI0)</td></tr>
<tr><td>X</td><td>X</td><td>1</td><td>0</td><td>RDR1 (ASCI1)</td></tr>
<tr><td>X</td><td>X</td><td>1</td><td>1</td><td>Not Used</td></tr>
</table></td></tr>
<tr><td>DMA Destination Address Register Channel 0L</td><td>(DAR0L)</td><td>2 3</td><td></td></tr>
<tr><td>DMA Destination Address Register Channel 0H</td><td>(DAR0H)</td><td>2 4</td><td></td></tr>
<tr><td>DMA Destination Address Register Channel 0B</td><td>(DAR0B)</td><td>2 5</td><td>Bits 0-3 are used for DAR0B.<table>
<tr><th>A_{19}</th><th>A_{18}</th><th>A_{17}</th><th>A_{16}</th><th>DMA Transfer Request</th></tr>
<tr><td>X</td><td>X</td><td>0</td><td>0</td><td>$\overline{DREQ_0}$ (external)</td></tr>
<tr><td>X</td><td>X</td><td>0</td><td>1</td><td>TDR0 (ASCI0)</td></tr>
<tr><td>X</td><td>X</td><td>1</td><td>0</td><td>TDR1 (ASCI1)</td></tr>
<tr><td>X</td><td>X</td><td>1</td><td>1</td><td>Not Used</td></tr>
</table></td></tr>
<tr><td>DMA Byte Count Register Channel 0L</td><td>(BCR0L)</td><td>2 6</td><td></td></tr>
<tr><td>DMA Byte Count Register Channel 0H</td><td>(BCR0H)</td><td>2 7</td><td></td></tr>
<tr><td>DMA Memory Address Register Channel 1L</td><td>(MAR1L)</td><td>2 8</td><td></td></tr>
<tr><td>DMA Memory Address Register Channel 1H</td><td>(MAR1H)</td><td>2 9</td><td></td></tr>
<tr><td>DMA Memory Address Register Channel 1B</td><td>(MAR1B)</td><td>2 A</td><td>Bits 0-3 are used for MAR1B.</td></tr>
<tr><td>DMA I/O Address Register Channel 1L</td><td>(IAR1L)</td><td>2 B</td><td></td></tr>
<tr><td>DMA I/O Address Register Channel 1H</td><td>(IAR1H)</td><td>2 C</td><td></td></tr>
</table>

(continued)

Register	Mnemonic	Address	Remarks
DMA Byte Count Register Channel 1L	(BCR1L)	2 E	
DMA Byte Count Register Channel 1H	(BCR1H)	2 F	
DMA Status Register	(DSTAT)	3 0	see below
DMA Mode Register	(DMODE)	3 1	see below

DMA Status Register (DSTAT)

bit	DE1	DE0	$\overline{DWE1}$	$\overline{DWE0}$	DIE1	DIE0	–	DME
During reset	0	0	1	1	0	0	1	0
R/W	R/W	R/W	W	W	R/W	R/W		R

DE1, DE0 — DMA Enable ch 1,0
$\overline{DWE1}$, $\overline{DWE0}$ — DMA Enable Bit Write Enable 1,0
DIE1, DIE0 — DMA Interrupt Enable 1,0
DME — DMA Master Enable

DMA Mode Register (DMODE)

bit	–	–	DM1	DM0	SM1	SM0	MMOD	–
During reset	1	1	0	0	0	0	0	1
R/W			R/W	R/W	R/W	R/W	R/W	

DM1, DM0 — Ch 0 Destination Mode 1, 0
SM1, SM0 — Ch 0 Source Mode 1,0
MMOD — Memory Mode Select

DM1, 0	Destination	Address
0 0	M	DAR0+1
0 1	M	DAR0−1
1 0	M	DAR0 fixed
1 1	I/O	DAR0 fixed

SM1, 0	Source	Address
0 0	M	SAR0+1
0 1	M	SAR0−1
1 0	M	SAR0 fixed
1 1	I/O	SAR0 fixed

MMOD	Mode
0	Cycle Steal Mode
1	Burst Mode

(continued)

Register	Mnemonic	Address	Remarks
DMA/Wait Control Resiter	(DCNTL)	3 2	see below
Interrupt Vector Low Register	(IL)	3 3	see below
INT/TRAP Control Register	(ITC)	3 4	see below
Refresh Control Register	(RCR)	3 6	see below

DMA/Wait Control Resiter (DCNTL) — Address 3 2

bit	MWI1	MWI0	IWI1	IWI0	DMS1	DMS0	DIM1	DIM0
During reset	1	1	1	1	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- MWI1, MWI0: Memory Wait Insertion
- IWI1, IWI0: I/O Wait Insertion
- DMS1, DMS0: $\overline{DREQi}$ Select, i = 1,0
- DIM1, DIM0: DMA Ch 1 I/O Memory Mode Select

MWI1,0	Number of wait states
0 0	0
0 1	1
1 0	2
1 1	3

IWI1,0	Number of wait states
0 0	0
0 1	2
1 0	3
1 1	4

DMSi	Sense
1	Edge sense
0	Level sense

DIM1,0	Transfer Mode	Address Increment/Decrement	
0 0	M→I/O	MAR1+1	IAR1 fixed
0 1	M→I/O	MAR1−1	IAR1 fixed
1 0	I/O→M	IAR1 fixed	MAR1+1
1 1	I/O→M	IAR1 fixed	MAR1−1

Interrupt Vector Low Register (IL) — Address 3 3

bit	IL7	IL6	IL5	–	–	–	–	–
During reset	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W					

- IL7, IL6, IL5: Interrupt Vector Low

INT/TRAP Control Register (ITC) — Address 3 4

bit	TRAP	UFO	–	–	–	ITE2	ITE1	ITE0
During reset	0	0	1	1	1	0	0	1
R/W	R/W	R				R/W	R/W	R/W

- TRAP: TRAP
- UFO: Undefined Fetch Object
- ITE2, ITE1, ITE0: $\overline{INT}$ Enable 2,1,0

Refresh Control Register (RCR) — Address 3 6

bit	REFE	REFW	–	–	–	–	CYC1	CYC0
During reset	1	1	1	1	1	1	0	0
R/W	R/W	R/W					R/W	R/W

- REFE: Refresh Enable
- REFW: Refresh Wait State
- CYC1, CYC0: Cycle Select

CYC1,0	Interval of Refresh Cycle
0 0	10 States
0 1	20
1 0	40
1 1	80

(continued)

Register	Mnemonic	Address	Remarks
MMU Common Base Register	(CBR)	3 8	

bit	CB7*	CB6	CB5	CB4	CB3	CB2	CB1	CB0
During reset	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

MMU Common Base Register

Register	Mnemonic	Address	Remarks
MMU Bank Base Register	(BBR)	3 9	

bit	BB7*	BB6	BB5	BB4	BB3	BB2	BB1	BB0
During reset	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

MMU Bank Base Register

Register	Mnemonic	Address	Remarks
MMU Common/Bank Area Register	(CBAR)	3 A	

bit	CA3	CA2	CA1	CA0	BA3	BA2	BA1	BA0
During reset	1	1	1	1	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

CA3–CA0: MMU Common Area Register

BA3–BA0: MMU Bank Area Register

Register	Mnemonic	Address	Remarks
Operation Mode Control Register	(OMCR)	3 E	

bit	LIRE	$\overline{\text{LIRTE}}$	$\overline{\text{IOC}}$	–	–	–	–	–
During reset	1	1	1	1	1	1	1	1
R/W	R/W	W	R/W					

$\overline{\text{IOC}}$: I/O Compatibility

$\overline{\text{LIRTE}}$: $\overline{\text{LIR}}$ Temporary Enable

LIRE: $\overline{\text{LIR}}$ Enable

Register	Mnemonic	Address	Remarks
I/O Control Register	(ICR)	3 F	

bit	IOA7	–	IOSTP	–	–	–	–	–
During reset	0	1	0	1	1	1	1	1
R/W	R/W		R/W					

IOSTP: I/O Stop

IOA7: I/O Address

Register	Mnemonic	Address	Remarks
Timer 2 Free-Running Counter L	(T2FRCL)	4 0	

bit	T2FRCL7	T2FRCL6	T2FRCL5	T2FRCL4	T2FRCL3	T2FRCL2	T2FRCL1	T2FRCL0
During reset	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Register	Mnemonic	Address	Remarks
Timer 2 Free-Running Counter H	(T2FRCH)	4 1	

bit	T2FRCH7	T2FRCH6	T2FRCH5	T2FRCH4	T2FRCH3	T2FRCH2	T2FRCH1	T2FRCH0
During reset	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Register	Mnemonic	Address	Remarks
Timer 2 Output Compare Register 1L	(T2OCR1L)	4 2	

bit	T2OCR1L7	T2OCR1L6	T2OCR1L5	T2OCR1L4	T2OCR1L3	T2OCR1L2	T2OCR1L1	T2OCR1L0
During reset	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Register	Mnemonic	Address	Remarks
Timer 2 Output Compare Register 1H	(T2OCR1H)	4 3	

bit	T2OCR1H7	T2OCR1H6	T2OCR1H5	T2OCR1H4	T2OCR1H3	T2OCR1H2	T2OCR1H1	T2OCR1H0
During reset	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Register	Mnemonic	Address	Remarks								
Timer 2 Output Compare Register 2L	(T2OCR2L)	4 4	bit	T2OCR2L7	T2OCR2L6	T2OCR2L5	T2OCR2L4	T2OCR2L3	T2OCR2L2	T2OCR2L1	T2OCR2L0
			During reset	1	1	1	1	1	1	1	1
			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Timer 2 Output Compare Register 2H	(T2OCR2H)	4 5	bit	T2OCR2H7	T2OCR2H6	T2OCR2H5	T2OCR2H4	T2OCR2H3	T2OCR2H2	T2OCR2H1	T2OCR2H0
			During reset	1	1	1	1	1	1	1	1
			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Timer 2 Input Capture Register L	(T2ICRL)	4 6	bit	T2ICRL7	T2ICRL6	T2ICRL5	T2ICRL4	T2ICRL3	T2ICRL2	T2ICRL1	T2ICRL0
			During reset	0	0	0	0	0	0	0	0
			R/W	R	R	R	R	R	R	R	R
Timer 2 Input Capture Register H	(T2ICRH)	4 7	bit	T2ICRH7	T2ICRH6	T2ICRH5	T2ICRH4	T2ICRH3	T2ICRH2	T2ICRH1	T2ICRH0
			During reset	0	0	0	0	0	0	0	0
			R/W	R	R	R	R	R	R	R	R
Timer 2 Control/status Register 1	(T2CSR1)	4 8	bit	ICF	OCF1	TOF2	EICI	EOCI1	ETOI	IEDG	OLVL1
			During reset	0	0	0	0	0	0	0	0
			R/W	R	R	R	R/W	R/W	R/W	R/W	R/W
Timer 2 Control/status Register 2	(T2CSR2)	4 9	bit	ICF	OCF1	OCF2	–	EOCI2	OLVL2	–	–
			During reset	0	0	0	1	0	0	0	0
			R/W	R	R	R	–	R/W	R/W	R/W	R/W
Comparator Control/status Register	(CCSR)	5 0	bit	RBIT	–	REF2	REF1	REF0	AIN2	AIN1	AIN0
			During reset	Note	1	1	0	1	1	0	0
			R/W	R		R/W	R/W	R/W	R/W	R/W	R/W
			Note: Undefined until the first comparison result is stored								
RAM Control Register	(RMCR)	5 1	bit	RMCR3	RMCR2	RMCR1	RMCR0	–	–	–	–
			During reset	0	0	0	0	1	1	1	1
			R/W	R/W	R/W	R/W	R/W				

Register	Mnemonic	Address	Remarks								
Port A Disable Register	(DERA)	5 3	bit	TEND1E	DREQ1E	CKSE	RXSE	TXSE	CKAIE	RXAIE	TAXIE
			During reset	0	0	0	0	0	0	0	0
			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Port A Input Data Register	(IDRA)	6 0	bit	IDRA7	IDRA6	IDRA5	IDRA4	IDRA3	IDRA2	IDRA1	IDRA0
			During reset	(Note 1)							
			R/W	R	R	R	R	R	R	R	R
Port A Output Data Register	(ODRA)	6 0	bit	ODRA7	ODRA6	ODRA5	ODRA4	ODRA3	ODRA2	ODRA1	ODRA0
			During reset	(Note 2)							
			R/W	W	W	W	W	W	W	W	W
Port B Input Data Register	(IDRB)	6 1	bit	IDRB7	IDRB6	IDRB5	IDRB4	IDRB3	IDRB2	IDRB1	IDRB0
			During reset	(Note 1)							
			R/W	R	R	R	R	R	R	R	R
Port B Output Data Register	(ODRB)	6 1	bit	ODRB7	ODRB6	ODRB5	ODRB4	ODRB3	ODRB2	ODRB1	ODRB0
			During reset	(Note 2)							
			R/W	W	W	W	W	W	W	W	W
Port C Input Data Register	(IDRC)	6 2	bit	IDRC7	IDRC6	IDRC5	IDRC4	IDRC3	IDRC2	IDRC1	IDRC0
			During reset	(Note 1)							
			R/W	R	R	R	R	R	R	R	R
Port C Output Data Register	(ODRC)	6 2	bit	ODRC7	ODRC6	ODRC5	ODRC4	ODRC3	ODRC2	ODRC1	ODRC0
			During reset	(Note 2)							
			R/W	W	W	W	W	W	W	W	W
Port D Input Data Register	(IDRD)	6 3	bit	IDRD7	IDRD6	IDRD5	IDRD4	IDRD3	IDRD2	IDRD1	IDRD0
			During reset	(Note 1)							
			R/W	R	R	R	R	R	R	R	R
Port D Output Data Register	(ODRD)	6 3	bit	ODRD7	ODRD6	ODRD5	ODRD4	ODRD3	ODRD2	ODRD1	ODRD0
			During reset	(Note 2)							
			R/W	W	W	W	W	W	W	W	W
Port E Input Data Register	(IDRE)	6 4	bit	IDRE7	IDRE6	IDRE5	IDRE4	IDRE3	IDRE2	IDRE1	IDRE0
			During reset	(Note 1)							
			R/W	R	R	R	R	R	R	R	R
Port E Output Data Register	(ODRE)	6 4	bit	ODRE7	ODRE6	ODRE5	ODRE4	ODRE3	ODRE2	ODRE1	ODRE0
			During reset	(Note 2)							
			R/W	W	W	W	W	W	W	W	W
Port F Input Data Register	(IDRF)	6 5	bit	IDRF7	IDRF6	IDRF5	IDRF4	IDRF3	IDRF2	IDRF1	IDRF0
			During reset	(Note 1)							
			R/W	R	R	R	R	R	R	R	R
Port F Output Data Register	(ODRF)	6 5	bit	ODRF7	ODRF6	ODRF5	ODRF4	ODRF3	ODRF2	ODRF1	ODRF0
			During reset	(Note 2)							
			R/W	W	W	W	W	W	W	W	W

Note: 1. Fetches terminal status.
2. Undefined until data is written.

Register	Mnemonic	Address	Remarks								
Port G Input Data Register	(IDRG)	6 6	bit	–	–	IDRG5	IDRG4	IDRG3	IDRG2	IDRG1	IDRG0
			During reset	1	1	(Note 1)					
			R/W			R	R	R	R	R	R
			Note: 1. Fetches terminal status								
Port A Data Direction Register	(DDRA)	7 0	bit	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
			During reset	0	0	0	0	0	0	0	0
			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Port B Data Direction Register	(DDRB)	7 1	bit	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
			During reset	0	0	0	0	0	0	0	0
			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Port C Data Direction Register	(DDRC)	7 2	bit	DDRC7	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0
			During reset	0	0	0	0	0	0	0	0
			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Port D Data Direction Register	(DDRD)	7 3	bit	DDRD7	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0
			During reset	0	0	0	0	0	0	0	0
			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Port E Data Direction Register	(DDRE)	7 4	bit	DDRE7	DDRE6	DDRE5	DDRE4	DDRE3	DDRE2	DDRE1	DDRE0
			During reset	0	0	0	0	0	0	0	0
			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Port F Data Direction Register	(DDRF)	7 5	bit	DDRF7	DDRF6	DDRF5	DDRF4	DDRF3	DDRF2	DDRF1	DDRF0
			During reset	0	0	0	0	0	0	0	0
			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

HITACHI AMERICA, LTD.
SEMICONDUCTOR AND IC DIVISION

HEADQUARTERS

Hitachi, Ltd.
New Marunouchi Bldg., 5-1,
Marunouchi 1-chome
Chiyoda-ku, Tokyo 100, Japan
Tel: Tokyo (03) 212-1111
Telex: J22395, J22432, J24491,
J26375 (HITACHY)
Cable: HITACHY TOKYO

U.S. SALES OFFICE

Hitachi America, Ltd.
Semiconductor and IC Division
2210 O'Toole Avenue
San Jose, CA 95131
Tel: 408-435-8300
Telex: 17-1581
Twx: 910-338-2103
Fax: 408-435-2748
Fax: 408-435-2749
Fax: 408-435-2782

REGIONAL OFFICES

MID-ATLANTIC REGION

Hitachi America, Ltd.
1700 Galloping Hill Rd.
Kenilworth, NJ 07033
201/245-6400

NORTHEAST REGION

Hitachi America, Ltd.
5 Burlington Woods Drive
Burlington, MA 01803
617/229-2150

SOUTH CENTRAL REGION

Hitachi America, Ltd.
Two Lincoln Centre, Suite 865
5420 LBJ Freeway
Dallas, TX 75240
214/991-4510

NORTH CENTRAL REGION

Hitachi America, Ltd.
500 Park Blvd., Suite 415
Itasca, IL 60143
312/773-4864

NORTHWEST REGION

Hitachi America, Ltd.
2210 O'Toole Avenue
San Jose, CA 95131
408/435-2200

SOUTHWEST REGION

Hitachi America, Ltd.
18300 Von Karman Avenue, Suite 730
Irvine, CA 92715
714/553-8500

SOUTHEAST REGION

Hitachi America, Ltd.
4901 N.W. 17th Way, Suite 302
Fort Lauderdale, FL 33309
305/491-6154

DISTRICT OFFICES

- Hitachi America, Ltd.
  3800 W. 80th Street, Suite 1050
  Bloomington, MN 55431
  612/896-3444

- Hitachi America, Ltd.
  80 Washington St., Suite 302
  Poughkeepsie, NY 12601
  914/485-3400

- Hitachi America, Ltd.
  6 Parklane Blvd., #558
  Dearborn, MI 48126
  313/271-4410

- Hitachi America, Ltd.
  6161 Savoy Dr., Suite 850
  Houston, TX 77036
  713/974-0534

- Hitachi (Canadian) Ltd.
  2625 Queensview Dr.
  Ottawa, Ontario, Canada K2A 3Y4
  613/596-2777

- Hitachi America, Ltd.
  401 Harrison Oaks Blvd., Suite #317
  Cary, NC 27513
  919/481-3908

HITACHI®

We make things possible

Hitachi America, Ltd.
Semiconductor and IC Division
2210 O'Toole Avenue, San Jose, CA 95131
1-408-435-8300

7.5M

Printed in U.S.A.